O’ZBEKISTON RESPUBLIKASI AXBOROT TEXNOLOGIYALARI VA KOMMUNIKATSIYALARINI RIVOJLANTIRISH VAZIRLIGI

O’ZBEKISTON RESPUBLIKASI OLIY VA O’RTA - MAXSUS TA’LIM VAZIRLIGI

TOSHKENT AXBOROT TEXNOLOGIYALARI UNIVERSITETI URGANCH FILIALI

“C++ TILIDA DASTURLASH” FANI BO’YICHA

O’QUV-USLUBIY MAJMUA

Tuzuvchilar: F.Yusupov
O.T.Allamov
O.Q.Yusupov

URGANCH - **2016**

Tuzuvchilar: TATU Urganch filiali t.f.n. dots. F.Yusupov,
TATU Urganch filiali kat. o’qit. O.T.Allamov,
TATU Urganch filiali kat. o’qit. O.Q.Xo’jayev

MUNDARIJA

I. NAZARIY MASHG’ULOT MATERIALLARI..... 4

II. TAJRIBA MASHG’ULOTLARI MATERIALLARI 3913

III. MUSTAQIL TA’LIM MAVZULARI 58385

IV. GLOSSARIY 5879

V. ADABIYOTLAR RO’YXATI..... 59092

I.NAZARIY MASHG'ULOT MATERIALLARI

1-SEMESTR UCHUN

1-Ma'ruza. Dasturlashga kirish, dasturlasning asosiy tushunchalari

Reja:

1. C++ dasturlash tili haqida
2. C++ tilida standart funksiyalarning yozilishi usullari

C++ dasturlash tili haqida

C++ dasturlash tili C tiliga asoslangan. C esa o'z navbatida B va BCPL Tillaridan kelib chiqqan. BCPL 1967 yilda Martin Richards tomonidan tuzilgan va operatsion sistemalarni yozish uchun mo'ljallangan edi. Ken Thompson o'zining B tilida BCPL ning ko'p hossalarini kiritgan va B da UNIX operatsion sistemasining birinchi versiyalarini yozgan.

C tilini Dennis Ritchie B dan keltirib chiqardi va uni 1972 yili ilk bor Bell Laboratories da, DEC PDP-11 kompyuterida qo'lladi. C o'zidan oldingi B va BCPL tillarining juda ko'p muhim tomonlarini o'z ichiga olish bilan bir qatorda o'zgaruvchilarni tiplashtirdi va bir qator boshqa yangiliklarni kiritdi. Boshlanishda C asosan UNIX sistemalarida keng tarqaldi. Hozirda operatsion sistemalarning asosiy qismi C/C++ da yozilmoqda. C mashina arhitekturasiga bog'langan tildir. Lekin yahshi rejalashtirish orqali dasturlarni turli kompyuter platformalarida ishlaydigan qilsa bo'ladi.

1983 yilda, C tili keng tarqalganligi sababli, uni standartlash harakati boshlandi. Buning uchun Amerika Milliy Standartlar Komiteti (ANSI) qoshida X3J11 tehnik komitet tuzildi. Va 1989 yilda ushbu standart qabul qilindi.

Standartni dunyo bo'yicha keng tarqatish maqsadida 1990 yilda ANSI va Dunyo Standartlar Tashkiloti (ISO) hamkorlikda C ning ANSI/ISO 9899:1990 standartini qabul qilishdi.

Shu sababli C da yozilgan dasturlar kam miqdordagi o'zgarishlar yoki umuman o'zgarishlarsiz juda ko'p kompyuter platformalarida ishlaydi.

C++ 1980 yillar boshida Bjarne Stroustrup tomonidan C ga asoslangan tarzda tuzildi. C++ juda ko'p qo'shimchalarni o'z ichiga olgan, lekin eng asosiysi u ob'ektlar bilan dasturlashga imkon beradi.

Dasturlarni tez va sifatli yozish hozirgi kunda katta ahamiyat kasb etmoda. Buni ta'minlash uchun ob'ektli dasturlash g'oyasi ilgari surildi. Huddi 70-chi yillar boshida strukturali dasturlash kabi, programmalarni hayotdagi jismlarni modellashtiruvchi ob'ektlat orqali tuzish dasturlash sohasida inqilob qildi.

C++ funksiya va ob'ektlarning juda boy kutubhonasiga ega. Yani C++ da dasturlashni o'rganish ikki qismga bo'linadi. Birinchisi bu C++ ni o'zini o'rganish, ikkinchisi esa C++ ning standart kutubhonasidagi tayyor ob'ekt/funksiyalarni qo'llashni o'rganishdir.

C++ tilida yaxshi dastur tuzish uchun “aql, farosat va sabr” kerak bo’ladi. Bu til asosan tizim sathida dasturlovchilar uchun yaratilgan.

C/C++ algoritmik tilining alifbosi:

1. 26 ta lotin va 32 ta kirill harflari (katta va kichik);
2. 0 dan 9 gacha bo’lgan arab raqamlari;
3. Maxsus belgilar: - + * / : ; . , % ? ! = “” № < > { } [ ] ( ) $ # & ^ va h.k.

Dastur bajarilishi jarayonida o’z qiymatini o’zgartira oladigan kattaliklar o’zgaruvchilar deyiladi. O’zgaruvchilarning nomlari harfdan boshlanuvchi xarf va raqamlardan iborat bo’lishi mumkin. O’zguruvchilarni belgilashda katta va kichik harflarning farqlari bor. (A va a harflari 2 ta o’zgaruvchini bildiradi) Har bir o’zgaruvchi o’z nomiga, toifasiga, xotiradan egallagan joyiga va son qiymatiga ega bo’lishi kerak. O’zgaruvchiga murojaat qilish uning ismi orqali bo’ladi. O’zgaruvchi uchun xotiradan ajratilgan joyning tartib raqami uning adresi hisoblanadi. O’zgaruvchi ishlatilishidan oldin u aniqlangan bo’lishi lozim.

O’zgaruvchilarning son qiymatlari quyidagi ko’rinishda yoziladi:

- Butun toifali o’nlik sanoq tizimsida: ular faqat butun sondan iborat bo’ladilar. Masalan: 5; 76; -674 va h.k.
- Sakkizlik sanoq tizimsidagi sonlar: 0 (nol) dan boshlanib, 0 dan 7 gacha bo’lgan raqamlardan tashkil topadi. Masalan: *x*=0453217; *s*=077;
- O’n oltilik sanoq tizimsidagi sonlar: 0 (nol) dan boshlanadi va undan keyin x yoki X harfi keladi, so’ngra 0-9 raqamlari va a-f yoki A-F harflaridan iborat ketma-ketliklar bo’ladi. Masalan: 10 s.s.dagi 22 soni 8 s.s. da 026, 16 s.s.da 0x16 shaklida bo’ladi.
- Haqiqiy toifali sonlar: ular butun va kasr qismlardan iborat bo’ladilar. Masalan: 8,1; -12,59 va x.k. Haqiqiy toifali sonlarning bu ko’rinishi oddiy ko’rinish deyiladi. Juda katta yoki juda kichik haqiqiy toifali sonlarni darajali (eksponensional) formada yozish qulay. Masalan: $7{,}204*10^{12}$ yoki $3{,}567*10^{-11}$ kabi sonlar 7.204e+12 va 3.567e-11 ko’rinishda yoziladi.
- Simvolli konstantalar. Ular qatoriga dastur bajarilishi ‘ ‘ ichida qabul qilinadigan simvollar kiradi.

C/C++ tilida har qanday o’zgaruvchi ishlatilishidan oldin e’lon qilinishi kerak. E’lon qilish degani ularning toifalarini aniqlab qo’yish demakdir.

C++ tilida quyidagi toifali o’zgaruvchilar ishlatiladi:

- Butun toifali kichik sonlar yoki simvollar uchun: char uning o’zgarish intervali -128 dan +127 gacha yoki apostrof ichidagi ixtiyoriy 1ta simvol. Xotiradan 1 bayt joy oladi. Simvollar ASCII kodlariga mos keladi. (ASCII – American Standart Code for Information Interchange)
- Butun toifali o’zgaruvchilar: int. Masalan: int a, i, j ; Bu yerda dasturda ishlatilayotgan a, i, j o’zgaruvchilarining toifasi butun ekanligi ko’rsatildi. Bu toifadagi o’zgaruvchilar 2 bayt joy egallaydi. Ularning o’zgarish intervali: -32768 dan +32767 gacha; (Hozirgi 32 razryadli kompyuterlarda 4 bayt joy oladi va oralig’i 2 marta oshgan).

- Butun toifali katta (uzun) o’zgaruvchilar: long. Masalan: long s, s2, aa34; Bu toifadagi o’zgaruvchilar 4 bayt joy egallaydi. Ular – 2147483648 dan +2147483647 oraliqdagi sonlarni qabul qilishi mumkin.
- Ishorasiz butun o’zgaruvchilar: unsigned short – 2 bayt joy oladi, o’zgarish intervali 0 dan 65535 gacha; unsigned long – 4 bayt joy oladi, o’zgarish intervali: 0 dan 4294967295 gacha; unsigned char – 1 bayt joy oladi, o’zgarish chegarasi 0 dan 255 gacha.
- Haqiqiy toifadagi o’zgaruvchilar: float. Masalan: float a, b: Bu yerda dasturda ishlatilayotgan a, b o’zgaruvchilarining toifasi haqiqiy ekanligi ko’rsatilgan. Bu toifadagi o’zgaruvchilar 4 bayt joy egallaydi va qabul qilish chegarasi 10^{-38} dan 10^{+38} gacha.
- Katta yoki kichik qiymatli o’zgaruvchilarni ifoda etishda double toifasi ishlatiladi. Ular uchun 8 bayt joy ajratiladi va qabul qilish chegarasi 10^{-304} dan 10^{+304} gacha.
- Juda katta yoki juda kichik qiymatli o’zgaruvchilar uchun long double toifasi ishlatiladi, u 10 bayt joy oladi va qabul qilish chegarasi $3.4*10^{-4932}$ dan $1.1*10^{-4932}$ gacha.
- Qator toifasidagi o’zgaruvchilar uchun ham char toifasi belgilangan. Ular ham 1 bayt joy oladi va 0 dan 256 tagacha bo’lgan simvollar ketma-ketligidan iborat bo’lishi mumkin. Satr toifasidagi o’zgaruvchilar qo’shtirnoq (“) ichida yoziladi.

C++ tilida o’zgaruvchilarni inisializasiya qilish degan tushuncha ham mavjud. Inisializasiya qilish degani o’zgaruvchini e’lon qilish barobarida unga boshlang’ich qiymatini ham berish demakdir. Masalan: int a=5, b, s=-100; - a, b, s o’zgaruvchilari butun toifali ekanligi ko’rsatildi va a o’zgaruvchisiga 5 (a=5), s o’zgaruvchisiga esa –100 (s=-100) boshlang’ich qiymatlar berildi.

Dastur bajarilishi jarayonida o’z qiymatini o’zgartira olmaydigan kattaliklar o’zgarmaslar deyiladi. Masalan: x=1; bo’lsa keyinchalik x=x+5 deb yozib

bo’lmaydi. O’zgarmaslarni const so’zi bilan ko’rsatiladi. Maslan: const int x=95; float y=9.17; (const lar simvol yoki nol (NULL) bo’lishi xam mumkin.)

C++ tilida standart funksiyalarning yozilishi

Funksiya	Ifodalanishi	Funksiya	Ifodalanishi
Sin x	sin(x)	$\sqrt{x}$	sqrt(x); pow(x,1/2.)
Cos x	cos(x)	$\lvert x \rvert$	abs(x) yoki fabs(x)
Tg x	tan(x)	Arctan x	atan(x)
e^x	exp(x)	Arcsin x	asin(x) ?
Ln x	log(x)	Arccos x	acos(x) ?
Lg x	log10(x)	$\sqrt[3]{x^2}$	pow(x,2/3.)
x^a	pow(x,a)	Log_2x	log(x)/log(2)

Masalan: $\frac{-b+\sqrt{b^2-4ac}}{2a}$ → (-b + sqrt(b*b-4*a*c)/(2*a); yoki

(-b+pow(b*b-4*a*c,1/2.)/(2*a);

$e^{\sin x} + tg^2(x+3)$ → exp(sin(x)) + pow(tan(x+3),2);

k=(m*5)+((7 % n) / (9+x));

C++ tilidagi dastur quyidagi tarkibdan tashkil topadi:

1. Direktivalar – # include <file.h> direktiva – instruksiya degan ma’noni beradi. C++ tilida dasturning tuzilishiga, ya’ni ehtiyojiga qarab, kerakli direktivalar ishlatiladi. Ular < > belgisi orasida keltiriladi. Umuman olganda quyidagi direktivalar mavjud (jami 32 ta):
   - #include <stdio.h> - S da oddiy kiritish/chiqarish dasturi uchun. Bu yerda std - standart, i – input, o - output degani.

- #include <iostream.h> - C++ da kiritish/chiqarish uchun, oddiy amallar bajarilsa.
- #include <math.h> - standart funksiyalarni ishlatish uchun.
- #include <conio.h> - dasturning tashqi ko'rinishini shakllantirish uchun.
- #include <string.h> - satr toifasidagi o'zgaruvchilar ustida amallar bajarish uchun.
- #include <stdlib.h> - standart kutubxona fayllarini chaqirish uchun.
- #include <time.h> - kompyuter ichidagi soat qiymatlaridan foydalanish uchun.
- #include <graphics.h> - C++ tilining grafik imkoniyatlaridan foydalanish uchun.

Bu fayllar maxsus kutubxona e'lon fayllari hisoblanadilar va ular aloxida INCLUDE deb nomlanadigan papkada saqlanadiar. Hozirda C++ kutubxonasini yangilandi va undagi fayllarning nomlaridan .h (head – bosh ma'nosida) kengaytmasi olib tashlandi va oldiga c harfi qo'shildi (C dan qolgan 18 tasiga). Bu fayllarda funksiya prototoifalari, toifalari, o'zgaruvchilar, o'zgarmaslar ta'riflari yozilgan bo'ladi.

Direktivalar dasturni uni kompilyasiya qilinishidan oldin tekshirib chiqadi.

2. Makroslar - # define makro qiymati. Masalan:

#define y sin(x+25) – u = sin(x+25) qiymati berildi;

#define pi 3.1415 - pi = 3.1415

#define s(x) x*x - s(x) = x*x (; belgisi qo'yilmaydi)

Global o'zgaruvchilarni e'lon qilish. Asosiy funksiya ichida e'lon qilingan o'zgaruvchilar lokal, funksiyadan tashqarida e'lon qilinganlari esa global o'zgaruvchilar deyiladi. Global o'zgaruvchilar dastur davomida ishlaydi va xotiradan ma'lum joyni egallaydi. O'zgaruvchini bevosita ishlatishdan oldin e'lon qilsa ham bo'ladi, u holda o'z lokal bo'ladi. Global o'zgaruvchilar nomi lokal o'zgaruvchilar nomi bilan bir xil bo'lishi ham mumkin. Bunday holatda lokal

o’zgaruvchining qiymati joriy funksiya ichidagini qiymatini o’zgartiradi, funksiyadan chiqishi bilan global o’zgaruvchilar ishlaydi.

Asosiy funksiya - main ( ) hisoblanadi. Bu funksiya dasturda bo’lishi shart. Umuman olganda C++ dagi dastur funksiyalardan iborat deb qaraladi. main ( ) funksiyasi { boshlanadi va dastur oxirida berkitilishi shart } . main – asosiy degan ma’noni beradi. Bu funksiya oldida uning toifasi ko’rsatiladi. Agar main ( ) funksiyasi beradigan (qaytaradigan) javob oddiy so’z yoki gaplardan iborat bo’lsa, hech qanday natija qaytarmasa, void so’zi keltiriladi. main ( ) funksiyasi dastur tomonidan emas, balki OS tomonidan chaqiriladi. OSga qiymat qaytarish shart emas, chunki u bu qiymatdan foydalanmaydi. Shuning uchun main ( ) funksiyasining turini *void* deb ko’rsatganimiz ma’qul. Har bir funksiyaning o’z argumenti bo’ladi, shuning uchun main funksiya ( ) lari ichiga uning parametri keltiriladi. Ba’zan u bo’sh bo’lishi ham mumkin. Bu funksiyadan chiqish uchun odatda *return* operatori ishlatiladi. 0 (nol) qiymatining qaytarilishi operasion tizimga ushbu dastur normal bajarilib turganini bildiradi. return orqali qaytadigan qiymat toifasi funksiya e’lonidagi qaytish toifasi bilan bir xil bo’lishi kerak. Masalan int main ( ) va 0 (nol) qiymat butun toifalidir. Bu funksiyadan so’ng lokal o’zgaruvchilar, qism dasturlar, ularning haqiqiy parametrlar e’lon qilinadi. So’ngra dasturning asosiy operatorlari (kiritish/chiqarish, hisoblash va h.k.) yoziladi. Agar bu operatorlar murakkab toifali bo’lsalar, ularni alohida {} qavslarga olinadi. C++ tilida dastur kichik harflarda yoziladi. Ba’zi operatorlar katta harflar bilan kelishi mumkin, bunday xollarda ular alohida aytib o’tiladi. Operatorlar oxiriga ; belgisi qo’yiladi. Operatorlar bir qatorga ketma-ket yozilishi mumkin. Dasturda izohlar xam kelishi mumkin, ular /* ....*/ belgisi orasiga olinadi. Agar izoh bir qatorda tugasa, uni // belgisidan keyin yoziladi. Masalan:

main ( ) // C++ tilining asosiy funksiyasi

Tilda quyidagi amallardan foydalanish mumkin:

Arifmetik amallar: +, -, /, *, %. Barcha amallar odatdagidek bajariladi, faqat bo’lish amali butunga bo’lish bajariladi, ya’ni agar butun sonlar ustida

bajarilayotgan bo'lsa, natija doim butun bo'ladi, ya'ni kasr qism tashlab yuboriladi (9/5=1; vaxolanki 1,8 bo'lishi kerak). Shuning uchun surat yoki maxrajiga nuqta (.) qo'yilsa, natija ham xaqiqiy bo'ladi (9./5=1.8). % belgisi (modul operatori) esa butun sonni butun songa bo'lgandan hosil bo'ladigan qoldiqni bildiradi.

Masalan: 9 % 5=4

Taqqoslash amallari: = = (tengmi?); != (teng emas); < ; > ; >=; <=

Mantiqiy amallar: && (and) mantiqiy ko'paytirish; || (or) mantiqiy qo'shish; ! (not) mantiqiy inkor. Mantiqiy amallarni ixtiyoriy sonlar ustida bajarish mumkin. Agar javob rost bo'lsa, natija 1 bo'ladi, agar javob yolg'on bo'lsa, natija 0 bo'ladi. Umuman olganda 0 (nol) dan farqli javob rost deb qabul qilinadi.

Masalan: i>50 && j==24 yoki s1 < s2 && (s3>50 || s4<=20);

Yoki $6 \le x \le 10$ yozuvini x>=6 && x<=10 deb yoziladi

Qiymat berish amallari:

a=5; b = 2*c; x = y = z =1; a = (b = c)*d // 3=5 deb yozib bo'lmaydi

qabul qildim va almashtirdim deb nomalandigan amallar:

+ = : a+=b → a = a + b;

- = : a-=b → a = a - b;

* = : a*=b → a = a * b;

/ = : a/=b → a = a / b;

% = : a%=b → a = a % b;

- inkrement operatsiyasi (++) ikki ma'noda ishlatiladi: o'zgaruvchiga murojaat qilinganidan keyin uning qiymati 1 ga oshadi (a++ postfiks ko'rinishi) va o'zgaruvchining qiymati uning murojaat qilishdan oldin 1 ga oshadi (++a prefix ko'rinishi);
- dekrement operatsiyasi (--), xuddi inkrement operatsiyasi kabi, faqat kamaytirish uchun ishlatiladi. Masalan: s = a + b++ (a ga b ni qo'shib keyin b ning qiymatini 1 ga oshiradi); s = a+(--b) (b ning qiymatini 1 ga kamaytirib, keyin a ga qo'shadi).

Yuqoridagi standart funksiyalardan tashqari yana quyidagi funksiyalar ham ishlatiladi:

- ceil (x) - x ni x dan katta yoki unga teng bo'lgan eng kichik butun songacha yaxlitlash. Masalan: ceil (12.6) = 13.0; ceil (-2.4) = -2.0;
- floor (x) - x ni x dan kichik bo'lgan eng katta butun songacha yaxlitlash. Masalan: floor (4.8) = 4.0; floor (-15.9) = -16.0; floor(12.1) = 12; floor(-12.1)=-13;
- fmod (x,y) – x / y ning qoldig'ini kasr son ko'rinishida berish. Masalan: fmod(7.3, 1.7) = 0.5;

Muhokama savollari

1. Tilning alifbosi.
2. O'zgarmaslar
3. O'zgaruvchilarning toifalari.
4. Standart funksiyalarning ko'rinishi.

Nazorat savollari:

1. C/C++ tilida o'zgarmaslar.
2. C/C++ tilida o'zgaruvchilarning toifalari
3. Kompanovka bosqichlarini ayting.
4. Standart funksiyalarning qo'llanishi.
5. Ifodalar haqida tushuncha.
6. Dastur tuzilishi.
7. Preprosessor direktivalari
8. Identifikator, o'zgaruvchilar va o'zgarmaslar.
9. O'zgaruvchilarning oddiy toifalari.
10. Amallar va ifodalar.
11. Standart matematik funksiyalar.
12. Oddiy arifmetik ifodalar.

2-ma’ruza. C++ da dastlabki dasturni yozish

Ma’ruza rejasi:

2.1 Hello, World!

2.2 Obyektlar, qiymat va toifalar

2.3 Hisoblash

2.4 Xatoliklar

Kalit so’zlar: *toifalar, xatolik, sintaktik xatolik, dastur ishlashi davomidagi xatolik, testlash, kompilyatsiya paytidagi xatolik, talab, mantiqiy xato, turlar xatosi*.

2.1 Hello, World!

Dasturlar. Kompyuterni biron bir amalni bajprishga majburlash uchun, siz (yoki boshqalar) unga nima xoxlayotganingizni aniq, batafsil aytishingiz kerak.

Bundan tashqari, biz o’zimiz bajarishimiz kerak bo’lgan vazifa tavsifini olamiz, masalan, "yaqin oradagi kinoteatrga qanday borish mumkin" yoki "to’lqinli pechda go’shtni qanday qovurish mumkin". bunday tavsiflar va dasturlar orasidagi farq aniqlik darajasida aniqlanadi: insonlar sog’lom aql bilan qo’llanmani noaniqligini aniqlashga harakat qiladilar, kompyuter bunday qila olmaydi. Masalan, "yo’lak bo’ylab o’nga, zinadan yuqoriga, so’ngra chapga" - yuqori qavatdagi yuvinish xonasini topish imkonini beruvchi aniq qo’llanma. Biroq, agar siz bunday sodda qo’llanmaga qarasangiz, u holda ular grammatik noaniqligi va to’liq emasligini ko’rish mumkin. Masalan, siz stol atrofida o’tiribsiz va yuvinish xonasiga qanday o’tishni so’radingiz. Sizga javob beruvchi, o’rningizdan turishingizni, uni aylanib o’tishingizni va boshqalarni aytishi shart emas. Yana sizga hyech kim sanchqini stolga qo’yishingiz, zanadan ko’tarilayotganda chiroqni yoqishingiz kerakligini, yuvinish xonasiga kirish uchun eshikni ochish kerakligini maslahat bermaydi.

Qarama-qarshi holatda bunga kompyuterning aqli yetmaydi. Unga barchasini aniq va batafsil tavsiflash kerak. Kompyuterga qo’llanmani batafsil tavsiflash uchun,

o’ziga xos grammatikaga ega bo’lgan aniq belgilangan til hamda biz bajarishni xoxlayotgan faoliyatlarni barcha ko’rinishlari uchun yaxshi aniqlikdagi lug’at kerak bo’ladi. Bunday til dasturlash tili va ko’p qamrovli masalalarni yechish uchun ishlab chiqilgan - C++ dasturlash tili deb nomlanadi.

Kompyuterlar, dasturlar va dasturlash bo’yicha falsafiy qarashlar 1-maruzada kengroq yoritib berilgan. Bu yerda biz juda oddiy dasturdan boshlanadigan kodni hamda uning bajarilishi uchun kerak bo’ladigan bir qancha usullar va qurilmalarni ko’rib chiqamiz.

Birinchi klassik dastur. Birinchi klassik dasturlarni variantlarini keltiramiz. U ekranga Hello, World! xabarini chiqaradi.

```
// Bu dastur ekranga "Hello, World! xabarini chiqaradi"
#include "std_lib_facilities.h"
intmain() // C++ da dasturlar main funksiyasidan boshlanadi
{
cout<< "Hello, World!\n"; // "Hello, World!" ni chiqarish
return 0;
}
```

Kompyuter bajarishi lozim bo’lgan bu buyruqlar to’plami, pazandalik resepti yoki yangi o’yinchoqni yig’ish bo’yicha qo’llanmani eslatadi. Eng boshidan boshlab, dasturning har bir satrini ko’rib chiqamiz:

```
cout<<"Hello, World!\n";// "Hello, World!" chiqarish
```

Aynan mana shu satr xabarni ekranga chiqaradi. U yangi satrga o’tuvchi belgi bilan Hello, World! belgilarini chop etadi; aks holda, Hello, World! belgisini chiqargandan so’ng kursor yangi satrning boshiga joylashtiriladi. Kursor - bu keyingi belgi qayerda chiqishini ko’rsatib turuvchi katta bo’lmagan yonib o’chib turuvchi belgi yoki satr.

C++ tilida satrli literallar qo’shtirnoq (") bilan belgilanadi; ya’ni "Hello, Word!\n" — bu belgilar satri. \n belgi - yangi satrga o’tishni bildiruvchi maxsus belgi. cout standart chiqarish oqimiga tegishli. "cout oqimida chiquvchilar" << chiqarish operatori yordamida ekranda aks etadi. cout “see-out” kabi talaffuz qilinadi, lekin “character putstream” ("belgilarni chiqarish oqimi") qisqartmasi hisoblanadi. Dasturlashda qisqartmalar yetarlicha keng tarqalgan. Albatta, qisqartmalar birinchi marta eslab qolish uchun noqulay ko’rinishi mumkin, lekin o’rganib qolgach ulardan voz kecha olmaysiz, ya’ni ular qisqa va boshqarib bo’ladigan dasturlarni yaratishga imkon beradi.

Satr oxiri

```
// "Hello, World!" chiqarish
```

izoh hisoblanadi. // (ya’ni, ikkita egri chiziqdan keyin (/)) belgidan keyin yozilgan barchasi izoh hisoblanadi. U kompilyator tomonidan inkor qilinadi va dasturni o’quvchi dasturchilar uchun mo’ljallangan. Bu holatda biz satrning birinchi qismi nimani anglatishini sizga xabar qilish uchun izohdan foydalandik.

Izohlar insonlar uchun dastur matnida ifodalashning iloji bo’lmagan foydali ma’lumotlarni qamrab oladi va dastur manzilini tasvirlaydi. Umuman olganda, izoh o’zingiz uchun ham foydali bo’lishi mumkin, ya’ni yaratgan dasturingizga xafta yoki yillar o’tib murojat qilganingizda nima uchun yaratilganini tezda anglab olishga yordam beradi. O’zingizni dasturlaringizni yaxshi xujjatlashtirishga harakat qiling.

Dastur ikkita auditoriya uchun yoziladi. Albatta, biz ularni bajaruvchi kompyuterlar uchun dastur yozamiz. Biroq biz kodni o’qish va modifikasiyalashga uzoq yillar sarflaymiz. Shunday qilib, dastur uchun ikkinchi auditoriya bo’lib boshqa dasturchilar hisoblanadi. Shuning uchun dasturning yaratilishini insonlar

o’rtasidagi muloqot shakli deb sanash mumkin. Albatta, insonlarni o’z dasturlarini birinchi o’quvchilari deb hisoblash maqsadga muvofiq: agar ular qiyinchilik bilan o’z yozganlarini tushunishsa, u holda dastur qachonlardir to’g’ri bo’lishi dargumon. Shu sababli, kod o’qish uchun mo’ljallanganligini esdan chiqarmaslik kerak - dastur onson o’qilishi uchun barcha imkoniyatlarni qilish kerak. Istalgan holatda izohlar faqat insonlar uchun kerak, kompyuter ularni inkor qiladi.

Dasturning birinchi satri - bu o’quvchilarga dastur nima qilishi kerakligi haqida xabar beradigan toifaviy izoh.

```
// Bu dastur ekranga "Hello, World!" xabarini chiqaradi.
```

Bu izohlarning foydali tomoni shundaki, bunda dastur kodi bo’yicha dastur nima qilayotganini tushunish mumkin, lekin biz aynan nima hohlayotganimizni aniqlash mumkin emas. Bundan tashqari, kodning o’zidan tashqari, biz izohlarda dasturning maqsadini qisqacha tushuntirishimiz mumkin. Ko’pincha bunday izohlar dasturning birinchi satrida joylashgan bo’ladi. Boshqa narsalar orasida, ular biz nima qilmoqchi ekanligimizni eslatib o’tadi.

Satr

```
#include "std_lib_facilities.h"
```

o’zi bilan #include direktivasini akslantiradi. U std_lib_facilities.h faylida tasvirlanganlarni imkoniyatlarini "faollashtirish" uchun kompyuterni majburlaydi. Bu fayl C++ (C++ tilining standart kutubxonasi) tilining barcha amalga oshirishlarida ko’zda tutilgan imkoniyatlaridan foydalanishni osonlashtiradi.

std_lib_facilities.h faylning imkoniyatlari berilgan dastur uchun shunda hisoblanadiki, uning yordami bilan biz C++ tilining standart kiritish-chiqarish vositalaridan foydalanish imkoniyatiga ega bo’lamiz. Bu yerda biz faqatgina cout standart chiqarish potoki va << chiqarish operatoridan foydalanamiz. #include direktivasi yordamida dasturga kiruvchi fayl odatda .h kengaytmasiga ega bo’ladi

va sarlavha (header) yoki sarlavha fayli (header file) deb nomlanadi. Sarlavha biz dasturimizda foydalanadigan cout kabi atamalarni aniqlashni o’z ichiga oladi.

Dastur bajarilishi boshlanadigan nuqtani dastur qanday aniqlaydi? U main nomli funksiyani ko’rib chiqadi va uni qo’llanmalarini bajarishni boshlaydi. Bizning “Hello, World!” dasturimizda main funksiyasi quyidagicha ko’rinadi:

```
int main() // C++ da dasturlash main funksiyasi yordamida amalga oshiriladi
{
cout << "Hello, World!\n"; // "Hello, World!" chiqarish
return 0;
}
```

Bajarishning boshlang’ich nuqtasini aniqlash uchun, C++ tilidagi har bir dastur main nomli funksiyadan tashkil topgan bo’lishi zarur. Bu funksiya to’rtta qismdan iborat.

1. Qiymat qaytaruvchi toifa, bu funksiyada - int toifa (ya’ni, butun son), funksiya chaqirish nuqtasida qanday natija qaytarishini aniqlaydi (agar u qandaydir qiymat qaytarsa). int so’zi C++ tilida zahiralangan hisoblanadi (kalit so’z), shuning uchun uni boshqa bir narsaning nomi sifatida foydalanish mumkin emas.

2. Nom, main berilgan holatda.

3. Parametrlar ro’yxati, aylana qavsda berilgan; berilgan holatda parametrlar ro’yxati bo’sh.

4. Funksiya tanasi, figurali qavsda berilgan va funksiya bajarishi kerak bo’lgan faoliyatlar ro’yxati.

Bundan kelib chiqadiki, C++ tilidagi kichik dastur quyidagicha ko’rinadi:

```
int main() { }
```

Bu dastur hech qanday amal bajarmaganligi uchun undan foyda kam. “Hello, World!” dasturining main funksiyasining tanasi ikkita qo’llanmadan iborat:

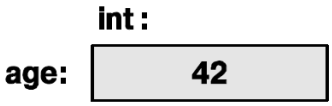
```
    cout << "Hello, World!\n"; // "Hello, World!" chiqish

    return 0;
```

Birinchidan, u ekranga Hello, World! satrini chiqaradi, so’ngra chaqirish nuqtasiga 0 (nol) qiymat qaytaradi. Qachonki main() funksiyasi tizim sifatida chaqirilsa, biz qiymat qaytarmasdan foydalanamiz. Lekin ayrim tizimlarda (Unix/Linux) bu qiymatdan dasturni muvaffaqiyatli bajarilishini tekshirish uchun foydalanish mumkin. main() funksiyasidagi qaytariluvchi Nol (0), dastur muvaffaqiyatli bajarilganini bildiradi.

2.2 Obyektlar, qiymat va toifalar

“Hello world” dasturi faqatgina ekranga yozuv chiqaradi holos. U hechnima o’qimaydi, ya’ni, foydalanuvchi hechnima kiritmaydi. Bu juda zerikarli. Haqiqiy dasturlar odatda biz unga kiritgan ma’lumotlarga qarab qandaydir hisob kitoblar o’tkazadi.

Ma’lumotlarni o’qib olish uchun bu ma’lumotlarni saqlashga joy bo’lishi kerak, boshqacha qilib aytganda, o’qib olingan ma’lumotni yozish uchun bizga kompyuter hotirasidan joy kerak. Bu joyni biz Obyekt deb ataymiz. Obyekt – bu ma’lum bir tur (bu joyda saqlash mumkin bo’lgan ma’lumot turi) ga ega bo’lgan hotiradagi joy. Nom berilgan obyekt esa o’zgaruvchi deyiladi. Masalan, satrlar *string* turiga ega bo’lgan o’zgaruvchilarga saqlanadi, butun sonlar esa – *int* turiga ega bo’lgan o’zgaruvchilarga saqlanadi. Obyektni uning ichiga ma’lum bir turdagi ma’lumotni saqlash mumkin bo’lgan “quti” deb tasavvur qilishimiz mumkin.



Masalan, rasmda *int* turiga mansub, nomga ega bo'lgan va 42 butun soni o'zida saqlagan obyekt tasvirlangan. Satrni turli o'zgaruchilarni kiritish qurilmasidan o'qib olib quyidagicha ekranga chop etishimiz mumkin:

```
// ismni o'qish va yozish

#include "std_lib_facilities.h"

int main()

{

cout << "Iltimos, ismingizni kiriting (keyin 'enter' tugmasini bosing):\n";

string first_name; // first_name — bu string turli o'zgaruvchi

cin >> first_name; // first_name o'zgaruvchiga belgilarni o'qib olamiz

cout << "Hello, " << first_name << "!\n";

}
```

#include direktivasi bizning barcha dasturlarimizda ishlatilgani uchun, chalkashishlardan qochish maqsadida biz buni o'rganishni ortga suramiz. Shu kabi ba'zida biz *main()* yoki boshqa bir funksiya ichiga yozilganda ishlaydigan kodlarni keltirib o'tamiz.

```
cout << "Iltimos, ismingizni kiriting (keyin 'enter' tugmasini bosing):\n";
```

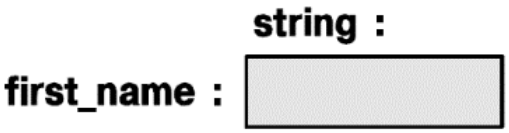
Biz sizni bu kodni testlash uchun to'liq dasturga qanday qo'shishni bilasiz deb hisoblaymiz.

main() funksiyasining birinchi qatori foydalanuvchiga ismini kiritishni taklif qiluvchi habar chiqaradi. Keyingi qatorlarda string turli *first_name* o'zgaruvchi e'lon qilindi, ekrandan shu o'zgaruvchiga ma'lumot o'qib olindi va ekranga *Hello*

so’zidan so’ng *first_name* o’zgaruvchining qiymati chiqarildi. Shu qatorlarni birin ketin ko’rib chiqamiz.

```
string first_name; // first_name — bu string turga ega bo'lgan o'zgaruvchi
```

Bu qatorda kompyuter hotirasidan belgilarni saqlash uchun joy ajratilmoqda va unga *first_name* nomi berilmoqda.

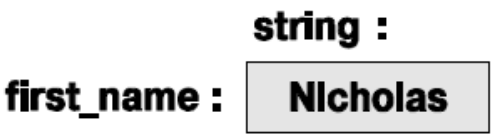


Xotiraan joy ajratish va unga nom berish jarayoni *e’lon qilish* deyiladi.

Keyingi qatorda kiritish qurilmasidan (klaviaturadan) o’zgaruvchiga ma’lumot o’qib olinmoqda:

```
cin >> first_name; // first_name o’zgaruvchisiga belgilarni o’qib olamiz
```

cin (“si-in” singari o’qiladi, ingilizcha character input so’zlari qisqartmasidir) nomi standart kutibhonada e’lon qilingan standart oqim kirituviga tegishli. Keyinigi >> (kirituv) operatorining operandi kirituv natijasi saqlanadigan joyni ko’rsatadi. Demak, agar biz Nicholas ismini kiritib yangi qatorga o’tganimizda (ya’ni enter tugmasini bosganimizda) *“Nicholas”* satri *first_name* o’zgaruvchisining qiymatiga aylanadi.



Yangi qatorga o’tish kompyuterning e’tiborini jalb yetish uchun muhim. Yangi qatorga o’tilmagunga qadar (enter tugmasi bosilmagunga qadar) kompyuter shunchaki belgilarni to’plab boradi. Bu kutish bizga ba’zi belgilarni o’chirish yoki boshqa belgilar bilan almashtirish imkonini beradi.

first_name o’zgaruvchisiga qiymatni kiritganimizdan so’ng uni kelgusida ishlatishimiz mumkin bo’ladi.

```
cout << "Hello, " << first_name << "!\n";
```

Bu qator Hello, so’zi va undan so’ng Nicholas ismini (*first_name* o’zgaruvchisining qiymati), ohirida undov belgisi (!) va yangi qatorga o’tish belgisi (‘*n*’)ni chiqaradi

```
Hello, Nicholas!
```

Agar biz takrorlanish va ko’p matn yozishni yoqtirganimizda yuqoridagi kodni quyidagicha yozishimiz mumkin:

```
cout << "Hello, ";
cout << first_name;
cout << "!\n";
```

E’tibor bering, Hello, so’zini bir qo’shtirnoqda chiqardik *first_name* o’zgaruvchisini bo’lsa qo’shtirnoqlarsiz yozdik. Qo’shtirnoqlar literal satrlar bilan ishlash uchun zarur, agar satr qo’shtirnoqsiz yozilgan bo’lsa demak u biror bir nomga (o’zgaruvchiga) murojaat qilgan bo’lamiz.

```
cout << "Ism" << " – " << first_name;
```

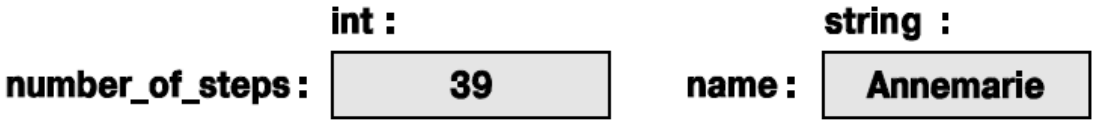
Bu yerda “Ism” uchta belgidan iborat satrni tashkil qiladi, *first_name* bo’lsa ekranga *first_name* o’zgaruvchisining qiymatini (bizning holatda Nicholas) chiqaradi. Demak natija quyidagicha ko’rinishga ega:

```
Ism – Nicolas
```

O’zgaruvchilar. Yuqorida keltirilgan misolda ko’rsatilganidek kompyuter hotirasida ma’lumot saqlash imkinoyitisiz kompyuterda hech qanday qiziqarli ish qilib bo’lmaydi. Ma’lumotlar saqlanadigan joyini obyekt deb ataymiz. Obyektga murojaat qilish uchun obyekt nomini bilish kerak. Nomlangan obyekt o’zgaruvchi deyiladi va unda qanday ma’lumotni saqlash mumkinligini aniqlovchi (masalan *int* turidagi obyektda butun sonlarni saqlash mumkin *string* turidagi obyektga ‘*Hello world*’ singari satrlarni saqlash mumkin) aniq bir turga ega bo’ladi. (masalan, *int* yoki *string*), bundan tashqari ular ustida amallar bajarish mumkin (masalan *int* turidagi obyekt ustida * operatori orqali ko’paytirish amalini bajarish mumkin, *string* turidagi obyektlarni esa <= operatori orqali taqqoslash mumkin). O’zgaruvchilarga yozilgan ma’lumotlar qiymat deyiladi. O’zgaruvchini aniqlash instruktsiyasi e’lon qilish deyiladi va o’zgaruvchi e’lon qilinayotgan paytda unga boshlang’ich qiymat berib ketsa bo’ladi. Quyidagi misolni ko’rib chiqamiz:

```
string name = "Annemarie";
int number_of_steps = 39;
```

Bu o’zgaruvchilarni quyidagicha tasavvur qilish mumkin:



Biz o’zgaruvchiga to’g’ri kelmaydigan turdagi ma’lumotni yoza olmaymiz.

```
string name2 = 39; // xato: 39 — satr emas
int number_of_steps = "Annemarie"; // xato: "Annemarie" —
                                   // butun son emas
```

Kompilyator har bir o’zgaruvchining turini saqlab qoladi va o’zgaruvchini uning turiga mos ravishda ishlatishingizga yo’l qo’yadi.

C++ tilida juda ko’plab turlar tanlovi mavjud. Lekin ularning atiga 5 tasidan foydalangan holda foydali dasturlar tuzish mumkin.

```
int number_of_steps = 39; // int — butun sonlar uchun
double flying_time = 3.5; // double — haqiqiy sonlar uchun
char decimal_point = '.'; // char — belgilar uchun
string name = "Annemarie"; // string — satrlar uchun
bool tap_on = true;  // bool — mantiqiy o'zgaruvchilar uchun
```

E’tibor bering, barcha o’zgaruvchilar o’zining yozilish uslubiga ega:

```
39  //  int: butun son
3.5 //  double: xaqiqiy son
'.' //  char: bittalik tirnoqcha ichida yagona belgi
"Annemarie"  //  string: qo’shtirnoq ichida yozilgan belgilar to’plami
true//  bool: yoki rost (true), yoki yolg’on (false) qiymatga ega
```

Boshqacha qilib aytganda, raqamlar ketma ketligi (masalan 1234, 2 yoki 973) butun sonni bildiradi, bittalik tirnoqcha ichidagi belgi (masalan, ‘1’, ‘@’ yoki ‘x’) belgini bildiradi, xaqiqiy sonlar esa (masalan, 123.123, 0.12, .98) haqiqiy sonlarni bildiradi, qo’shtirnoq ichidagi belgilar ketma ketligi esa (masalan, “123141”, “Howdy!” yoki “Annemarie”), satrni bildiradi. Batafsil ma’lumot ilova qilingan adabiyotlarda berilgan.

Kiritish va tur. Kiritish operatori ma’lumotlar turiga juda ta’sirchan, ya'ni u kirituv amalga oshayotgan o'zgaruvhi turiga mos ravishda ma'lumotlarni o'qiydi. Quyidagi misolga e’tibor bering:

```
// ism va yoshni kiritish
int main()
{
cout << "Iltimos ismingiz va yoshingizni kiriting\n";
string first_name; // string turdagi o’zgaruvchi
int age; // integer turidagi o’zgaruvchi
cin >> first_name; // string turdagi ma’lumotni o’qib olamiz
cin >> age;  // integer turidagi ma’lumotni o’qib olamiz
cout << "Hello, " << first_name << " (age " << age << ")\n";
}
```

Demak siz klaviaturada Carlos 22 ni tersangiz kiritish operatori >> first_name o’zgaruvchisiga Carlos ni 22 sonini esa age o’zgaruvchisiga o’qib oladi va quyidagi natijani ekranga chop etadi:

```
Hello, Carlos (age 22)
```

Nega Carlos 22 satri butunlayicha *first_name* o’zgaruvchisiga yozilmaganining sababi satrlarni o’qish ajratish belgisi (whitespace) ya’ni probel yoki tabulyatsiya belgisi uchrashi bilan yakunlanadi. Bunday holatda ajratish belgisi kiritish operatori >> tomonidan tashlab ketiladi va sonni o’qishga o’tiladi.

Agar siz klaviaturada 22 Carlos ni terib ko’rsangiz kutilmagan natijaga guvoh bo’lasiz. 22 soni first_name o’zgaruvchisiga yoziladi, chunki 22 ham belgilar ketma ketligi hisoblanadi. Boshqa tomondan esa Carlos butun son emas va u o’qilmasdan tashlab ketiladi. Natijada esa ekranga 22 soni va daviomda “(age” literali va ihtiyoriy son masalan -9842 yoki 0 chop etiladi. Nega? Chunki siz age o’zgaruchisining boshlang’ich qiymatini kiritmadingiz va hech nima kiritmadingiz, natijada unda musor qiymat qolib ketdi. Hozir esa shunchaki age o’zgaruchisiga boshlang’ich qiymat berib qo’yamiz.

```
// ism va yoshni kiritish (2- usul)
int main()
{
cout << "Iltimos ismingiz va yoshingizni kiriting\n";
string first_name = "???";// string turidagi o’zgaruvchi
// ("???" ism kiritilmaganligini bildiradi")
int age = -1;    // int turidagi o'zgaruvchi (-1 “yosh aniqlanmaganligini bildiradi”)
cin >> first_name >> age; // satr undan so’ng butun sonni o’qiymiz
cout << "Hello, " << first_name << " (age " << age << ")\n";
}
```

Endi 22 Carlos satrini kiritish quyidagi natijaga olib keladi:

```
Hello, 22 (age -1)
```

E’tibor bering, biz kiritish operatori orqali bir nechta qiymatlarni kiritishimiz mumkin, bitta chiqarish operatori bilan ularni chop etishimiz mumkin. Bundan tashqari chiqarish operatori << ham kiritish operatori >> singari turlarga sezuvchandir, shuning uchun string turidagi o’zgaruvchi va bir qator satrlar bilan birgalikda butun son (int) turdagi o’zgaruvchini chop etishimiz mumkin.

string turidagi obyektni kiritish operatori >> orqali kiritish ajratish belgisi uchraganda to’xtatiladi boshqacha qilib aytganda kiritish operatori alohida so’zlarni o’qiydi. Ba’zida bizga bir nechta so’zlarni o’qish kerak bo’ladi. Buning ko’plab usuli bor, masalan ikkita so’zdan iborat ismni o’qib olish mumkin:

```
int main()
{
cout << "Iltimos ism, familiyangizni kiriting\n";
string first;
string second;
cin >> first >> second; // ikkita satr o’qib olamiz
cout << "Hello, " << first << ' ' << second << '\n';
}
```

Bu yerda biz kiritish opratorini >> ikki marta ishlatdik. Agar bu so’zlarni ekranga chiqarish kerak bo’lsa ular orasidan probel qo’yish zarur.

Amallar va operatorlar. O’zgaruvchilarning turlari ularda qanday ma’lumot saqlanishidan tashqari ular bilan qanday amallar bajarish mumkinligini ham ko’rsatadi:

```
int count;
cin >> count; // kiritish operatori >> butun sonni count obyektiga yozadi
```

```
    string name;
    cin >> name; // kiritish operatori kiritilgan satrni name o’zgaruvchisiga yozadi
    int c2 = count+2; // + operatori ikki sonni qo’shadi
    string s2 = name + " Jr. "; // + operator belgilar bilan to’ldiradi
    int c3 = count-2;     // - ikki sonni ayiradi
    string s3 = name - "Jr. ";    // xato: - operatori satrlar uchun aniqlanmagan.
```

Xato o’rnida biz kompiltatorning dasturni kompilyatsiya qilmasligini nazarda tutyapmiz. Kompilyator har bir o’zgaruvchiga qanday amallarni bajarish mumkinligini biladi va xato qilsihga yo’l qo’ymaydi. Lekin kompilyator qaysi o’zgaruvchilarga qanday amallarni bajarish mumkinligi haqida bilmaydi va quyidagidek be’mani xatolarga yo’l qo’yib beradi:

```
int age = -100;
```

Ko’rinib turibdiki, inson manfiy yoshga ega bo’la oilmaydi, lekin hech kim bu haqida kompilyatorga aytmadi, shuning uchun bunday hollarda kompilyator hech qanday xatolik haqida habar bermaydi. Quyida eng ko’p tarqalgan turlar uchun amallar ro’yxati keltirilgan:

	Bool	**char**	**int**	**double**	**string**
Tenglash (qiymat	=	=	=	=	=
Qo’shish			+	+	
Ulash					+
Ayirish			–	–	
Ko’paytirish			*	*	
Bo’lish			/	/	
Qoldiq			%		
Birga inkrement			++	++	

1ga			– –	– –	
n ga inkrement			**+=n**	**+=n**	
Ohiriga qo’shish					+=
n ga qo’shish			**-=n**	**-=n**	
Ko’paytirib tenglash			***=**	***=**	
Bo’lib tenglash			/=	/=	
Qoldiq olish va			**%=**		
s fayldan x ga o’qish	**s>>x**	**s>>x**	**s>>x**	**s>>x**	**s>>x**
x ni s faylga yozish	**s<<x**	**s<<x**	**s<<x**	**s<<x**	**s<<x**
Teng	==	==	==	==	==
Teng emas	**!=**	**!=**	**!=**	**!=**	**!=**
Katta	>	>	>	>	>
Katta yoki teng	>=	>=	>=	>=	>=
Kichig	<	<	<	<	<
Kichig yoki teng	<=	<=	<=	<=	<=

Bo’sh kataklar amal bu turlarga to’g’ridan to’g’ri qo’llanib bo’lmasligini ko’rsatadi.Vaqti bilan barcha opreatorlarni tushuntirib o’tamiz.

Xaqiqiy sonlarga doir quidagi na’munani ko’rib chiqamiz:

```
// operatorlar ishini ko’rsatuvchi sodda dastur
int main()
{
cout << "Iltimos haqiqiy son kiriting: ";
double n;
cin >> n;
cout << "n == " << n
<< "\nn+1 == " << n+1
<< "\ntri raza po n == " << 3*n
<< "\ndva raza po n == " << n+n
<< "\nn kvadrati == " << n*n
```

```
    << "\nyarim n == " << n/2
    << "\nn ning kvadrat ildizi == " << sqrt(n)
    << endl; // yangi qatorga o’tishning sinonimi ("end
of line")
    }
```

Ko’rinib turibdiki, oddiy arifmetik amallar odatiy ko’rinishga ega va ularning vazifasi bizga maktab dasturidan ma’lum. Albatta haqiqiy sonlar ustidagi barcha amallar ham operatorlar ko’rinishida tasvirlanmagan, masalan, kavadrat ildiz funksiya ko’rinishida tasvirlangan. Ko’plab amallar funksiyalar sifatida tasvirlangan. Bu holda esa n ning kvadrat ildizini topish uchun standart kutubhonalardagi sqrt(n) funksiyasi ishlatilmoqda.

string turi uchun kam operatorlar ko’zda tutilgan lekin bu tur ustida ko’plab amallar funksiyalar sifatida ifodalangan. Bundan tashqari ularga operatorlarni ham qo’llash mumkin:

```
// ism familyani kiritish
int main()
{
cout << "Iltimos, ism va familiyangizni
kiriting\n";
string first;
string second;
cin >> first >> second; // ikki satrni o’qib olamiz
string name = first + ' ' + second; // satrlarni
birlashtiramiz
cout << "Hello, " << name << '\n';
}
```

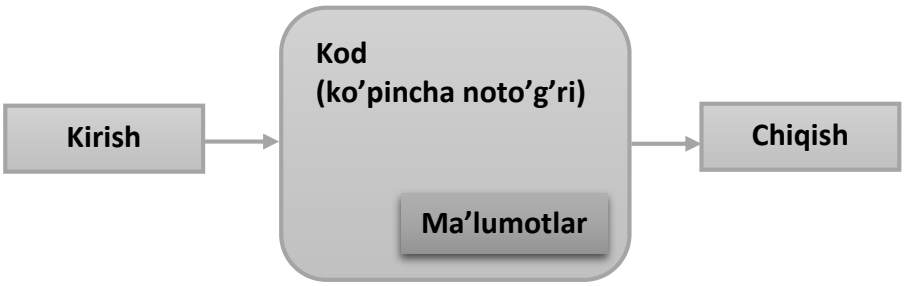
Satrlar uchun + operatori birlashtirishni bildiradi; boshqacha qilib aytganda, agar s1 va s2 o’zgaruvchilari string turiga ega bo’lsa, unda s1 + s2 ham s1 satrdan keyin s2 satr belgilari kelgan satr hisoblanadi. Masalan s1 “Hello” qiymatiga, s2 bo’lsa “World” qiymatiga ega bo’lsa, unda s1 + s2 “HelloWorld” qiymatiga ega bo’ladi. Satrlar ustidagi amallarning asosiylaridan biri bu taqqoslashdir.

```
// ismni kiritish va taqqoslash
int main()
{
cout << "Iltimos ikkita ism kiriting\n";
string first;
string second;
cin >> first >> second; // ikkita satr o’qib olamiz
if (first == second) cout << "ismlar bir hil\n";
if (first < second)
cout << first << " alifbo bo’yicha birinchi keladi
" << second <<'\n';
if (first > second)
cout << first << " alifbo bo’yicha keyin keladi "
<< second <<'\n';
}
```

Bu yerda harakatlarni tanlash uchun if operatori ishlatilgan. Bu operator haqida ilovada keltirilgan adabiyotlardan o’qishingiz mumkin.

2.3. Hisoblash

Barcha dasturlar nimanidir hisoblaydi; boshqacha aytganda ular kirishda qandaydir ma’lumotlarni oladi va qandaydir natijalarni chiqaradi. Bundan tashqari, dasturlar bajariladigan qurilmaning o’zi kompyuter deb nomlanadi (ingliz tilidan tarjima qilganda computer - hisoblagich). Bu nuqtai nazar biz kiritish va chiqarish muomalasiga keng rioya qilishimizda to’g’ri va oqilona hisoblanadi.

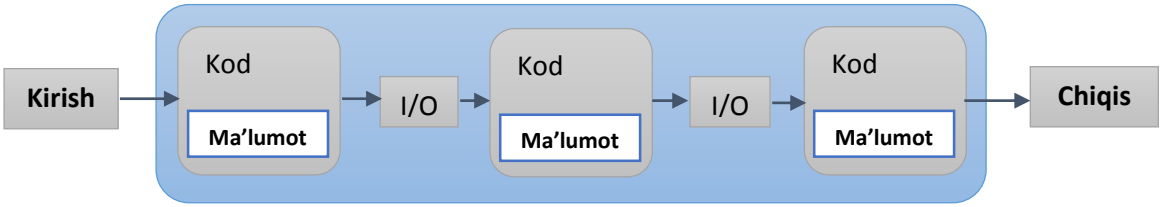


Kiruvchi ma’lumotlar klaviaturadan, sichqonchadan, sensor ekrandan, fayllardan, kiritishning boshqa qurilmalaridan va dasturning boshqa qismlaridan kiritilishi mumkin. "Kiritishning boshqa qurilmalari" kategoriyasi qiziqarli ma’lumotlar manbaini o’z ichiga oladi: musiqiy klavishli pultlar, videoyozuv qurilmalari, harorat hisoblagichlari, raqamli videokameralar sensori va shu kabilar. Bu qurilmalarning turlari chegarasi yo’q.

Dasturga kiruvchi ma’lumotlarni qayta ishlash uchun odatda ma’lumotlar tuzilmasi (data structures) yoki ularning tarkibi (states) deb nomlanuvchi maxsus ma’lumotalrdan foydalaniladi. Masalan, kalendarni aks ettiruvchi dasturda turli mamlakatlardagi bayram kunlari ro’yxati va ish yuzasidan uchrashuvlaringiz keltirilgan bo’lishi mumkin. Ma’umotlarning ayrimlari avval boshidan dastur qismi hisoblanadi, boshqalari esa, qachonki dastur ma’lumotlarni o’qiganda va ularda foydali ma’lumotalrni uchratganda sodir bo’ladi. Masalan, kalendarni aks ettiruvchi dastur, ish bo’yicha uchrashuvlaringizni kiritishni boshlaganingizda ro’yhatni yaratishi mumkin. Bu holatda, asosiy kiruvchi ma’lumot uchrashuv oy va kuniga so’rov yuborishda va ish bo’yicha uchrashuvlar ma’lumotlarini kiritishda hisoblanadi.

Kiruvchi ma’lumot turli xil manbalardan kirishi mumkin. Shunga o’xshash, natijalar ham turli xil qurilmalarda chop etilishi mumkin: boshqa dasturlar yoki dastur qismlari ekranida. Chiquvchi qurilmaga tarmoq interfeyslari, musiqiy sintezatorlar, elektr motorlar, quvvat generatorlari, isitgichlar va boshqalar kiradi.

Dasturchi nuqtai nazaridan kiritish-chiqarishda eng muxim va qiziqarli kategoriyalar “boshqa dasturga” va “dasturning boshqa qismlariga” hisoblanadi. Qanday qilib dasturni o’zaro ta’sir etuvchi qismlar ko’rinishida tasavvur qilish va ma’lumotlarga o’zaro kirish va ma’lumotlar almashishni ta’minlash mumkinligi. Bu dasturlashning kalit savollari. Ularni grafik ko’rinishda tasvirlaymiz.



I/O qisqartmasi kiritish-chiqarishni bildiradi. Bunday holatda dasturning bir qismidan chiqish keyingi qismga kirish hisoblanadi. Dasturning bu qismlari doimiy xotira qurilmasida ma’lumotlarni saqlash yoki tarmoq aloqalari orqali uzatiladigan asosiy xotirada saqlanadigan malumotlarga kirishga ega bo’ladi.

Dastur qismida kiruvchi ma’lumotlar, odatda argument, dastur qismidan chiquvchi ma’lumotlar esa natija deb ataladi.

Hisoblash deb, aniq kiruvchi ma’lumotlar asosida va aniq natijalarni hosil qiluvchi qandaydir ta’sirlarga aytiladi. Eslatib o’tamizki, 1950 yilgacha AQSh da kompyuter deb, hisoblashlarni amalga oshiruvchi insonlar atalgan, masalan, hisobchilar, navigator, fizik. Hozirgi kunda oddiygina ko’plab hisoblashlarni kompyuyeterga topshirdik, bularning orasida kalkulyator eng soddasi hisoblanadi.

Ifoda. Dasturning asosiy konstruktor qurilmasi ifoda hisoblanadi. Ifoda aniqlangan operandlar soni asosida qandaydir qiymatlarni hisoblaydi. Sodda ifodalar o’zida oddiy literal konstantalarni ifodalaydi, masalan, 'a', 3.14 yoki "Norah".

O’zgaruvchilar nomi ham ifoda hisoblanadi. O’zgaruvchi – bu nomga ega bo’lgan obyekt. Misol ko’ramiz.

```
// maydonni hisoblash:
int length = 20; // literal butun qiymat
// (o’zgaruvchini inisializasiyalash uchun foydalaniladi)
int width = 40;
int area = length*width; // ko’paytirish
```

Bu yerda length so’zi, o’zlashtiruvchi operator chap operandalarini belgilovchi “length nomli obyektni” anglatadi, shuning uchun bu ifoda quyidagicha o’qiladi: “length nomli obyektga 99 raqamini yozish”. O’zlashtiruvchi yoki inisializasiyalovchi operatorning chap qismida (u “length o’zgaruvchining lvalue” deb nomlanadi) va bu operatorlarning o’ng qismida (bu holatda u “length o’zgaruvchining rvalue”, “length nomli obyektning qiymati” yoki oddiy “length qiymati”) joylashgan length o’zgaruvchini ajratib olish kerak. Bu kontekstda nomlangan quti ko’rinishida o’zgaruvchilarni tasvirlash foydali.

Boshqacha aytganda, length – bu 99 tagacha qiymatni o’z ichiga oluvchi int toifali obyekt nomi. Ba’zida length nomi (lvalue sifatida) qutiga, ba’zida esa (rvalue sifatida) – mana shu qutida saqlanayotgan qiymatning o’ziga tegishli bo’ladi.

+ va * operatorlari yordamida ifodalarni birlashtirib, quyida ko’rsatilganidek murakkabroq ifodalarni yaratishimiz mumkin. Ifodalarni guruhlash kerak bo’lganda qavslardan foydalanish mumkin.

```
int perimeter = (length+width)*2; // qo'shish va ko'paytirish
```

Qavssiz bu ifodani quyidagi ko'rinishda yozish mumkin:

```
int perimeter = length*2+width*2;
```

juda qo'pol va xatoliklarni keltirib chiqaradi.

```
int perimeter = length+width*2; // length bilan width*2 qo'shish
```

Oxirgi xatolik mantiqiy hisoblanadi va kompilyator uni topa olmaydi. Kompilyator aniq ifodada inisializasiyalangan perimeter nomili o'zgaruvchini ko'radi. Agar ifoda natijasi ma'noga ega bo'lmasa, bu sizning muammoyingiz. Siz perimetrning matematik aniqliklarini bilasiz, kompilyator esa yo'q.

Dasturda operatorlar bajarilish tartibini aniq belgilaydigan, oddiy matematik qoidalardan foydalaniladi, shuning uchun `length+width*2`, `length+(width*2)` ni anglatadi. Shunga o'xshash `a*b+c/d`, `a*(b+c)/d` emas `(a*b)+(c/d)` ni anglatadi.

Qavsdan foydalanishning birinchi qoidasi shundaki: "Agar ikkilanayotgan bo'lsang qavsdan foydalan". Umuman olganda dasturchi `a*b+c/d` formula qiymatida ikkilanmaslik uchun ifodalarni to'g'ri shakllantirishni o'rganishi kerak. Operatorlardan keng foydalanish, masalan `(a*b)+(c/d)` dastur o'qilishini susaytiradi.

Nima uchun biz o'qilishiga to'xtalib o'tdik? Chunki sizning kodingizni faqatgina siz emas, balki boshqa dasturchilar ham o'qishi mumkin, chalkashtirilgan kod o'qishni sekinlashtiradi va uning tahliliga to'sqinlik qiladi. Qo'pol kodni faqatgina o'qish emas, balki uni to'g'irlash ham qiyin bo'ladi. Yomon yozilgan kod odatda mantiqiy xatoliklarni berkitib turadi. Uning o'qilishida qancha ko'p kuch ketgan bo'lsa, o'zingizni va boshqalarni uning to'g'riligiga ishontirish shuncha qiyin

bo’ladi. Juda ham qiyin bo’lgan ifodalarni yozmang va har doim ma’noli nomlarni tanlashga harakat qiling.

```
a*b+c/d*(e-f/g)/h+7 // juda qiyin
```

2.4 Xatoliklar

Dastur ishlab chiqishda xatolardan qochib bo’lmaydi, lekin dasturning ohirgi nusxasi iloji boricha xatolarsiz bo’lishi zarur.

Xatolarning ko’plab turi bor:

- *Kompilyatsiya paytidagi xato*. Kompilyator tomonidan aniqlangan xatolar.

Bularni dasturlash tilining qaysi qoidalarnini buzishiga qarab bir nechta turlarga bo’lishimiz mumkin.

  - Sintaktik xatolar;
  - Tiplar bilan yo’lga qo’yilgan xatolar.

- *Bog’lanishlar paytidagi xato.* Bu bog'lanishlar tahrirlovchisi tomonidan obyekt fayllarni bajarilish moduliga qo'shish paytida topilgan xatolar.
- *Bajarilish paytidagi xatolar*. Bu dastur bajarilish davomida vujudga kelgan xatolar. Bularni quyidagi turlarga bo’lish mumkin:
  - Kompyuter tomonida aniqlangan xatolar (uskunaviy taminot va/yoki operatsion tizim tomonidan aniqlangan xatolar);
  - Kutubhonalar tomonidan aniqlangan xatolar (masalan standart kutubhonalar tomonidan);
  - Foydalanuvchining dasturi tomonidan aniqlangan xatolar.
- *Mantiqiy xatolar*. Dasturchi tomonidan noto'g'ri vaziyatlar yuzaga kelayotganda topilgan xatolar.

Oddiy qilib aytganda dasturchining vazifasi – barcha xatolarni bartaraf etish. Lekin ko’p xollarda bu vazifa amallab bo’lmas bo’ladi. Aslida haqiqiy dasturlar uchun “barcha xatolar” deganda nimani tushunish juda qiyin. Masalan, dastur ishlab turgan paytda biz kompyuterni elektor manbaidan uzib qo’ysak buni xato sifatida qarab unga qarshi chora ko’rish kerakmi? Ko’p hollarda rad javobini beramiz, lekin

meditisina monitoringi dasturida yoki qo’ng’iroqlarni yo’naltirish dasturlarida bunday emas. Bunday xollarda foydalanuvchi sizning dasturingizdan ma’lum bir hisoblashlarni davom ettirishini talab qiladi. Asosiy savol shundan iborat: Dasturingiz o’zi xatolikni aniqlashi kerakmi yoki yo’qligida.

Agar bu aniq ko’rsatilmagan bo’lsa, biz sizning dasturingiz quyidagi shartlarni bajarishi kerak:

1. Istalgan kiruvchi ma’lumotlarda dasturdan talab qilingan hisoblash ishlarini bajarishi kerak.
2. Barcha to’g’irlab bo’lmas hollarda kerakli xabarni chiqarishi kerak.
3. Uskunaviy ta’minotning barcha xatoliklari xaqida ma’lumot berishi shart emas.
4. Dasturiy ta’minotning barcha xatosini chiqarishi shart ema.
5. Xatolik topilganda dastur ishini yakunlashi kerak.

3-5 ko’rsatilgan shartlarga javob bermaydigan dasturlarni ko’rib chiqmaymiz. Shu bilan birga 1 va 2 professional talablardan biri hisoblanadi, professionallik esa bizning maqsadimiz. 100% mukammalikka erisha olmasakda u qisman mavjud bo’lishi zarur.

Dastur tuzishda xatolar odatiy hol va ulardan qochib bo’lmaydi. Bizning fikrimizcha, jiddiy dasturlar yaratishda, xatolarni chtlab o’tish, topish va to’g’irlash 90% ko’proq vaqtni oladi. Xavfsizligi muhim bo’lgan dasturlarda esa bu vaqt undan xam ko’p bo’ladi. Kichik dasturlarda xatolardan osongina qochish mumkin, lekin katta dasturlarda bu xatoga yo’l qo’yish ehtimoli juda yuqori.

Biz dastur tuzishda quyidagi uch uslubni taklif qilamiz:

- Dasturiy ta’minotni xatoliklarini iloji boricha kamaytirib ishlab chiqish.
- Ko’plab xatolarni dstur ishlab chiqish va testlash jarayonida to’g’irlab ketish.
- Qolgan xatolarni jiddiy emasligiga ishonch hosil qilish.

Bu uslublarning birortasi ham o’z o’zidan xatolarni to’g’ri bo’lishini ta’minlamaydi. Shuning uchun ham biz barcha uch uslubni ham ishlatamiz.

Ishonchli dasturlarni ishlab chiqishda malaka katta ahamiyatga ega.

Xatolarning bir nechta manbalarini keltirib o’tamiz.

Yomon tasvirlash. Agar biz dasturning vazifasini tasavvur qila olmasak, uning “qora burchak” larini tekshirib chiqishimiz juda qiyin bo’ladi.

Chala dasturlar. Dasturni yaratish davomida biz e’tiborga olmay ketgan xatoliklar albatta yuzaga chiqadi. Bizning vazifamiz barcha holatlar to’g’ri ishlab chiqilganligiga ishonch hosil qilish.

Ko’zda tutilmagan argumentlar. Funksiyalar argumentlar qabul qiladi. Agar funksiya ko'zda tutilmagan argument qabul qilsa, muammo hosil bo'ladi, masalan standart kutubhonada mavjud sqrt() funksiyasiga manfiy ishoralik son berilsa. sqrt() funksiyasi faqatgina musbat xaqiqiy son qabul qilgani uchun u to’g’ri natija qaytara olmaydi. Bunday muammolar 5.5.3 bo’limda ko’rib o’tiladi.

Ko’zda tutilmagan kiruvchi ma’lumotlar. Odatda dasturlar malumotlarni o’qib olishadi (masalan, klaviaturadan, fayldan, lokal va global tarmoqlardan). Odatda dasturlar kiruvchi ma’lumotlar uchun ko’plab shart qo’yadi, masalan foydalanuvchi son kiritishi kerakligi. Agar foydalanuvchi kutilayotgan son o’rniga “Bas qil!” degan satrni kiritsachi? Muammoning bunday turi 5.6.3 va 10.6 bo’limlarida ko’rib chiqiladi.

Kutilmagan holat. Ko'plab dasturlar ko'plab ma'lumotlarni (holatlarni) saqlashga mo'ljallangan, masalan tizimnig qismlari. Ular safiga manzillar ro’yxati, telefon katalogi va havo harorati haqidagi ma’lumotlar vector turidagi obyektga yozilgan bo’lsin. Bu ma’lumotlar to’liq bo’lmasa yoki noto’g’ri bo’lsa nima bo’ladi? Bunday holatda dasturning turli qismlari boshqaruvni saqlab qolishi zarur.

Mantiqiy xatolar. Bunday xatolar dasturdan talab etilgan vazifani bajarmaslikka olib keladi. Bunday xatolarni topib bartaraf etishimiz zarur.

Sintaktik xatolar.

area() funksiyasini quyidagicha chaqirsak nima sodir bo'ladi:

```
int s1 = area(7; // xato: qavs tushirib qoldirilgan
)

int s2 = area(7) // xato: nuqta vergul tushirib
qoldirilgan ;

Int s3 = area(7);     // xato: Int — tur emas

int s4 = area('7); // xato: tirnoqcha tushirib
qoldirilgan '
```

Xar bir qator sintaktik xatoga ega, boshqacha qilib aytganda ular C++ grammatikasiga to'g'ri kelmaydi. Afsuski barcha hollarda ham xatolarni dasturchi tushinishiga oson qilib ifodalash qiyin. Natijada eng oddiy sintaktik xatolar xam tushunarsiz ifodalanadi, bundan tashqari xatolikka ko'rsatayotgan qator ham bir oz uzoqroqda joylashgan bo'ladi. Shuning uchun kompilyator ko'rsatayotgan qatorda hech qanday xatolik ko'rmayotgan bo'lsangiz biroz yuqoriroq qatorlarni tekshirib chiqing.

Turlar bilan bog'liq xatoliklar.

Sintaktik xatolarni to'g'irlaganingizdan so'ng kompilyator turlarni e'lon qilishdagi xatoliklar xaqida ma'lumot bera boshlaydi. Masalan e'lon qilinmagan o'zgaruvchi, yoki unga berilayotgan qiymatning to'g'ri kelmasligi to'g'risidagi xatoliklar:

```
int x0 = arena(7);    // xato: e'lon qilinmagan
funksiya

int x1 = area(7);     // xato: argumentlar soni
noto'g'ri

int x2 = area("seven",2); // xato: birinchi
argument noto'g'ri tipga ega
```

Xato emas.

Kompilyator bilan ishlash davomida qaysidir payt u sizning xatolaringizni oldindan bilishini xoxlaysiz va ularni xato sifatida qaramasligini hohlaysiz. Lekin malakangiz oshgani sari siz kompilyatorni iloji boricha ko'roq xatolar topishini hohlab qolasiz. Quyidagi misolni ko'rib chiqamiz:

```
int x4 = area(10,-7); // OK: lekin tomoni -7 ga teng bo'lgan to'g'ri to'rtburchakni qanday tasavvur qilish kerak?

int x5 = area(10.7,9.3); // OK: lekin aslida area(10,9) chaqirilyapti

char x6 = area(100, 9999); // OK: lekin natija kesib tashlanadi
```

Kompilyator x4 o'zgaruvchisi xaqida xech qanday xabar chiqarmaydi. Uning qarashi bo'yicha area(10, -7) to'g'ri hisoblanadi: area() funksiyasi ikkita butun son talab qiladi, siz esa ikki butun sonni unga yuboryapsiz; hech kim bu sonlar musbat bo'lishi xaqida gapirmagan.

Nazorat savollari

1. To'rtta asosiy xatoliklar turini ayting va ularni qisqacha ifodalab bering.
2. Tinglovchilar dasturidagi qaysi xatolarni tashlab ketishimiz mumkin?
3. Xar qanday yakunlangan proyekt nimani kafotlashi kerak?
4. Xatoliklarni topish va bartaraf etishning uchta asosiy uslubini sanab o'ting.
5. Nega biz xato qidirishni yoqtirmaymiz?
6. Sintaktik xato nima? Beshta misol keltiring.
7. Turlar xatosi nima? Beshta misol keltiring.
8. Mantiqiy xato nima? Uchta mmisol keltiring.

Foydalanilgan adabiyotlar

1. Bjarne Stroustrup. Programming: Principles and Practice Using C++ (2nd Edition). Person Education, Inc. 2014. second printing, January 2015.
2. Harry Hariom Choudhary, Bjarne M Stroustrup. C++ Programming Professional.: Sixth Best Selling Edition for Beginner's & Expert's 2014.
3. http://www.stroustrup.com/4th.html
4. http://www.cplusplus.com/

3-Ma’ruza. C++ da o’zgaruvchilar

Reja:

3.1 O’zgaruvchilar

3.2 Konstantalar

3.3 Turlar bilan ishlash

Kalit so’zlar: *bad(), iostream, kiritish amali, buffer, istream, chiqarish amali, clear(), ofstream, kiritish tugallanganligi belgisi, close(), open(), eof() oqimining holati, ostream, kiritish qurilmasi, fail(), unget(), chiqarish qurilmasi, good(), qurilma drayveri, fayl, ifstream.*

3.1. O’zgaruvchilar

O’zgaruvchilar ob’ekt sifatida. C tilining asosiy tushunchalaridan biri nomlangan xotira qismi – ob’ekt tushunchasidir. Obektning xususiy xoli bu o’zgaruvchidir. O’zgaruvchiga qiymat berilganda unga ajratilgan xotira qismiga shu qiymat kodi yoziladi. O’zgaruvchi qiymatiga nomi orqali murojaat qilish mumkin, xotira qismiga esa faqat adresi orqali murojaat qilinadi. O’zgaruvchi nomi bu erkin kiritiladigan identifikatordir. O’zgaruvchi nomi sifatida xizmatchi so’zlarni ishlatish mumkin emas.

O’zgaruvchilarni ta’riflash. C tilida o’zgaruvchini aniqlash uchun kompyuterga uning turi (masalan, int, char yoki float) hamda ismi haqida ma’lumot beriladi. Bu axborot asosida kompilyatorga o’zgaruvchi uchun qancha joy ajratish lozim va bu o’zgaruvchida qanday turdagi qiymat saqlanishi mumkinligi haqida ma’lumot aniq bo’ladi. O’zgaruvchi nomi identifikator bo’lib, xizmatchi so’zlardan farqli bo’lishi kerak.

Har bir yacheyka bir bayt o’lchovga ega. Agar o’zgaruvchi uchun ko’rsatilgan tur 4 baytni talab qilsa, uning uchun to’rtta yacheyka ajratiladi. Aynan o’zgaruvchini

turiga muvofiq ravishda kompilyator bu o’zgaruvchi uchun qancha joy ajratish kerakligini aniqlaydi.

Kompyuterda qiymatlarni ifodalash uchun bitlar va baytlar qo’llaniladi va xotira baytlarda hisoblanadi.

O’zgaruvchilar turlari. O’zgaruvchilarning quyidagi turlari mavjud:

char – bitta simvol;

long char – uzun simvol;

int – butun son;

short yoki **short int** – qisqa butun son;

long yoki **long int** – uzun butun son;

float haqiqiy son;

long float yoki **double** – ikkilangan haqiqiy son;

long double – uzun ikkilangan haqiqiy son.

Butun sonlar ta’riflanganda ko’rilgan turlar oldiga unsigned (ishorasiz) ta’rifi qo’shilishi mumkin. Bu ta’rif qo’shilgan butun sonlar ustida amallar mod 2^n arifmetikasiga asoslangandir. Bu yerda n soni int turi xotirada egallovchi razryadlar sonidir. Agar ishoraciz k soni uzunligi int soni razryadlar sonidan uzun bo’lsa, bu son qiymati k mod 2^n ga teng bo’ladi. Ishorasiz k son uchun ga –k amali 2^n–k formula asosida hisoblanadi. Ishorali, ya’ni signed turidagi sonlarning eng katta razryadi son ishorasini ko’rsatish uchun ishlatilsa unsigned (ishorasiz) turdagi sonlarda bu razryad sonni tasvirlash uchun ishlatiladi.

O’zgaruvchilarni dasturning ixtiyoriy qismida ta’riflash yoki qayta ta’riflash mumkin.

Misol uchun:

int a, b1, ac; yoki

int a;

int b1;

int ac;

O’zgaruvchilar ta’riflanganda ularning qiymatlari aniqlanmagan bo’ladi. Lekin o’zgaruvchilarni ta’riflashda inisializasiya ya’ni boshlang’ich qiymatlarini ko’rsatish mumkin.

Misol uchun:

int i = 0;

char c = ’k’;

Typedef ta’riflovchisi yangi turlarni kiritishga imkon beradi.

Misol uchun yangi COD turini kiritish:

typedef unsigned char COD;

COD simbol;

Butun turlar o’lchami. Bir xil turdagi o’zgaruvchilar uchun turli kompyuterlarda xotiradan turli hajmdagi joy ajratilishi mumkin. Lekin bitta kompyuterda bir xil turdagi ikkita o’zgaruvchi bir xil miqdorda joy egallaydi.

Masalan, char turli o’zgaruvchi bir bayt hajmni egallaydi. Ko’pgina kompyuterlarda short int (qisqa butun) turi ikki bayt, long int turi esa 4 bayt joy egallaydi. Butun qiymatlar o’lchovini kompyuter sistemasi va ishlatiladigan kompilyator aniqlaydi. 32 – razryadli kompyuterlarda butun o’zgaruvchilar 4 bayt joy egallaydi.

3.2. Konstantalar

Konstantalar turlari. Konstanta bu o’zgartirish mumkin bo’lmagan qiymatdir. C tilida besh turdagi konstantalar ishlatilishi mumkin: simvollar, butun sonlar, haqiqiy sonlar, sanovchi konstantalar va nul ko’rsatkich.

Belgili o’zgarmaslar. Belgili o’zgarmaslar odatda bir bayt joyni egallaydi va bu 256 xil belgini saqlash uchun yetarlidir. Char turi qiymatlarini 0..255 sonlar to’plamiga yoki ASCII belgilar to’plamiga interpretasiya qilish mumkin.

ASCII belgilari deganda kompyuterlarda qo’llaniladigan standart belgilar to’plami tushuniladi. ASCII - bu American Standard Code for Information Interchange (Amerikaning axborot almashinishi uchun standart kodi) degan ma’noni anglatadi.

Misol uchun ‘x’,’*’,’\012’,’\0’,’\n’- bitta simvolli konstanta; ‘dd’,’\n\t’,’\x07\x07’ ikki simvolli konstantalar.

C kompilyatorida tekstlarni formatlovchi bir nechta maxsus belgilardan foydalaniladi. (Ulardan eng ko’p tarqalgani jadvalda keltirilgan).

Maxsus belgilar axborotlarni ekranga, faylga va boshqa chiqarish qurilmalariga chiqarishda formatlash uchun qo’llaniladi.

Maxsus '\' simvolidan boshlangan simvollar eskeyp simvollar deyiladi. Simvolli konstanta qiymati simvolning kompyuterda qabul qilingan sonli kodiga tengdir.

ESC (eskeyp) simvollar jadvali:

Yozilishi	Ichki kodi	Simvoli(nomi)	Ma’nosi
\a	0x07	bel (audible bell)	Tovush signali
\b	0x08	bs (bascspase)	Bir qadam qaytish
\f	0x0C	ff (form feed)	Sahifani o’tkazish
\n	0x0A	lf (line feed)	Qatorni o’tkazish
\r	0x0D	cr (carriage return)	Karetkani qaytarish
\t	0x09	ht (horizontal tab)	Gorizontal tabulyasiya

\v	0x0B	vt (vertical tab)	Vertikal tabulyasiya
\\	0x5C	\ (bacslash)	Teskari chiziq
\'	0x27	‘ (single out)	Apostrif (oddiy qavs)
\"	0x22	" (double quote)	Ikkilik qavs
\?	0x3F	? (question mark)	Savol belgisi
\000	000	ixtiyoriy (octal number)	Simvol sakkizlik kodi
\xhh	0xhh	ixtiyoriy (hex number)	Simvol o’n oltilik kodi

Ma’lumotlarning butun son turi. Butun sonlar o’nlik, sakkizlik yoki o’n oltilik sanoq sistemalarida berilishi mumkin.

O’nlik sanoq sistemasida butun sonlar 0-9 raqamlari ketma-ketligidan iborat bo’lib, birinchi raqami 0 bo’lishi kerak emas. Lekin yagona 0 bo’lishi mumkin.

Sakkizlik sanoq sistemasida butun sonlar 0 bilan boshlanuvchi 0-7 raqamlaridan iborat ketma-ketlikdir.

O’n oltilik sanoq sistemasida butun son 0x yoki 0X bilan boshlanuvchi 0-9 raqamlari va a-f yoki A-F harflaridan iborat ketma-ketlikdir.

Masalan, 15 va 22 o’nlik sonlari sakkizlikda 017 va 026, o’n oltilikda 0xF va 0x16 shaklda tasvirlanadi.

Ma’lumotlarning uzun butun son turi.

Oxiriga l yoki L harflari qo’yilgan o’nlik, sakkizlik yoki o’n oltilik butun son.

Ma’lumotlarning ishorasiz (unsigned) butun son turi:

Oxiriga u yoki U harflari qo’yilgan o’nlik, sakkizlik yoki o’n oltilik oddiy yoki uzun butun son.

Ma’lumotlarning haqiqiy son turi. Ma’lumotlarning haqiqiy son turi olti qismdan iborat bo’lishi mumkin: butun qism, nuqta, kasr qism, e yoki E belgisi, o’nlik daraja va F yoki f suffikslari.

Masalan : 66., .0, .12, 3.14F, 1.12e-12.

Ma’lumotlarning uzun haqiqiy son turi:

Oxiriga L yoki l suffikslari qo’yilgan haqiqiy son.

Masalan: 2E+6L;

Sanovchi konstanta. Sanovchi konstantalar enum xizmatchi so’zi yordamida kiritilib, int turidagi sonlarga qulay so’zlarni mos qo’yish uchun ishlatiladi.

Misol uchun:

enum{one = 1,two = 2,three = 3};

Agar son qiymatlari ko’rsatilmagan bo’lsa eng chapki so’zga 0 qiymati berilib qolganlariga tartib bo’yicha o’suvchi sonlar mos qo’yiladi:

enum{zero,one,two};

Bu misolda avtomatik ravishda konstantalar quyidagi qiymatlarni qabul kiladi:

zero = 0, one = 1, two = 2;

Konstantalar aralash ko’rinishda kiritilishi ham mumkin:

enum(zero,one,for = 4,five,seeks}.

Bu misolda avtomatik ravishda konstantalar quyidagi qiymatlarni qabul kiladi:

zero = 0, one = 1, for = 4;five = 5,seeks = 6;

Yana bir misol:

Enum BOOLEAN {NO, YES};

Konstantalar qiymatlari:

NO = 0, YES = 1;

Nul ko'rsatkich. NULL- ko'rsatkich yagona arifmetik bo'lmagan konstantadir. Konkret realizasiyalarda null ko'rsatkich 0 yoki 0L yoki nomlangan konstanta NULL orqali tasvirlanishi mumkin. Shuni aytish lozimki, bu konstanta qiymati 0 bo'lishi yoki '0' simvoli kodiga mos kelishi shart emas.

Mantiqiy konstanta. Mantiqiy konstantalar true(rost) va false(yolg'on) qiymatlardan iborat. C tilida butun sonlar va ifodalar mantiqiy konstantalar sifatida qaraladi. Ichki ko'rinishi false – 0, ixtiyoriy boshqa qiymat true deb qaraladi.

Satrli konstanta. Satrli konstantalar C tili konstantalariga kirmaydi, balki leksemalari alohida turi hisoblanadi. Shuning uchun adabiyotlarda satrli konstantalar satrli leksemalar deb ham ataladi.

Satrli konstanta bu ikkilik qavslarga olingan ixtiyoriy simvollar ketma-ketligidir. Misol uchun "Men satrli konstantaman".

Satrlar orasiga eskeyp simvollar ham kirishi mumkin. Bu simvollar oldiga \ belgisi qo'yiladi. Misol uchun :

"\n Bu satr \n uch qatorga \n joylashadi".

Satr simvollari xotirada ketma-ket joylashtiriladi va har bir satrli konstanta oxiriga avtomatik ravishda kompilyator tomonidan '\0' simvoli qo'shiladi. Shunday satrning xotiradagi hajmi simvollar soni +1 baytga tengdir.

Ketma-ket kelgan va bo'shliq, tabulyasiya yoki satr oxiri belgisi bilan ajratilgan satrlar kompilyasiya davrida bitta satrga aylantiriladi. Misol uchun:

"Salom" "Toshkent"

satrlari bitta satr deb qaraladi.

"Salom Toshkent"

Bu qoidaga bir necha qatorga yozilgan satrlar ham bo'ysunadi. Misol uchun:

"O'zbekistonga "

"bahor "

"keldi"

qatorlari bitta qatorga mos:

"O'zbekistonga bahor keldi"

Agar satrda '\' belgisi uchrasa va bu belgidan so'ng to '\n' satr oxiri belgisigacha bo'shliq belgisi kelsa bu bo'shliq belgilari '\' va '\n' belgisi bilan birga satrdan o'chiriladi. Satrning o'zi keyingi satrda kelgan satr bilan qo'shiladi.

"O'zbekistonga \

bahor \

keldi"

qatorlari bitta qatorga mos:

"O'zbekistonga bahor keldi"

Nomlangan konstantalar. C tilida o'zgaruvchilardan tashqari nomlangan konstantalar kiritilishi mumkin. Bu konstantalar qiymatlarini dasturda o'zgartirish mumkin emas. Konstantalar nomlari dasturchi tomonidan kiritilgan va xizmatchi so'zlardan farqli bo'lgan identifikatorlar bo'lishi mumkin. Odatda nom sifatida katta lotin harflari va ostiga chizish belgilari kombinasiyasidan iborat identifikatorlar ishlatiladi. Nomlangan konstantalar quyidagi shaklda kiritiladi:

const tur konstanta_nomi = konstanta_qiymati.

Misol uchun:

const double EULER = 2.718282;

const long M = 99999999;

const R = 765;

Oxirgi misolda konstanta turi ko’rsatilmagan, bu konstanta int turiga tegishli deb hisoblanadi.

3.3. Turlar bilan ishlash

Turlarni keltirish. Turlarni keltirish (type casting) ma’lum turdagi o’zgaruvchi boshqa turdagi qiymat qabul qilganda foydalaniladi. Ba’zi turlar uchun keltirish avtomatik ravishda bajariladi. Avtomatik turlarni keltirish o’zgaruvchi turi hajmi qiymatni saqlashga yetarli bo’lganda bajariladi. Bu jarayon kengaytirish *(widening)* yoki yuksaltirish *(promotion)* deb ataladi, chunki, kichik razryadli tur katta razryadli turga kengaytiriladi. Bu holda turlarni avtomatik keltirish xavfsiz deb ataladi. Masalan int turi char turidagi qiymatni saqlashga yetarli, shuning uchun turlarni keltirish talab qilinmaydi. Teskari jarayon toraytirish *(narrowing)* deb ataladi, chunki qiymatni o’zgartirish talab etiladi. Bu holda turlarni avtomatik keltirish xavfli deb ataladi. Masalan haqiqiy turni butun turga keltirilganda kasr qism tashlab yuboriladi.

Amallarda turlarni avtomatik keltirish. Binar arifmetik amallar bajarilganda turlarni keltirish quyidagi qoidalar asosida amalga oshiriladi:

short va char turlari int turiga keltiriladi;

Agar operandlardan biri long turiga tegishli bo’lsa ikkinchi operand ham long turiga keltiriladi va natija ham long turiga tegishli bo’ladi;

Agar operandlardan biri float turiga tegishli bo’lsa ikkinchi operand ham float turiga keltiriladi va natija ham float turiga tegishli bo’ladi;

Agar operandlardan biri double turiga tegishli bo’lsa ikkinchi operand ham double turiga keltiriladi va natija ham double turiga tegishli bo’ladi;

Agar operandlardan biri long double turiga tegishli bo’lsa ikkinchi operand ham long double turiga keltiriladi va natija ham long double turiga tegishli bo’ladi;

Ifodalarda turlarni avtomatik keltirish. Agar ifodada short va int turidagi o’zgaruvchilar ishlatilsa, butun ifoda tiri int ga ko’tariladi. Agar ifodada biror o’zgaruvchi turi— long bo’lsa, butun ifoda turi long turga ko’tariladi. Ko’zda tutilganidek hamma butun konstantalar int turiga ega deb qaraladi. Hamma butun konstantalar oxirida L yoki 1 simvoli turgan bo’lsa, long turiga ega.

Agar ifoda float turidagi operandga ega bo’lsa, butun ifoda float turiga ko’tariladi. Agar biror operand double turiga ega bo’lsa, butun ifoda turi double turiga ko’tariladi.

Turlar bilan ishlovchi amallar. Turlarni o’zgartirish amali quyidagi ko’rinishga ega:

(tur_nomi) operand;

Bu amal operandlar qiymatini ko’rsatilgan turga keltirish uchun ishlatiladi. Operand sifatida konstanta, o’zgaruvchi yoki qavslarga olingan ifoda kelishi mumkin. Misol uchun (long)6 amali konstanta qiymatini o’zgartirmagan holda operativ xotirada egallagan baytlar sonini oshiradi. Bu misolda konstanta turi o’zgarmagan bo’lsa, (double)6 yoki (float)6 amali konstanta ichki ko’rinishini ham o’zgartiradi. Katta butun sonlar haqiqiy turga keltirilganda sonning aniqligi yo’qolishi mumkin.

Masalan:

```
int x = 1.7+1.8;

int y = (int)1.7+(int)1.8;
```

Bu amallar bajarilishi natijasida x o’zgaruvchi qiymati 3 ga y o’zgaruvchi qiymati ikkiga teng bo’ladi.

sizeof amali operand sifatida ko’rsatilgan ob’ektning baytlarda xotiradagi hajmini hisoblash uchun ishlatiladi. Bu amalning ikki ko’rinishi mavjud:

sizeof ifoda

sizeof (tur)

Shuni ta’kidlab o’tish lozimki sizeof funksiyasi preprosessor qayta ishlash jarayonida bajariladi, shuning uchun dastur bajarilish jarayonida vaqt talab etmaydi.

Misol uchun:

sizeof 3.14 = 8

sizeof 3.14f = 4

sizeof 3.14L = 10

sizeof(char) = 1

sizeof(double) = 8.

Nazorat savollari

1. Zamonaviy kompyuterlarning kiritish va chiqarish vositalari qanchalik turlicha bo’lishi mumkin?
2. istream oqimi nima qiladi?
3. ostream oqimi nima qiladi?
4. Fayl nima?
5. Fayl formati nima?
6. Dasturga ma’lumotlar kiritish va chiqarish uchun ishlatiladigan 4 xil tipdagi qurilmalarni ayting..
7. Faylni o’qishning 4 bosqichini aytib bering.
8. Faylga yozishning 4 bosqichini aytib bering.
9. Oqimlarning 4 xil holatini ayting va aniqlang.
10. Ushbu kiritishga oid masalalarni yechish usullari haqida gapiring.

10.1. Foydalanuvchi mumkin bo’lgan diapazondan oshib ketuvchi qiymat kiritdi.

10.2. Ma’lumotlar tugadi (fayl oxiri).

10.3. Foydalanuvchi noto’g’ri tipdagi qiymat kiritdi.

11. Kiritishning chiqarishga nisbatan qiyinligi nimada?

12. Chiqarishning kiritishga nisbatan qiyinligi nimada?

Foydalanilgan adabiyotlar

1. Bjarne Stroustrup. Programming: Principles and Practice Using C++ (2nd Edition). Person Education, Inc. 2014. second printing, January 2015.
2. Harry Hariom Choudhary, Bjarne M Stroustrup. C++ Programming Professional.: Sixth Best Selling Edition for Beginner's & Expert's 2014.
3. http://www.stroustrup.com/4th.html
4. http://www.cplusplus.com/

4-Ma’ruza. C++ da kiritish chiqarish operatorlari va arifmetik amallar

Ma’ruza rejasi:

4.1. Ma’lumotlarni kiritish va chiqarish

4.2. Amallar

Kalit so’zlar:.

4.3. Ma’lumotlarni kiritish va chiqarish

Formatli chiqarish – printf. Chiqarish printf funksiyasi ko’rsatilgan parametrlarni standart oqimga chiqarish uchun ishlatiladi. Standart oqim tushunchasi keyingi boblarda yoritiladi. Xozircha standart oqim sifatida monitor tushunilishi yetarlidir.

Funksiya **stdio.h** modulida joylashgan bo’lib, umumiy ko’rinishi quyidagichadir:

printf(control,arg1,arg2,…)

Bunda control boshqaruvchi qator deb atalib ikki turdagi simvollardan iborat bo’ladi: oddiy chiqariluvchi simvollar va navbatdagi parametrni o’zgartirib chiqaruvchi spesifikasiyalar.

Har bir spesifikasiya % simvolidan boshlanib o’zgartirish turini ko’rsatuvchi simvol bilan tugaydi.

O’zgartirish simvollari quyidagilardan iborat.

Butun sonlar uchun:

d– parametr ishorali o’nlik butun songa aylantiriladi.

u - parametr ishorasiz o’nlik butun songa aylantiriladi.

o – parametr ishorasiz va birinchi raqami 0 bo'lmagan sakkizlik songa aylantiriladi.

x – parametr ishorasiz va 0x belgisiz o'n oltilik songa aylantiriladi.

X – parametr xuddi x kabi. Faqat harf bilan ko'rsatiluvchi raqamlar katta harf ya'ni A,B,C,D,E,F sifatida yoziladi.

Haqiqiy sonlar uchun:

e – parametr float yoki double turidagi son deb qaraladi va ishorali m.nnnnnne+-xx ko'rinishidagi o'nlik songa keltiriladi.

E – parametr xuddi e kabi. Faqat mantissa belgisi katta harf ya'ni E sifatida yoziladi.

f - parametr float yoki double turidagi son deb qaraladi va ishorali m.nnnnnn ko'rinishidagi o'nlik songa keltiriladi.

g – parametr berilgan son qiymati va aniqligi uchun eng ixcham %e yoki %f tanlaydi.

G – parametr xuddi g kabi. Faqat mantissa belgisi katta harf ya'ni E sifatida yoziladi.

Simvol va satr uchun:

c – parametr bitta simvol deb qaraladi.

s – parametr satr simvollar no'linchi simvol uchramaguncha yoki ko'rsatilgan sondagi simvollar bosiladi.

Misol:

```
#include <stdio.h>
int main()
{
int num = -27; int number = 27; float f = 123.456;
```

```
char r = 'a'; char str[4] = "abc";
printf("%d\n",    num);      /*    -27       */
printf("%u\n",    number);   /*    27        */
printf("%o\n",    number);   /*    33        */
printf("%x\n",    number);   /*    lb     */
printf( "%f\n",   f);        /*  123.456001   */
printf("%e\n",    f);        /*    1.23456e+02       */
printf("%E\n",    f);        /*    1.23456E+02       */
printf("%c\n",    r);        /*    a         */
printf("%s\n", str);         /*  abc       */
return 0;
}
```

Prosent % belgisi va o’zgartirish simvoli orasiga quyidagi simvollarni qo’yish mumkin.

Chiqarilayotgan argument chapga tekislash lozimligini ko’rsatuvchi minus belgisi.

Maydon minimal uzunligini ko’rsatuvchi raqamlar qatori.

Maydon uzunligini keyingi raqamlar qatoridan ajratuvchi nuqta.

Biror qatordan qancha simvol ajratib olish lozimligini hamda float yoki double turidagi sonlarda nuqtadan keyin qancha kasr raqamlari bosib chiqarilishini ko’rsatuvchi raqamlar ketma-ketligi.

Chiqarilayotgan son long turiga tegishli ekanligini ko’rsatuvchi uzun o’nlik markeri l.

% dan keyingi simvol o’zgartirish simvoli bo’lmasa u bosmaga chiqariladi.

% simvolini o’zini bosmaga chiqarish uchun %% belgisini berish lozim.

Quyidagi jadval har xil spesifikasiyalarni "HELLO, WORLD" (12 simvolli) so’zini bosishga ta’sirini ko’rsatadi. Bu yerda har bir maydon uzunligini ko’rsatish uchun maydon oxiriga ikki nuqta qo’yilgan.

:%10S:	:HELLO, WORLD:
:%10-S:	:HELLO, WORLD:
:%20S:	: HELLO, WORLD:
:%-20S:	:HELLO, WORLD :
:%20.10S:	: HELLO, WOR:
:%-20.10S:	:HELLO, WOR :
:%.10S:	:HELLO, WOR:

Turlar maksimal va minimal qiymatlari. Turli turlar maksimal va minimal qiymatlari <LIMITS.H> faylidagi konstantalarda saqlanadi.

Quyidagi dasturda char turidagi bitlar soni va char turi maksimal va minimal qiymati ekranga chiqariladi

```
#include<stdio.h>
#include<limits.h>
int main()
{
printf("CHAR_BIT = %d\n",CHAR_BIT);
printf("CHAR_MIN = %d CHAR_MAX =
%d\n",CHAR_MIN,CHAR_MAX);
return 0;
}
```

Quyidagi dasturda short, int, long turlari maksimal va minimal qiymati ekranga chiqariladi

```
#include<stdio.h>
#include<limits.h>
int main()
{
printf("SHRT_MIN = %d SHRT_MAX =
%d\n",SHRT_MIN,SHRT_MAX);
printf("INT_MIN = %d INT_MAX = %d\n",INT_MIN,INT_MAX);
printf("LONG_MIN = %ld LONG_MAX =
%ld\n",LONG_MIN,LONG_MAX);
return 0;
}
```

Quyidagi dasturda unsigned char, unsigned short, unsigned int, unsigned long turlari maksimal va minimal qiymati ekranga chiqariladi

```
#include<stdio.h>
#include<limits.h>
int main()
{
printf("UCHAR_MAX = %u\n",UCHAR_MAX);
printf("USHRT_MAX = %u\n",USHRT_MAX);
printf("UINT_MAX = %u\n",INT_MAX);
printf("ULONG_MAX = %ul\n",ULONG_MAX);
return 0;
}
```

Formatli kiritish Scanf. Scanf funksiyasi stdio.h modulida joylashgan bo'lib, umumiy ko'rinishi quyidagichadir:

Scanf(control, arg1, arg2,…)

Funksiya standart oqimdan simvollarni o'qib boshqaruvchi qator asosida formatlab mos parametrlarga yozib qo'yadi. Parametr ko'rsatkich bo'lishi lozim.

Boshqaruvchi qator quyidagi o’zgartirish spesifikasiyalaridan iborat:

Bo’shliq, tabulyasiya, keyingi qatorga o’tish simvollari;

Oddiy simvollar (% dan tashqari) kiritish oqimidagi navbatdagi simvollar bilan mos kelishi lozim;

% simvolidan boshlanuvchi spesifikasiya simvollari;

% simvolidan boshlanuvchi qiymat berishni ta’qiqlovchi * simvoli;

% simvolidan boshlanuvchi maydon maksimal uzunligini ko’rsatuvchi son;

quyidagi spesifikasiya simvollarini ishlatish mumkin:

d – ishorali o’nli butun son kutilmoqda.

o – ishorali sakkizlik butun son kutilmoqda.

x –ishorali o’n oltilik butun son kutilmoqda.

h - ishorasiz o’nlik son kutilmoqda.

c – bitta simvol kutilmoqda.

s – satr kutilmoqda.

f - float turidagi son kutilmoqda. Kiritilayotgan sonning butun raqamlari va nuqtadan so’ng kasr raqamlari soni va E yoki e belgisidan so’ng mantissa raqamlari soni ko’rsatilishi mumkin.

```
#include <stdio.h>
int main(void)
{
 unsigned width, precision;
 int number = 256;
 double weight = 242.5;
 printf("What field width?\n");
```

```
 scanf("%d", &width);
 printf("The number is :%*d:\n", width, number);
 printf("Now enter a width and a precision:\n");
 scanf("%d %d", &width, &precision);
 printf("Weight = %*.*f\n", width, precision, weight);
 printf("Done!\n");
 return 0;
}
```

Lokal va global o’zgaruvchilar. C tilida o’zgaruvchi ta’rifi albatta blok boshida joylashishi lozim.

O’zgaruvchi mavjudlik sohasi deb shu o’zgaruvchiga ajratilgan xotira mavjud bo’lgan dastur qismiga aytiladi. O’zgaruvchi ko’rinish sohasi deb o’zgaruvchi qiymatini olish mumkin bo’lgan dastur qismiga aytiladi. Biror blokda ta’riflangan o’zgaruvchi lokal o’zgaruvchi deyiladi. Har qanday blokdan tashqarida ta’riflangan o’zgaruvchi global o’zgaruvchi deyiladi.

Lokal o’zgaruvchi mavjudlik va ko’rinish sohasi ta’rifdan to shu ta’rif joylashgan blok oxirigachadir.

Tashqi blokdagi o’zgaruvchi nomi shu blokda joylashgan yoki shu blokdagi ichki blokda o’zgaruvchi nomi bilan bir xil bo’lmasligi kerak.

Global o’zgaruvchi mavjudlik sohasi ta’rifdan to dastur oxirigachadir.

Agar ichki blokdagi o’zgaruvchi nomi global o’zgaruvchi nomi bilan bir xil bo’lsa lokal o’zgaruvchi ko’rinish sohasida global o’zgaruvchi ko’rinmay qoladi.

Misol:

```
#include<stdio.h>
int i = 5;
int k = 6;
```

```
int main()
{
int i = 9;
printf("%d\n",i);
printf("%d\n",k);
return 0;
}
```

Natija:

9

6

4.2. Amallar

Arifmetik amallar. Amallar odatda unar, ya’ni bitta operandga qo’llaniladigan amallarga va binar, ya’ni ikki operandga qo’llaniladigan amallarga ajratiladi.

Binar amallar additiv ya’ni + qo’shish va – ayirish amallariga, hamda multiplikativ, ya’ni * ko’paytirish, / bo’lish va % modul olish amallariga ajratiladi.

Butun sonni butun songa bo’lganda natija butun songacha yaxlitlanadi. Misol uchun, 20/3 = 6; (-20)/3 = -6; 20/(-3) = -6.

Modul amali butun sonni butun songa bo’lishdan hosil bo’ladigan qoldiqqa tengdir. Agar modul amali musbat operandlarga qo’llanilsa, natija ham musbat bo’ladi, aks holda natija ishorasi kompilyatorga bog’liqdir.

Unar amallarga ishorani o’zgartiruvchi unar minus – va unar plyus + amallari kiradi. Bundan tashqari inkrement ++ va dekrement -- amallari ham unar amallarga kiradi.

Inkrement ++ unar amali qiymatni 1 ga oshirishni ko’rsatadi. Amalni prefiks, ya’ni ++i ko’rinishda ishlatish oldin o’zgaruvchi qiymatini oshirib, so’ngra foydalanish lozimligini, postfiks esa i++ ko’rinishda ishlatish oldin o’zgaruvchi qiymatidan foydalanib, so’ngra oshirish kerakligini ko’rsatadi. Misol uchun, i ning

qiymati 2 ga teng bo’lsin, u holda 3+(++i) ifoda qiymati 6 ga, 3+i++ ifoda qiymati 5 ga teng bo’ladi. Ikkala holda ham i ning qiymati 3 ga teng bo’ladi.

Dekrement -- unar amali qiymatni 1 ga kamaytirishni ko’rsatadi. Bu amal ham prefiks va postfiks ko’rinishda ishlatilishi mumkin. Bu ikki amalni faqat o’zgaruvchilarga qo’llash mumkin.

Amallar ustivorligi. Murakkab ifodalarda qaysi amal birinchi navbatda bajarilishi operator prioritetiga bog’liq.

Masalan: x = 5+3*8.

Ko’paytirish qo’shishga nisbatan yuqoriroq prioritetga ega. Shuning uchun bu ifoda qiymati 29 ga teng bo’ladi.

Agarda ikkita matematik ifodaning prioriteti teng bo’lsa, ular chapdan o’ngga qarab ketma-ket bajariladi.

Masalan: x = 5+3+8*9+6*4.

Bu ifodada birinchi ko’paytirish amallari chapdan o’ngga qarab bajariladi 8*9 = 72 va 6*4 = 24. Keyin qo’shish amallari bajariladi. Natijada x = 104 qiymatga ega bo’ladi.

Lekin, barcha amallar ham bu tartibga amal qilmaydi. Masalan, o’zlashtirish amali o’ngdan chapga qarab bajariladi.

Additiv amallarining ustivorligi multiplikativ amallarining ustivorligidan pastrokdir.

Unar amallarning ustivorligi binar amallardan yuqoridir.

Razryadli amallar. Razryadli amallar natijasi butun sonlarni ikkilik ko’rinishlarining har bir razryadiga mos mantiqiy amallarni qo’llashdan hosil bo’ladi. Masalan, 5 kodi 101 ga teng va 6 kodi 110 ga teng.

6&5 qiymati 4 ga, ya’ni 100 ga teng.

6|5 qiymati 7 ga, ya’ni 111 ga teng.

6^5 qiymati 3 ga, ya’ni 011 ga teng.

~6 qiymati 4 ga, ya’ni 010 ga teng.

Bu misollarda amallar ustivorligi oshib borishi tartibida berilgandir.

Bu amallardan tashqari M<<N chapga razryadli siljitish va M>>N o’ngga razryadli siljitish amallari qo’llaniladi. Siljitish M butun sonning razryadli ko’rinishiga qo’llaniladi. N nechta pozisiyaga siljitish kerakligini ko’rsatadi.

Chapga N pozisiyaga surish bu operand qiymatini ikkining N chi darajasiga ko’paytirishga mos keladi. Misol uchun 5<<2 = 20. Bu amalning bitli ko’rinishi: 101<<2 = 10100.

Agar operand musbat bo’lsa, N pozisiyaga o’ngga surish chap operandni ikkining N chi darajasiga bo’lib kasr qismini tashlab yuborishga mosdir. Misol uchun 5>>2 = 1. Bu amalning bitli ko’rinishi 101>>2 = 001 = 1. Agarda operand qiymati manfiy bo’lsa ikki variant mavjuddir: arifmetik siljitishda bo’shatilayotgan razryadlar ishora razryadi qiymati bilan to’ldiriladi, mantiqiy siljitishda bo’shatilayotgan razryadlar nullar bilan to’ldiriladi.

Razryadli surish amallarining ustivorligi o’zaro teng, razryadli inkor amalidan past, qolgan razryadli amallardan yuqoridir. Razryadli inkor amali unar amalga qolgan amallar binar amallarga kiradi.

Nisbat amallari. Nisbat amallari qiymatlari 1 ga teng agar nisbat bajarilsa va aksincha 0 ga tengdir. Nisbat amallari arifmetik turdagi operandlarga yoki ko’rsatkichlarga qo’llaniladi.

Misollar:

1! = 0 qiymati 1 ga teng;

1 == 0 qiymati 0 ga teng;

3> = 3 qiymati 1 ga teng;

3>3 qiymati 0 ga teng;

2< = 2 qiymati 1 ga teng;

2<2 qiymati 0 ga teng;

Katta >, kichik <, katta yoki teng > = , kichik yoki teng < = amallarining ustivorligi bir xildir.

Teng = = va teng emas ! = amallarining ustivorligi o'zaro teng va qolgan amallardan pastdir.

Mantiqiy amallar. C tilida mantiqiy tur yo'q. Shuning uchun mantiqiy amallar butun sonlarga qo'llanadi. Bu amallarning natijalari quyidagicha aniqlanadi:

x||y amali 1 ga teng agar x>0 yoki y>0 bo'lsa, aksincha 0 ga teng

x&&y amali 1 ga teng agar x>0 va y>0 bo'lsa, aksincha 0 ga teng

!x amali 1 ga teng agar x>0 bo'lsa, aksincha 0 ga teng

Bu misollarda amallar ustivorligi oshib borish tartibida berilgandir.

Inkor ! amali unar qolganlari binar amallardir.

Qiymat berish amali. Qiymat berish amali = binar amal bo'lib chap operandi odatda o'zgaruvchi o'ng operandi esa ifodaga teng bo'ladi. Misol uchun

z = 4.7+3.34

Bu qiymati 8.04 ga teng ifodadir. Bu qiymat Z o'zgaruvchiga ham beriladi.

Bu ifoda oxiriga nuqta vergul (;) belgisi qo'yilganda operatorga aylanadi.

z = 4.7+3.34

Bitta ifodada bir necha qiymat berish amallari qo'llanilishi mumkin. Misol uchun:

c = y = f = 4.2+2.8;

Bundan tashqari C tilida murakkab qiymat berish amali mavjud bo’lib, umumiy ko’rinishi quyidagichadir:

O’zgaruvchi_nomi **amal** = ifoda;

Bu yerda **amal** quyidagi amallardan biri *,/,%,+,-, &,^,|, <<,>>.

Misol uchun:

x+ = 4 ifoda x = x+4 ifodaga ekvivalentdir;

x* = a ifoda x = x*a ifodaga ekvivalentdir;

x/ = a+b ifoda x = x/(a+b) ifodaga ekvivalentdir;

x>> = 4 ifoda x = x>>4 ifodaga ekvivalentdir;

Imlo belgilari amal sifatida. C tilida ba’zi bir imlo belgilari ham amal sifatida ishlatilishi mumkin. Bu belgilar oddiy () va kvadrat [] qavslardir. Oddiy qavslar binar amal deb qaralib ifodalarga yoki funksiyaga murojaat qilishda foydalaniladi. Funksiyaga murojaat qilish quyidagi shaklda amlga oshiriladi:

<funksiya nomi> (<argumentlar ro’yxati>). Misol uchun sin(x) yoki max(a,b).

Kvadrat qavslardan massivlarga murojaat qilishda foydalaniladi. Bu murojaat quyidagicha amalga oshiriladi:

<massiv nomi>[<indeks>]. Misol uchun a[5] yoki b[n][m].

Vergul simvolini ajratuvchi belgi sifatida ham amal sifatida ham qarash mumkin. Vergul bilan ajratilgan amallar ketma-ketligi bir amal deb qaralib, chapdan o’ngga hisoblanadi va oxirgi ifoda qiymati natija deb qaraladi. Misol uchun:

d = 4,d+2 amali natijasi 6 ga teng.

Shartli amal. Shartli amal ternar amal deyiladi va uchta operanddan iborat bo’ladi:

<1-ifoda>?<2-ifoda>:<3-ifoda>

Shartli amal bajarilganda avval 1- ifoda hisoblanadi. Agar 1-ifoda qiymati 0 dan farqli bo'lsa 2- ifoda hisoblanadi va qiymati natija sifatida qabul qilinadi, aks holda 3-ifoda hisoblanadi va qiymati natija sifatida qabul qilinadi.

Misol uchun modulni hisoblash: x<0?-x:x yoki ikkita son kichigini hisoblash a<b?a:b.

Shuni aytish lozimki shartli ifodadan har qanday ifoda sifatida foydalanish mumkin. Agar F FLOAT turpga,a N – INT turga tegishli bo'lsa,

(N > 0) ? F : N ifoda N musbat yoki manfiyligidan qat'iy nazar DOUBLE turiga tegishli bo'ladi.

Shartli ifodada birinchi ifodani qavsga olish shart emas.

Amallar ustivorligi jadvali

Rang	Amallar	Yo'nalish
1	() [] -> :: .	Chapdan o'ngga
2	! ~ + - ++ -- & * (tur) sizeof new delete tur()	O'ngdan chapga
3	. * ->*	Chapdan o'ngga
4	* / % (multiplikativ binar amallar)	Chapdan o'ngga
5	+ - (additiv binar amallar)	Chapdan o'ngga
6	<< >>	Chapdan o'ngga
7	< < = > = >	Chapdan o'ngga
8	= ! =	Chapdan o'ngga
9	&	Chapdan o'ngga
10	^	Chapdan o'ngga

11	|	Chapdan o’ngga
12	&&	Chapdan o’ngga
13	||	Chapdan o’ngga
14	?:(shartli amal)	O’ngdan chapga
15	= * = / = % = + = - = & = ^ = | = << = >> =	O’ngdan chapga
16	, (vergul amali)	Chapdan o’ngga

Nazorat savollari

1. Butun sonli va haqiqiy turlarni qanday farqi bor?
2. Ishorasiz unsigned turining xossalarini ko’rsating.
3. Ishorasiz unsigned short int va long int turlarining o’zaro farqi nimada?
4. Birinchi qaysi funksiya bajariladi?
5. Simvolli kiritish funksiyalari.
6. Shartli amal umumiy ko’rinishi.
7. Turlarni keltirish qoidalari.
8. Quyidagi #include direktivasi qanday vazifani bajaradi.
9. Bosh main() funksiyasining o’ziga xos xususiyati nimadan iborat?
10. Izohlar bir necha qatorda yozilishi mumkinmi?

5-Ma’ruza. C++ da satrli kattaliklar v ular bilan ishlash

Ma’ruza rejasi:

5.1 Belgili axborot va satrlar

5.2 So’zlar massivlari

5.3 Izlash va tartiblash

Kalit so’zlar: *delete, masofa keltirish, delete[], new, indeks, this, indeksirlash, [] bo’sh xotira, void*, konteyner, ro’yxat, manzil, nolinchi ko’rchsatkich, tugun, adres olish &, bo’shatish, ko’rsatkich, virtual destruktor, xotira, xotira chiqishi, destruktor, toifani o’zlashtirish, resurslar chiqishi, a’zo destruktori.*

5.1. Belgili axborot va satrlar

Satrlar. C da belgili ma’lumotlar uchun char turi qabul qilingan. Belgili axborotni taqdim etishda belgilar, simvolli o’zgaruvchilar va matnli konstantalar qabul qilingan.

Misollar:

```
const char c = 'c';

char a,b;
```

S dagi satr - bu nul-belgi - \0 (nul-terminator)- bilan tugallanuvchi belgilar massivi. Nul-terminatorning holatiga qarab satrning amaldagi uzunligi aniqlanadi. Bunday massivdagi elementlar soni, satr tasviriga qaraganda, bittaga ko’p.

Simvolli massivlar quyidagicha inisializasiya qilinadi:

char capital[] = "TASHKENT"; Bu holda avtomatik ravishda massiv elementlari soni aniqlanadi va massiv oxiriga satr ko’chirish '\0' simvoli qo’shiladi.

Yuqoridagi inisializasiyani quyidagicha amalga oshirish mumkin:

```
    char capital[] =
{'T','A','S','H','K','E','N','T','\0'};
```

Bu holda so’z oxirida '\0' simvoli aniq ko’rsatilishi shart.

Qiymat berish operatori yordamida satrga qiymat berish mumkin emas. Satrni massivga yoki kiritish paytida yoki nomlantirish yordamida joylashtirish mumkin.

```
    #include <stdio.h>
    #include <stdlib.h>
    int main()
    {
    char s1[10] = "string1";
    int k = sizeof(s1);
    printf("\n%s %d",s1,k);
    char s2[] = "string2";
    k = sizeof(s2);
    printf("\n%s %d",s2,k);
    char s3[] = {'s','t','r','i','n','g','3','\0'};
    k = sizeof(s3);
    printf("\n%s %d",s3,k);
    char *s4 = "string4";//satr ko'rsatkichi, uni
o'zgartirib bo'lmaydi
    k = sizeof(s4);
    printf("\n%s %d",s4,k);
    system("pause");
    return 0;
    }
```

Natija:

string1 10

string2 8

string3 8

string4 4

Keyingi misolda kiritilgan so’zdan berilgan harfni olib tashlash dasturi berilgan.

```
#include <stdio.h>
int main()
{
char s[100];
scanf("%s",&s);
int i, j;
for ( i = j = 0; s[i] ! = '\0'; i++)
if ( s[i] ! = 'c' )
s[j++] = s[i];
s[j] = '\0';
printf("%s",s);
return 0;
}
```

Xar safar 's' dan farqli simvol uchraganda, u j pozisiyaga yoziladi va faqat shundan so’ng j ning qiymati 1 ga oshadi. Bu quyidagi yozuvga ekvivalent:

```
if ( s[i] ! = c )
s[j] = s[i];
j++;
```

Funksiyalar va satrlar. Funksiyalarda satrlar ishlatilganda ularning chegarasini ko’rsatish shart emas. Satrlarning uzunligini hisoblash len funksiyasini quyidagicha ta’riflash mumkin:

```
int len(char c[])
{ int m = 0;
```

```
for(m = 0;c[m]! = ’0’; m++);
return m;
};
```

Shu funksiyadan foydalanilgan dasturni keltiramiz:

```
#include <stdio.h>
int len(char c[])
{
int m = 0;
while(c[m++]);
return m-1;
};
int main()
{
char e[] = "Pro Tempore!";
printf("\n%d", len(e));
return 0;
}
```

Bu funksiyaning standart varianti strlen deb ataladi va bu funksiyadan foydalanish uchun string.h sarlavha faylidan foydalanish lozim.

Satrdan nusxa olish funksiyasi strcpy ni C tilida quyidagicha ta’riflash mumkin:

```
#include <stdio.h>
void strlen(char s1[], char s2[])
{
int i = 0;
while(s2[i]! = '\0') s1[i++] = s2[i];
s1[i] = s2[i];
}
```

```
int main()
{
char s1[] = "aaa";
char s2[] = "ddd";
strcpy(s1,s2);
printf("%s",s1);
return 0;
}
```

Natija:

ddd

Berilgan satrni teskariga aylantiruvchi funksiya:

```
reverse(char s[])
{
int c, i, j;
for(i = 0, j = strlen(s) - 1; i < j; i++, j--)
c = s[i];
s[i] = s[j];
s[j] = c;
}
```

Keyingi misolimizda T qatorni S qator oxiriga ulovchi STRCAT(S,T) funksiyasini ko’rib chiqamiz:

```
strcat(char s[], t[])
 {
int i, j;
i = j = 0;
while (s[i] ! = '\0')
i++;
while((s[i++] = t[j++]) ! = '\0')
```

```
}
```

5.2. So’zlar massivlari

So’zlar massivini kiritish. C tilida so’zlar massivlari ikki o’lchovli simvolli massivlar sifatida ta’riflanadi. Misol uchun:

char name[4][5].

Bu ta’rif yordamida har biri 5 ta harfdan iborat bo’lgan 4 ta so’zli massiv kiritiladi. So’zlar massivlari quyidagicha inisializasiya qilinishi mumkin:

char Name[3][8] = { "Anvar","Mirkomil","Yusuf"}.

Bu ta’rifda har bir so’z uchun xotiradan 8 bayt joy ajratiladi va har bir so’z oxiriga ' \0' belgisi kuyiladi.

So’zlar massivlari inisializasiya qilinganda so’zlar soni ko’rsatilmasligi mumkin. Bu holda so’zlar soni avtomatik aniqlanadi:

char comp[][9] = { "kompyuter","printer","kartridj"}.

Quyidagi dasturda berilgan harf bilan boshlanuvchi so’zlar ro’yxati bosib chiqariladi:

```
#include <stdio.h>
int main()
{
char a[10][10];
char c = 'a';
int i;
for (i = 0;i<3;i++) scanf("%s",&a[i]);
for (i = 0;i<3;i++)
if (a[i][0] == c) printf("\n%s",a[i]);
return 0;
```

```
}
```

Quyidagi dasturda fan nomi, talabalar ro'yxati va ularning baholari kiritiladi. Dastur bajarilganda ikki olgan talabalar ro'yxati bosib chiqariladi:

```
#include <stdio.h>
int main()
{
char a[10][10];
char s[10];
int k[10];
scanf("%s",&s);
for (int i = 0;i<3;i++)
{
scanf("%s",&a[i]);
scanf("%d",&k[i]);
};
for (int i = 0;i<3;i++)
if (k[i] == 2) printf("%s\n",a[i]);
return 0;
}
```

Funksiyalar va so'zlar massivlari. Satrli massivlar funksiya argumenti sifatida ishlatilganda satrlarning umumiy uzunligi aniq ko'rsatilishi shart.

Misol tariqasida ixtiyoriy sondagi satrlar massivini alfavit bo'yicha tartiblash funksiyasidan foydalanilgan dasturni ko'rib chiqamiz:

```
#include <stdio.h>
#define m 10
void sort(int n, char a[][m])
{
char c;
```

```
int i,j,l;
for (i = 0;i<n;i++)
for (j = i+1;j<m;j++)
if (a[i][0]<a[j][0] )
for(l = 0;l<m;l++)
{
c = a[i][l];
a[i][l] = a[j][l];
a[j][l] = c;
};
};
int main()
{
char aa[][m] = {"Alimov","Dadashev","Boboev"};
sort(3,aa);
for(int i = 0; i<3;i++) printf("%s\n",aa[i]);
return 0;
}
```

5.3. Izlash va tartiblash

Ikkiga bo’lib izlash. Quyidagi dasturda tartiblangan massivda ikkiga bo’lib kalit sonni izlash algoritmi asosida tuzilgan funksiyadan foydalanish keltirilgan:

```
#include<stdio.h>
#include<conio.h>
int bsearch(int a[],int key,int n)
{
int m1,m2,m;
m1 = 0;m2 = n-1;
while(m1< = m2)
```

```
{
m = (m2+m1)/2;
if (a[m] == key) return m;
if (a[m]>key) m2 = --m;
if (a[m]<key) m1 = ++m;
}
return -1;
};
int main(int argc, char* argv[])
{
int m;
int a[] = {5,6,9,11};
m = bsearch(a,7,4);
printf("%d",m);
getch();
return 0;
}
```

Keyingi misolda shu funksiya satrlar uchun varianti keltirilgan:

```
#include<stdio.h>
#include<string.h>
#include<conio.h>
#define size 5
int strbsearch(char a[][size],char key[],int n)
{
int m1,m2,m,pr;
m1 = 0;m2 = n-1;
while(m1< = m2)
{
```

```
m = (m2+m1)/2;
pr = strcmp(a[m],key);
if (pr == 0) return m;
if (pr == -1) m2 = --m;
if (pr == 1) m1 = ++m;
}
return -1;
};
int main(int argc, char* argv[])
{
int m;
char a[][size] = {"aaa","aab","aac","aad"};
m = strbsearch(a,"aab",4);
printf("%d",m);
getch();
return 0;
}
```

Tezkor tartiblash. Quyidagi dasturda tezkor tartiblash algoritmiga asoslangan funksiyadan foydalanilgan. Algoritm mohiyati shundan iboratki, avval yetakchi element tanlanadi. Funksiyada yetakchi element sifatida boshlang'ich element tanlanadi. Shundan so'ng massiv ikki qismga ajratiladi. Yetakchi elementdan kichik elementlar past qismga, katta elementlar yuqori qismga to'planadi. Shundan so'ng rekursiya asosida algoritm ikkala qismga alohida qo'llanadi.

```
#include<stdio.h>
#include<conio.h>
int a[] = {5,4};
void sqsort(int k1, int k2)
{
```

```
if(k1 < k2){
int i, j, k;
i = k1; j = k2;
while(i < j)
{
if (a[k1] > a[i]) {i++; continue;}
if (a[k1] < a[j]) {j--; continue;}
k = a[j]; a[j] = a[i]; a[i] = k;
}
k = a[k1]; a[k1] = a[i]; a[i] = k;
sqsort(k1, i);
sqsort(i+1, k2);
};
}
int main(int argc, char* argv[])
{
int i;
sqsort(0, 1);
for(i = 0; i < 2; i++) printf("%d ", a[i]);
getch();
return 0;
}
```

Nazorat savollari

1. Satr simvolli massivdan qanday farq qiladi?
2. Bir o’lchovli massivlarni inisializasiya qilish usullarini ko’rsating.
3. Ko’p o’lchovli massiv ta’rifi xususiyatlarini keltiring.
4. Ko’p o’lchovli masivlar inisializasiyasi xususiyatlari.
5. Satrlarni inisializasiya qilish usullarini ko’rsating.

6. So’zlar massivi qanday kiritiladi?
7. Qanday qilib bir o’lchovli massivlar formal parametrlar sifatida ishlatilishi mumkin?
8. Qanday qilib ko’p o’lchovli massivlar formal parametrlar sifatida ishlatilishi mumkin?
9. Satr ta’riflash usullari.
10. Satrlar funksiya parametri sifatida.

6-Ma’ruza. Tarmoqlanish operatorlari: sonlar va belgilarni solishtrish

Ma’ruza rejasi:

6.1 O’tish operatori.

6.2 Shartli operatorning qisqa ko’rinishi.

6.3 Shartli operatorning uzun ko’rinishi.

6.4 Tanlash operatori.

Kalit so’zlar: *mantiqiy qo’shish va ko’paytirish, inkor amali, tarmoqlanish, shartli operator, o’tish operatori, tanlash operatori*

Ko’p masalalarning yechimi ma’lum bir shart yoki shartlarning qo’yilishiga qarab bajariladi. Bunday jarayonlarni tarmoqlanuvchi hisoblash jarayoni deyiladi. Tarmoqlanuvchi hisoblash jarayonlari tarkibida yana tarmoqlanish bo’lishi mumkin. Bundaylarni murakkab tarmoqlanuvchi jarayonlar deb ataladi. Algoritmik tilda kattaliklarning istalgan xossasi shu ondagi qiymatlari uchun bajarilishi yoki bajarilmasligi **shart** deyiladi. Masalan: a = v tenglik uchun a=3 va v=3,1 bo’lganda shart bajarilmaydi. N tub son deyilsa va N=19 bo’lsa, shart bajariladi, N=15 bo’lganda shart bajarilmaydi.

Tarmoqlanuvchi jarayonlarni tashkil etishda shartsiz o’tish va shartli o’tish operatorlaridan foydalaniladi. Shunday jarayonlar mavjudki, shartning bajarilishiga qarab, dasturning u yoki bu qismiga o’tishga to’g’ri keladi. Bunday hollarda shartsiz o’tish operatori ishlatiladi.

1. Shartsiz o’tish operatori: **goto n;** bu yerda n – metka, belgi bo’lib, jarayon o’tishi kerak bo’lgan joyni ko’rsatadi. Metka xarf, son va ular aralashmasidan iborat bo’lishi mumkin. 1ta operatorga bir nechta metkalarni qo’yish mumkin. (Usta dasturchilar goto n operatoridan kamroq foydalanadilar.)

2. Shartli o’tish operatori: **if (shart) operator;** Uning ishlashi quyidagicha: agar shart rost bo’lsa keltirilgan operator bajariladi, agar shart yolg’on bo’lsa, keyingi qatorga o’tiladi. Ko’pincha bu ko’rinish ishlatilganda 2ta operatorlar

aralashib ketmasligi uchun goto operatori ishlatildi. Agar if so'zidan keyin bir nechta operatorlar keladigan bo'lsa, ularni alohida {} qavslarga olinadi. (bu usul kamroq ishlatiladi) Masalan:

$$Y = \begin{cases} \text{Sin x, agar x <5} \\ \sqrt[3]{x^2}\text{, agar } x \geq 5 \end{cases}$$

```
     # include <iostream.h>
     # include <math.h>
     void main ( )
     {  float x, y;                          // x va u ning
toifasi xaqiqiy
         cin >> x;                          // x ning son
qiymati kiritiladi
         if  (x<5)
// agar  x<5 bo'lsa
       { y=sin(x); goto cc; }                    // 1-
funksiya ishlaydi
          y=pow(x, 2/3.);                    // aks xolda
2-funksiya ishlaydi
     cc: cout << "y="<< y<< endl;     // u ning javobi
beriladi. ss-metka
          }
// main funksiyasi berkitildi.
```

3. if (shart) 1-operator(lar); else 2-operator(lar);

Masalan: yukoridagi misolni kurib utamiz:

```
     # include <iostream.h>
     # include <math.h>
```

```
void main ( )
{  float x, y;
    cin >> x;
    if  (x<5)   y=sin(x); else   y=pow(x, 2/3.);
    cout << "y="<< y<< endl;   }
```

Izox: if – else konstruksiyasi ichida yana if – else konstruksiyasi ishlatilishi mumkin. Bunda bir nechta if operatoridan iborat ichma-ich joylashgan konstruksiya xosil buladi. Bunday xollarda else sO'zi O'ziga yakin turgan if ga tegishli buladi.

shart ? 1-operator (lar) : 2-operator (lar);

Masalan: x <5 ? y=sin(x) : y=pow(x,2/3.);

(agar shart rost bulsa, 1-operator, aks xolda 2-operator bajariladi)

Tanlash operatori – O'zgaruvchining kiymatiga karab kup tarmok ichidan bittasi tanlanadi. Bu operatorning kurinishi kuyidagicha:

```
switch  (ifoda yoki o'zgaruvchi)
{ case  1-qiymat :  operator(lar); break;
   case  2-qiymat :  operator(lar); break;
   case  n - qiymat : operator(lar); break;
   default :  aks holdagi operator (lar);    }
```

Masalan:

$$Y=\begin{cases}\sin x, & \text{agar } x=1\\ \cos x, & \text{agar } x=2\\ \operatorname{tg} x, & \text{agar } x=3\\ \sqrt{x}, & \text{boshka barcha xollarda } (x>0)\end{cases}$$

```
# include <iostream.h>
# include <math.h>
void main ( )
{ int x;  float y;
cin >> x;
switch  (x)
```

```
{ case  1 :  y=sin(x);  break;
   case  2 :  y=cos(x); break;
   case  3 : tan(x);       break;
   default :  y=sqrt(x);  }
cout << "y="<<y<<"x="<<x<<endl;
getch ( );  }
```

Izox: 1) switch operatoridagi ifoda yoki o'zgaruvchi butun toifali bo'lishi shart!

2) switch operatori satrlaridagi break sO'zi tushib kolsa, joriy case operatoridan keyingi case bloklari ichidagi ifodalar xam bajarilaveradi.

Tekshirilayotgan shartlar bir nechta bo'lishi xam mumkin. Bunday xollarda ularni murakkab shart deyiladi. Bunday shartlarni kuyidagi mantikiy amallar orkali ifoda etiladi:

&& - mantikiy kupaytirish (va)

|| – mantikiy kushish (yoki)

! - mantikiy inkor (emas)

Masalan: $6 \le x \le 10$ bulsa, (x >=6) && (x<=10)

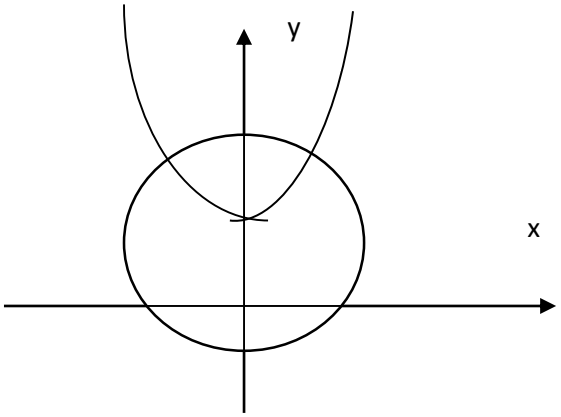
$Y > 0$ va $x < 4$ yoki $z \ge 5$ bylsa, (y >0) && (x<4) || (z>5)

if (sin(x) > 1) && (5 / 2 = 0) y:=1; else y:=0; {ikkala shart xam yolgon kiymatga ega, shuning uchun y=0 buladi.}

Mantikiy kiymatlar ustida amallar bajarilganida kuyidagi natijalar olinadi: (+ true 1; - false 0 degan ma'noda)

A	**B**	**! A**	**!B**	**A && B**	**A || B**
+	+	-	-	+	+
+	-	-	+	-	+
-	+	+	-	-	+
-	-	+	+	-	-

Mantiqiy masala: ixtiyoriy berilgan M(x,u) nukta $u=x^2$ va $x^2+u^2=4$ aylana bilan kesishgan sohaga yoki shu aylananing 4-choragi tashqarisiga tushishini tekshiring.

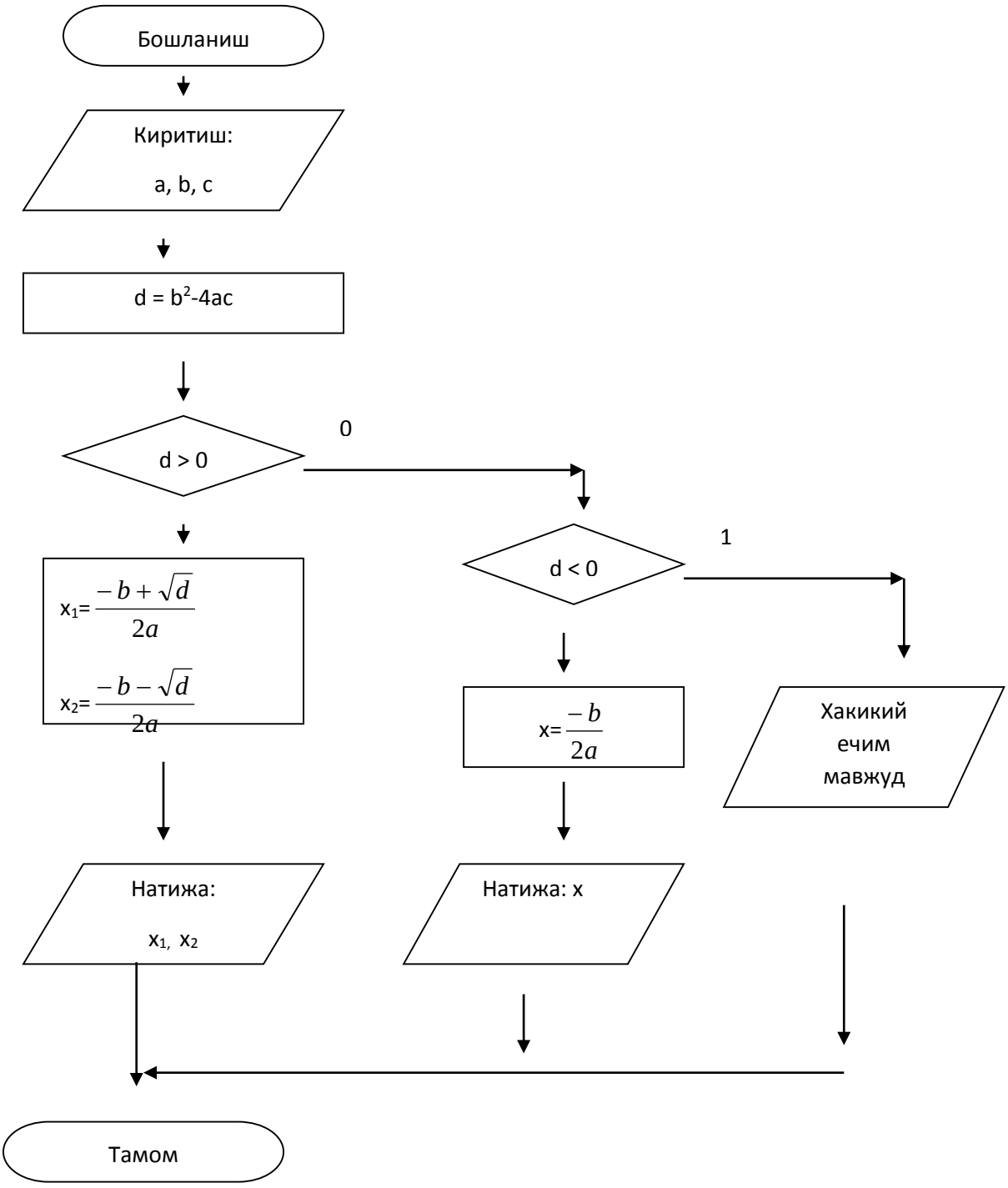


Демак:

$y > x^2$ and $x^2 + y^2 <= 4$
or x > 0 and y < 0 and
$x^2 + y^2 > = 4$
x = 1, y = 1 → false
x=1, y=0 → true
x = -2, y = 0.5 → false
x =2, y= -2 → true

```
# include   <iostream. h>
void main ( )
{   float x, y;   int n;
cout << "nuqtaning koordinatalarini kiriting:";
 cin >> x>>y;
if  ((y>=x*x  && x*x+y*y<=4) ||
(x>0 && y<0  &&  x*x+y*y>=4)) n=1;
else n=0;
cout << "x="<<x<<endl;
cout << "y="<<y<< endl;  cout << "n="<<n<< endl;  }
```

3-misol: $ax^2+bx+c=0$ kurinishdagi kvadrat tenglamaning xakikiy yechimlarini topish algoritmini tO'zing. (a≠0; b≠0; c≠0;)


Бошланиш
Киритиш:
a, b, c
d = b²-4ac
d > 0
0
d < 0
1
x₁= (−b + √d) / 2a
x₂= (−b − √d) / 2a
x= −b / 2a
Хакикий ечим мавжуд
Натижа:
x₁, x₂
Натижа: x
Тамом

```
    # include <iostream.h>
    # include <math.h>
    void main ( )
    float a, b, c, d, x, x1,
x2;
    cout << "Tenglamaning
koeffisientlarini kiriting:
";
    cin >> a>>b>>c;
    d = b*b - 4*a*c;
    if  ( d == 0 )  { x=- b
/ (2*a);
    cout <<"x="<<x<<endl;
goto  b15; }
    if  ( d > 0 )
    { x1 = (- b + sqrt(d)) /
(2*a);
       x2 = (- b - sqrt(d))
/ (2*a);
    cout
<<"x1="<<x1<<"x2="<<x2<<endl;
}
    else  cout <<"yechimi
yuk"<< endl;
    b15 : }
```

```
    # include <iostream.h>
    # include <math.h>
    void main ( )
    float a, b, c, d, x, x1,
x2;  int v;
    cout << "Tenglamaning
koeffisientlari:";
    cin >> a>>b>>c;
    d = b*b - 4*a*c;
    if  (d<0)  v=0;
    if  (d = = 0)  v=1; else
v=2;
    switch ( v )
    { case 0: cout
<<"yechimi yuk"<< endl;
break;
    case  1 : { x=- b /
(2*a);
    cout <<"  x="<<x<<endl;
}   break;
    case  2:  { x1 = (- b +
sqrt(d)) / (2*a);
       x2 = (- b - sqrt(d))
/ (2*a);
    cout
<<"x1="<<x1<<"x2="<<x2<<endl;
}  break; }
```

Nazorat savollari:

1. Shartli o'tish operatorining vazifasi
2. Shartli o'tish operatorlarining ko'rinishlari
3. Murakkab operatorlar qachon va qanday qo'llaniladi?
4. Shartli o'tish operatori ichida yana shartli operator qatnashishi mumkinmi?
5. Shartni ifodalovchi blok sxema tO'zing.
6. Tanlash operatorining vazifasi.
7. Tanlash operatoridagi **break** ning vazifasi.

8. Tanlash operatoridagi o’zgaruvchilarning tiplari qanday bo’lishi kerak?

7-Ma’ruza. Ko’p tarmoqlanishlar va variant tanlash operatorlari

Ma’ruza rejasi:

7.1 Tanlash operatorlari

Kalit so’zlar: *delete, masofa keltirish, delete[], new, indeks, this, indeksirlash, [] bo’sh xotira, void*, konteyner, ro’yxat, manzil, nolinchi ko’rchsatkich, tugun, adres olish &, bo’shatish, ko’rsatkich, virtual destruktor, xotira, xotira chiqishi, destruktor, toifani o’zlashtirish, resurslar chiqishi, a’zo destruktori.*

Tanlash operatorlari

Shartli operator. Shartli operator ikki ko’rinishda ishlatilishi mumkin:

```
if (ifoda)
1- operator
else
2- operator
 yoki
if (ifoda)
1-operator
```

Shartli operator bajarilganda avval ifoda hisoblanadi; agar qiymat rost ya’ni noldan farqli bo’lsa 1- operator bajariladi. Agar qiymat yolg’on ya’ni nol bo’lsa va else ishlatilsa 2-operator bajariladi. Operator else qismi har doim eng yaqin if ga mos qo’yiladi.

```
if( n>0)
 if(a>b)
  Z = a;
 else
  Z = b;
```

Agar else qismni yuqori if ga mos qo’yish lozim bo’lsa, figurali qavslar ishlatish lozim.

```
if( n>0) {
 if(a>b)
  z = a;
}
else
  z = b;
```

Misol tariqasida uchta berilgan sonning eng kattasini aniqlash dasturi:

```
#include<stdio.h>
int main()
{
float a,b,c,max;
scanf("%f",&a);
scanf("%f",&b);
scanf("%f",&c);
if (a>b)
if (a>c) max = a; else max = c;
else
if (b>c) max = b; else max = c;
printf("\n max = %f", max);
return 0;
}
```

Keyingi misolda kiritilgan ball va maksimal ball asosida baho aniqlanadi:

```
#include<stdio.h>
int main()
```

```
{
int ball,max_ball,baho;
printf( "\n ball = ");
scanf("%d",&ball);
printf("\n max_ball = ");
scanf("%d",&max_ball);
float d = (float)ball/max_ball;
if (d>0.85) baho = 5; else
{
if (d>0.71) baho = 4; else
{
if (d>0.55) baho = 3; else baho = 2;
}
}
printf("\n baho = %d",baho);
return 0;
}
```

Kalit bo'yicha tanlash operatori. Kalit bo'yicha tanlash switch operatori umumiy ko'rinishi quyidagicha:

```
switch(<ifoda>) {
case <1-qiymat>:<1-operator>
     …
break;
     …
default: <operator>
     …
case: <n-operator>;
 }
```

Oldin qavs ichidagi butun ifoda hisoblanadi va uning qiymati hamma variantlar bilan solishtiriladi. Biror variantga qiymat mos kelsa shu variantda ko’rsatilgan operator bajariladi. Agar biror variant mos kelmasa default orqali ko’rsatilgan operator bajariladi. Uzish break operatori ishlatilmasa shartga mos kelgan variantdan tashqari keyingi variantdagi operatorlar ham avtomatik bajariladi. Quyidagi default, break va belgilangan variantlar ixtiyoriy tartibda kelishi mumkin. Umuman default yoki break operatorlarini ishlatish shart emas. Belgilangan operatorlar bo’sh bo’lishi ham mumkin.

Misol tariqasida bahoni son miqdoriga qarab aniqlash dasturini ko’ramiz.

```
#include <stdio.h>
int main()
{
int baho;
scanf("%d", &baho);
switch(baho)
{
case 2:printf("\n yomon");break;
case 3:printf("\n o'rta");break;
case 4:printf("\n yahshi");break;
case 5:printf("\n alo");break;
default: printf("\n noto'g'ri kiritilgan");
};
return 0;
}
```

Keyingi misolda kiritilgan simvol unli harf ekanligi aniqlanadi:

```
#include <stdio.h>
int main()
{
char c;
```

```
scanf("%c", &c);
switch(c)
{
case 'a':
case 'u':
case 'o':
case 'i':
printf("\n Simvol unli");break;
default: printf("\n Simvol unli emas");
};
return 0;
}
```

8,9-Ma’ruza. Takrorlanuvchi jarayonlar va ularni dasturlash. For, while va do operatorlari yordamida takrorlanishlarni dasturlash

Ma’ruza rejasi:

8.1 Sikl operatorlari

8.2 O’tish operatorlari

Kalit so’zlar: *delete, masofa keltirish, delete[], new, indeks, this, indeksirlash, [] bo’sh xotira, void*, konteyner, ro’yxat, manzil, nolinchi ko’rchsatkich, tugun, adres olish &, bo’shatish, ko’rsatkich, virtual destruktor, xotira, xotira chiqishi, destruktor, toifani o’zlashtirish, resurslar chiqishi, a’zo destruktori.*

8.1 Sikl operatorlari

Oldingi shartli while operatori. Oldingi shartli while operatori quyidagi umumiy ko’rinishga egadir:

```
while(ifoda)
Operator
```

Bu operator bajarilganda avval ifoda hisoblanadi. Agar uning qiymati 0 dan farqli bo’lsa operator bajariladi va ifoda qayta hisoblanadi. To ifoda qiymati 0 bo’lmaguncha sikl qaytariladi.

Agar dasturda while (1); satr qo’yilsa bu dastur hech qachon tugamaydi.

Misol. Berilgan n gacha sonlar yig’indisi.

```
#include <stdio.h>
void main()
{
long n,i = 1,s = 0;
scanf("%d",&n);
```

```
while (i< = n )
 s+ = i++;
printf("\n s = %d",s);
}
```

Bu dasturda s+ = i++ ifoda s = s+i; i = i+1 ifodalarga ekvivalentdir.

Quyidagi dastur to nuqta bosilmaguncha kiritilgan simvollar va qatorlar soni hisoblanadi:

```
#include <stdio.h>
int main()
{
int nc = 0,nl = 0;
char c;
while ((c = getchar())! = '.' )
{
++nc;
if (c = = '\n') ++nl;
};
printf("satrlar = %d simvollar = %d \n",nl,nc);
return 0;
}
```

Keyingi shartli do-while operatori. Keyingi shartli do-while operatori umumiy ko’rinishi quyidagicha:

```
do
Operator
while(ifoda)
```

Sikl operatorining bu ko’rinishida avval operator bajariladi so’ngra ifoda hisoblanadi. Agar uning qiymati 0 dan farqli bo’lsa operator yana bajariladi va hokazo. To ifoda qiymati 0 bo’lmaguncha sikl qaytariladi.

Misol. Berilgan n gacha sonlar yig’indisi.

```
#include <stdio.h>
int main()
{
long n,i = 1,s = 0;
scanf("%d",&n);
do
s+ = i++;
while (i< = n);
printf("\n s = %d",s);
return 0;
}
```

Bu dasturning kamchiligi shundan iboratki agar n qiymati 0 ga teng yoki manfiy bo’lsa ham, sikl tanasi bir marta bajariladi va s qiymati birga teng bo’ladi.

Parametrli for operatori. Parametrli for operatori umumiy ko’rinishi quyidagicha:

```
for( 1-ifoda;2- ifoda; 3-ifoda)
Operator
```

Bu operator quyidagi operatorga mosdir.

```
1-ifoda;
while(2-ifoda) {
operator
3-ifoda
```

```
}
```

Misol. Berilgan n gacha sonlar yig’indisi.

```
#include <stdio.h>
int main()
{
int n;
scanf("%d",&n);
int s = 0;
for(int i = 1;i< = n; i++) s+ = i;
printf("\n%d",s);
return 0;
}
```

Siklda bir nechta schyotchikni qo’llanilishi. Parametrli for siklining sintaksisi unda bir nechta o’zgaruvchi - schyotchikni qo’llanilishiga, siklni davom etishini murakkab shartlarini tekshirishga va sikl schyotchiklari ustida ketma-ket bir nechta operasiyani bajarilishiga imkon beradi.

Agarda bir nechta schyotchikka qiymat o’zlashtirilsa yoki ular o’rtasida bir nechta operasiya bajarilsa, bu ifodalar vergul bilan ajratilgan holda ketma – ket yoziladi.

for siklida bir nechta schyotchikni qo’llanilishi

```
#include <stdio.h>
#include<conio.h>
int main()
{
int i,j;
for (i = 0, j = 0; i<3; i++, j++)
printf("i:%d j:%d\n",i,j);
getch();
```

```
return 0;
}
```

Hatija:

i: 0 j: 0

i: 1 j: 1

i: 2 j: 2

8.2 O’tish operatorlari

Uzish break operatori. Ba’zi hollarda sikl bajarilishini ixtiyoriy joyda to’xtatishga to’g’ri keladi. Bu vazifani break operatori bajarishga imkon beradi. Bu operator darxol sikl bajarilishini to’xtatadi va boshqaruvni sikldan keyingi operatorlarga uzatadi.

Misol:

```
#include <stdio.h>
int main()
{
int n;
while(1)
{
scanf("%d",&n);
if(n == 1||n == 0) break;
}
printf("Sikl tugadi");
return 0;
}
```

Bu misolda while(1) operatori yordamida cheksiz sikl hosil qilinadi. Agar 1 yoki 0 soni kiritilsa sikl to’xtatiladi.

Qaytarish continue operatori. Sikl bajarilishiga ta’sir o’tkazishga imkon beradigan yana bir operator continue operatoridir. Bu operator sikl qadamini bajarilishini to’xtatib for va while da ko’rsatilgan shartli tekshirishga o’tkazadi.

Misol:

```
#include <stdio.h>
int main()
{
int n;
for(;;)
{
scanf("%d",&n);
if(n == 1||n == 0) continue;
break;
}
printf("Sikl tugadi");
return 0;
}
```

Bu misolda for(;;) operatori yordamida cheksiz sikl hosil qilinadi. Agar 1 yoki 0 sonlardan farqli son kiritilsa sikl to’xtatiladi.

O’tish operatori goto. O’tish operatorining ko’rinishi:

goto <identifikator>. Bu operator identifikator bilan belgilangan operatorga o’tish kerakligini ko’rsatadi.

Misol uchun goto A1;…;A1:y = 5;

Strukturali dasturlashda goto operatoridan foydalanmaslik maslahat beriladi. Lekin ba’zi hollarda o’tish operatoridan foydalanish dasturlashni osonlashtiradi.

Misol uchun bir necha sikldan birdan chiqish kerak bo’lib qolganda, to’g’ridan-to’g’ri break operatorini qo’llab bo’lmaydi, chunki u faqat eng ichki sikldan chiqishga imkon beradi.

```
#include <stdio.h>
int main()
{
int n = 16,s = 0;
int i,j;
for(i = 1;i<5;i++)
for(j = 1;j<5;j++)
{
if(i*j>n) goto A;
C++;
}
A:printf("Sikl tugadi s = %d",s);
return 0;
}
```

10-Ma’ruza. Umumiy takrorlanish algoritmlari va ichma-ich takrorlanishlar

Ma’ruza rejasi:

10.1 C++tilida takrorlanuvchi jarayonlarni dasturlash

10.2 Continue operatori

Kalit so’zlar: *delete, masofa keltirish, delete[], new, indeks, this, indeksirlash, [] bo’sh xotira, void*, konteyner, ro’yxat, manzil, nolinchi ko’rchsatkich, tugun, adres olish &, bo’shatish, ko’rsatkich, virtual destruktor, xotira, xotira chiqishi, destruktor, toifani o’zlashtirish, resurslar chiqishi, a’zo destruktori.*

10.1 C++tilida takrorlanuvchi jarayonlarni dasturlash

Agar dastur bajarilish jarayonida operator yoki operatorlar guruhi bir necha marta qayta-qayta bajarilsa, bunday jarayonlarni takrorlanuvchi (siklik) jarayon deyiladi. C++ tilida siklni 3 xil ko’rinishda tashkil qilish mumkin.

1. Sharti avval tekshiriladigan takrorlanish (oldshartli sikl):

```
while (shart) operator (lar);
```

Bu yerda operatorlar while da ko’rsatilgan shart yolg’on bo’lgunicha takrorlanadi. Takrorlanish tanasi murakkab bo’lsa, ya’ni 1 tadan ortiq operatorlar qatnashsa, ularni alohida {} ichiga olish kerak bo’ladi.

```
Masalan: b = 2*(a+5); a € [1, 10]; h=1;
# include <iostream.h>
# include <math.h>
void main ( )
{  int a=1, b;
 while (a<=10)
 {  b = 2*(a+5); cout << “b=” <<b;
```

```
 cout << "a=" <<a << endl;
 a++ ; }
}
```

Ekranda 10 ta a va b larning qiymatlari paydo bo'ladi.

2- misol.

```
# include <iostream.h>
 void main ( )
 {  int i = 10;
    while ( i++ < =15)
    cout << "Salom!!!"<< endl;  }
```

Ekranda 5 marta "Salom!!!" yozuvi paydo bo'ladi.

2. Sharti keyin tekshiriladigan takrorlanish (so'ngshartli sikl):

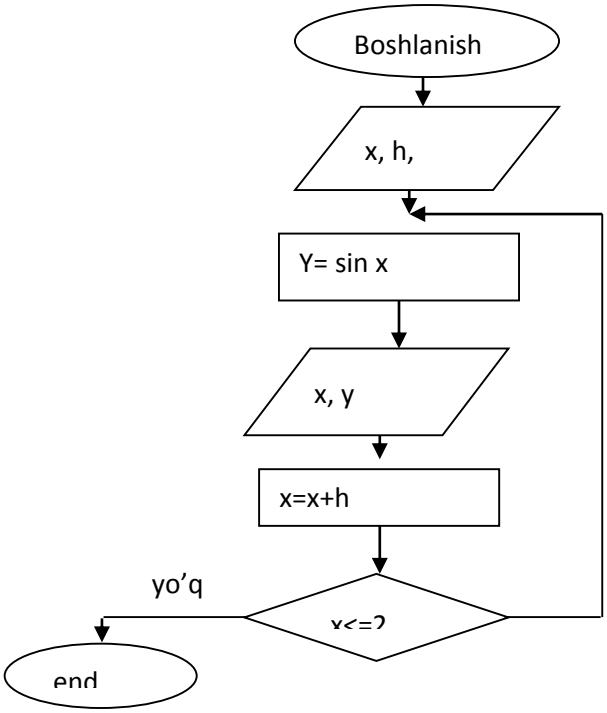
```
do
operator (lar)
while (shart);
```

Takrorlanish while da ko'rsatilgan shart yolg'on bo'lgunicha davom etadi.

Masalan: y=sinx; x € [1,2] ; h=0.1

```
# include <iostream.h>
# include <math.h>
 void main ( )
{ float x=1, y;
 do
  { y=sin(x); cout << "x="<<x<<"  y="<<y<<endl;
x+ = 0.1;  }
   while (x<=2); getch(); }
```

Masalaning algoritmi quyidagi ko'rinishga ega bo'ladi:



2- misol. Dastur klaviaturadan 20 soni kiritilishini kutadi.

```
# include <iostream>
void main ( )
{ int n;
  do
  { cin >> n;
  cout << "Sonni qayta kiriting!="<<n; }
  while (n!=20);
 }
```

3. Parametrli takrorlanish (sikl):

Umumiy ko'rinishi

```
for (bosh qiymat; shart; o'zgarish qadami)
operator (lar);
```

Operatorlar 1 tadan ortiq bo'lsa ularni alohida qavslar -{} ichiga olinadi.

1-misol. Y = cos x; x € [2,3] ; h=0,2;

```
# include <iostream.h>
# include <math.h>
void main ( )
{ float x, y;
  for (x=2; x<=3; x+ = 0.2)
  { y=cos(x);  cout << "x="<< x << "  y="<< y << endl; }
}
```

2-misol. 100 gacha bo'lgan juft sonlarni ekranga chiqarish dasturi.

```
# include <iostream.h>
  void main ( )
 {   int i = 2;
  while (i<=100)
  { cout << "i="<< i;  i += 2; }  }
```

….
for (int i=2; i<=100; i +=2)
cout << "i="<< i;
	do
	cout << "i="<< i; i += 2;
	while (i<=100);

3-misol: 1 dan 100 gacha bo'lgan 3 raqami bilan tugaydigan sonlarni ekranga chiqarish dasturini tuzing (2 xil usulda).

……
int i=3;
while (i <=100)
	…..
	for (i = 3;i <= 100; i + = 10)
	cout << "i="<< i;

```
    { cout << "i="<<
i;
       i + = 10; }
```

4-misol. Qadimiy masala. Bir odam 100 so'm bilan bozorga bordi. Bozorda 1 ta sigir 10 so'm, 1 ta qo'y 3 so'm, 1 ta echki 0.5 so'm va xarid qilingan qoramollarning umumiy soni 100 ta bo'lsa, nechta sigir, qo'y va echki sotib olindi?

Sigirlar soni: x, qo'ylar soni y, echkilar soni z deb olinsa,

```
       # include
<iostream.h>
    int main ( )
    {
    int x, y, z, s;
    for (x=1; x<=100;
x++)
    for (y=1; y<=100;
y++)
    if (19*x + 5*y = =
100)
    { z = 100 - x - y;
      cout << "x="<<x;
      cout << "y="<<y;
      cout << "z="<<z; }
       return 0;
       }
```

```
    # include
<iostream.h>
    int main ( )
    {
    int x, y, z, s;
    for (x=1; x<=100;
x++)
    for (y=1; y<=100;
y++)
    for (z=1; z<=100;
z++)
    if (x + y + z = =
100)
    {  cout << "x="<<x;
      cout << "y="<<y;
      cout << "z="<<z; }
       return 0;
       }
```

10.2 Continue operatori

Bu operator yordamida sikl parametrining biror qiymatida hisoblashni to’xtatib, keyingi qiymatida hisoblashni davom ettirish mumkin. Masalan: y = 2x funksiyasini x € [1,18] oraliqda h=1 qiymat bilan hisoblash kerak, lekin x=6 yoki x=13 qiymatlarida hisoblashni bajarmaslik kerak.

```
# include <iostream.h>
void main ( )
{ int x, y;
for (x=1; x<=18; x++)
 { if (( x = = 6) || (x = = 13)) continue;
  y = 2*x; cout << “x=”<< x << “  y=” << y << endl;
 }  }
```

2 - misol. 10 ta ketma-ket kiritiladigan butun musbat sonlar yig’indisini hisoblash dasturini tuzing. Agar son manfiy bo’lsa, yig’indiga qo’shmaslik kerak.

```
# include <iostream.h>
void main ( )
{   int x, i, s=0;
   for ( i =1; i <=10; i ++)
   { cin >> x;
   if ( x < 0)  continue;
   s = s + x; }          // s +=x deb yozsa ham
bo’ladi.
   cout << “s=”<< s << endl;    }
```

3-misol. $Y = x^n$ funksiyasini rekurrent formula orqali hisoblash dasturini tuzing. Bu yerda n - butun son, x – ixtiyoriy haqiqiy son.

```
# include <iostream.h>
# include <conio.h>
void main ( )
{ float x=2.56, y=1;
```

```
for ( int n=1; n<=10; n++)
y = y * x;                    // y * = x;
cout <<"y="<<y<<endl;
getch ( );  }
```

4-misol. $Y=\sum_{n=1}^{8}\frac{x^2}{n!} = x^2 + \frac{x^2}{1*2} + \frac{x^2}{1*2*3} + ... + \frac{x^2}{1*2*3*4*5*6*7*8}$

x - ixtiyoriy haqiqiy son.

```
# include <iostream.h>
# include <conio.h>
void main ( )
{ float x=3.75, y=0; long p=1;
for ( int n=1; n<=8; n++)
{ p = p * n; y = y + x*x / p; }
cout << "y="<<y<<endl;
getch ( );
}
```

5-misol. $S=\cos x + \frac{\cos 2x}{2} + \frac{\cos 3x}{3} + ... + \frac{\cos nx}{n}$;

bu yerda $\frac{\pi}{5} \le x \le \frac{9\pi}{5}$ n=10

```
# include <iostream.h>
# include <conio.h>
# include <math.h>
    void main ( )
{ float a, b, h, x, s, pi=3.14;
a = pi / 5;  b=9 * pi / 5; h=(b-a) / 10;
x = a; cout. precision (3);
while ( x<=b )
```

```
{ s = 0;
for ( int n=1; n<=10; n++)
s = s + cos(n*x) / n;
cout <<"s="<<s<<endl;
x = x + h;
}
getch ( ); }
```

6-misol. Boy bilan olim bahslashibdilar. Olim boyga har kuni (30 kun) 100000 so'm beradigan bo'libdi. Boy esa olimga 1-kun 1 tiyin, 2-kun 2 tiyin, 3-kun 4 tiyin, 4-kun 8 tiyin va h.k. pul beradigan bo'libdi. Bahsda kim yutadi? Dasturini tuzing.

Olim → 30*100000 = 3000000 so'm

Boy → $\sum_{i=0}^{30} 2^i$ sum → 10737418 so'm

```
# include <iostream.h>
# include <conio.h>
# include <math.h>
void main ( )
{ int s, s1=1, k = 0;
for ( int i=1; i<=30; i++)
{ k = k + s1;
s1 = s1*2; }
k = k / 100;
s = 30*100000;
cout <<"boy olimga beradi:"<<k<<endl;
cout <<"olim boyga beradi:" << s << endl;
getch ( );
}
```

Nazorat savollari:

1. Sharti avval tekshiriladigan takrorlanish
2. Sharti keyin tekshiriladigan takrorlanish
3. Parametrli takrorlanish
4. Dasturda takrorlanishlarni tashkil etish.
5. Takrorlanuvchi dastur nima?
6. Murakkab takrorlanishlar
7. **continue** operatori
8. **return** operatori

11,12-Ma’ruza. C++ da funksiyalar: tuzilishi va undan foydalanish. . Qiymat qaytarmaydigan funksiyalar va ular yordamida masala yechish

Ma’ruza rejasi:

11.1 Funksiya tushunchasi

11.2 Funksiya parametrlari

11.3 Funksiyadan foydalanish

Kalit so’zlar:*, ro’yxat, manzil, nolinchi ko’rchsatkich, tugun, adres olish &, bo’shatish, ko’rsatkich, virtual destruktor, xotira, xotira chiqishi, destruktor, toifani o’zlashtirish, resurslar chiqishi, a’zo destruktori.*

Programma ta’minotini yaratish amalda murakkab jarayon hisoblanadi. Programma tuzuvchi programma kompleksini bir butun-likdagi va uning har bir bo‘lagining ichki mazmunini va ularning sezilmas farqlarini hisobga olishi kerak bo‘ladi.

Programmalashga tizimli yondoshuv shundan iboratki, program-ma tuzuvchi oldiga qo‘yilgan masala oldindan ikkita, uchta va undan ortiq nisbatan kichik masala ostilarga bo‘linadi. O‘z navbatida bu masala ostilari ham yana kichik masala ostilariga bo‘linishi mumkin. Bu jarayon toki mayda masalalarni oddiy standart amallar yordamida echish mumkin bo‘lguncha davom etadi. SHu yo‘l bilan masalani dekompozitsiyalash amalga oshiriladi.

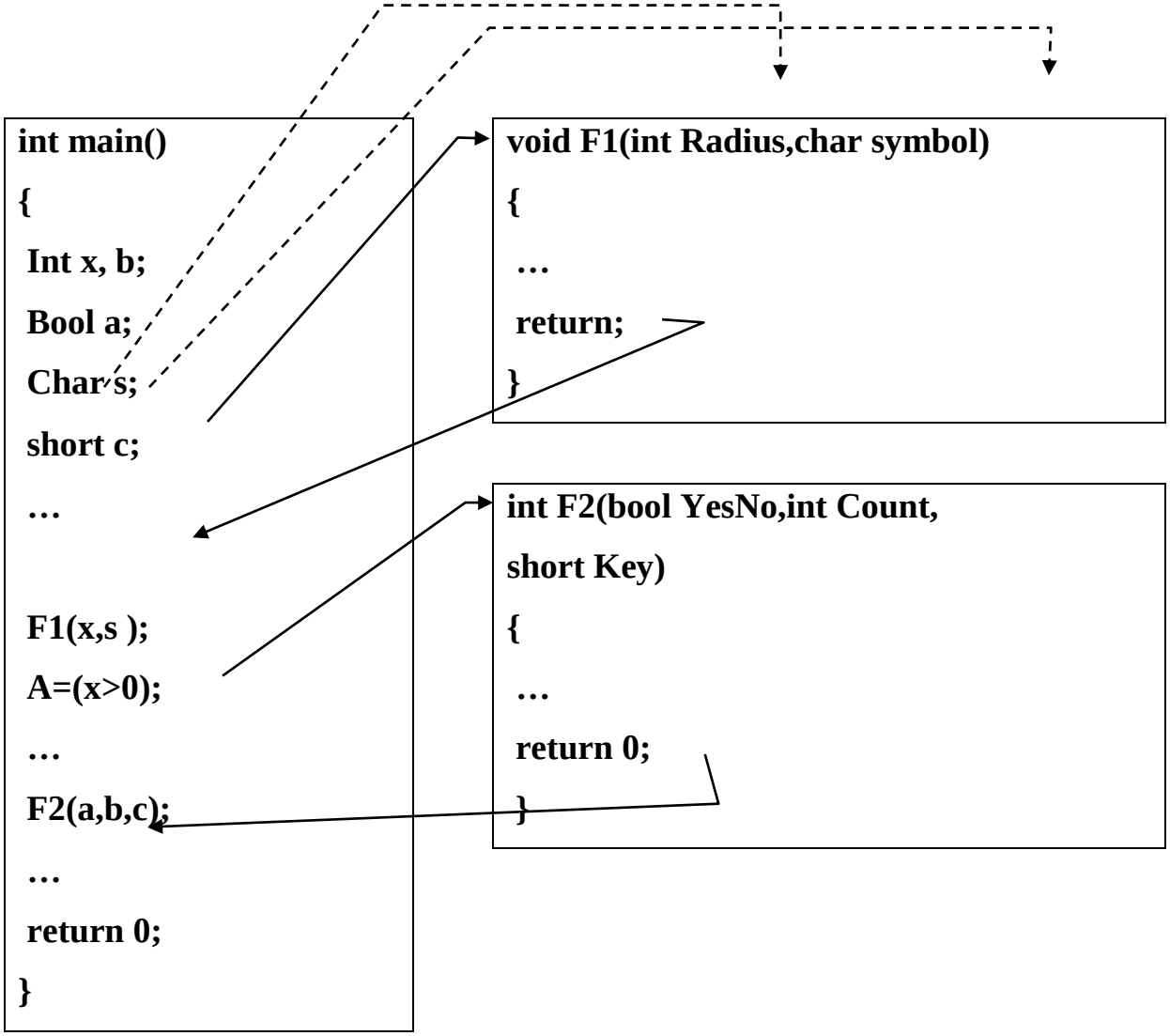
Ikkinchi tomondan, programmalashda shunday holatlar kuzatila-diki, unda programmaning turli joylarida mazmunan bir xil algo-ritmlarni bajarishga to‘g‘ri keladi. Algoritmning bu bo‘laklari asosiy echilayotgan masaladan ajratib olingan qandaydir masala ostini echishga mo‘ljallangan bo‘lib, etarlicha mustaqil qiymatga (natijaga) egadir. Misol uchun quyidagi masalani ko‘raylik:

Berilgan $a_0,a_1,...,a_{30}$, $b_0,b_1,...,b_{30}$, $c_0,c_1,...,c_{30}$ va x,y,z haqiqiy sonlar uchun

$$\frac{\left(a_0x^{30}+a_1x^{29}+\ldots+a_{30}\right)^2-\left(b_0y^{30}+b_1y^{29}+\ldots+b_{30}\right)}{c_0(x+z)^{30}+c_1(x+z)^{29}+\ldots+c_{30}}$$

ifodaning qiymati hisoblansin.

Funksiya tanasidagi return operatori yoki oxirgi operator bajargandan keyin avtomatik ravishda bosh funksiyaga qaytish amalga oshiriladi.



11.1-rasm. Bosh funksiyadan boshqa funksiyalarni chaqirish va qaytish

Aksariyat hollarda main() funksiyasining parametrlar ro‘yxati bo‘sh bo‘ladi. Agar yuklanuvchi programmani ishga tushirishda,buyruq satri orqali yuklanuvchi programma ishga tushirilganda, unga parametrlarni uzatish (berish) zarur bo‘lsa, main() programmasi funksiyasining sintaksisi o‘zgaradi:

```
int main(int argc, char* argv[]);
```

Bu erda argc - uzatiladigan parametrlar soni, argv[]- bir-biridan punktuatsiya belgilari (va probel) bilan ajratilgan parametrlar ro‘yxatini o‘z ichiga olgan massivga ko‘rsatkich.

Quyida funksiyalarni e’lon qilish, chaqirish va aniqlashga misollar keltirilgan:

```
// funksiyalar e’loni
int Mening_funksiyam(int Number, float Point);
char Belgini_uqish();
void bitni_urnatish(short Num);
void Amal_yoq(int,char);
// funksiyalarni chaqirish
result=Mening_funksiyam(Varb1,3.14);
symb=Belgini_uqish();
bitni_urnatish(3);
Amal_yoq(2,Smbl);
// funksiyalarni aniqlash
int Mening_funksiyam(int Number,float Point);
{ int x;
  ...
 return x;}
char Belgini_uqish()
{
 char Symbol;
 cin>>Symbol;
 return Symbol;
};
void bitni_urnatish(short number)
{
 global_bit=global_bit | number;
};
```

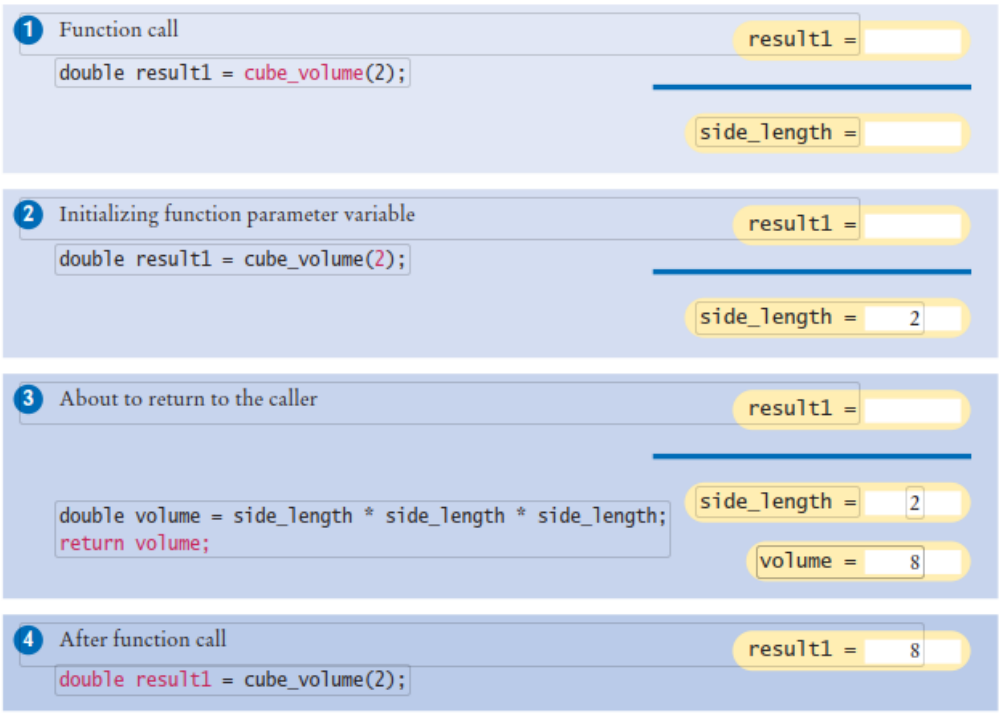
```
void Amal_yoq(int x, char ch){};
```

Misollar

ch05/cube.cpp

```
1 #include <iostream>
2
3 using namespace std;
4
5 /**
6 Computes the volume of a cube.
7 @param side_length the side length of the cube
8 @return the volume
9 */
10 double cube_volume(double side_length)
11 {
12 double volume = side_length * side_length *
side_length;
13 return volume;
14 }
15
16 int main()
17 {
18 double result1 = cube_volume(2);
19 double result2 = cube_volume(10);
20 cout << "A cube with side length 2 has volume " <<
result1 << endl;
21 cout << "A cube with side length 10 has volume " <<
result2 << endl;
```

```
22
23 return 0;
24 }
```



ch05/intname.cpp

```
1 #include <iostream>
2 #include <string>
3
4 using namespace std;
5
6 /**
7 Turns a digit into its English name.
8 @param digit an integer between 1 and 9
9 @return the name of digit ("one" ... "nine")
10 */
11 string digit_name(int digit)
12 {
13 if (digit == 1) return "one"; 14 if (digit == 2) return "two"; 15 if (digit == 3) return "three";
16 if (digit == 4) return "four"; 17 if (digit == 5) return "five"; 18 if (digit == 6) return "six";
19 if (digit == 7) return "seven"; 20 if (digit == 8) return
```

```
"eight"; 21 if (digit == 9)
return "nine";
    22 return "";
    23 }
    24
    25 /**
    26 Turns a number
between 10 and 19 into
its English name.
    27 @param number an
integer between 10 and 19
    28 @return the name
of the given number
("ten" ... "nineteen")
    29 */
    30 string teen_name(int
number)
    31 {
    32 if (number == 10)
return "ten";
    33 if (number == 11)
return "eleven";
    34 if (number == 12)
return "twelve";
    35 if (number == 13)
return "thirteen";
    36 if (number == 14)
return "fourteen";
    37 if (number == 15)
return "fifteen";
    38 if (number == 16)
return "sixteen";
    39 if (number == 17)
return "seventeen";
    40 if (number == 18)
return "eighteen";
    41 if (number == 19)
return "nineteen";
    42 return "";
    43 }
    44
    45 /**
    46 Gives the name of
the tens part of a number
between 20 and 99.
    47 @param number an
integer between 20 and 99
    48 @return the name
of the tens part of the
number ("twenty" ...
"ninety") 49 */
    50 string tens_name(int
number)
    51 {
    52 if (number >= 90)
return "ninety";
```

```
53 if (number >= 80) return "eighty";
54 if (number >= 70) return "seventy";
55 if (number >= 60) return "sixty";
56 if (number >= 50) return "fifty";
57 if (number >= 40) return "forty";
58 if (number >= 30) return "thirty";
59 if (number >= 20) return "twenty";
60 return "";
61 }
62
63 /**
64 Turns a number into its English name.
65 @param number a positive integer < 1,000
66 @return the name of the number (e.g. "two hundred seventy four")
67 */
68 string int_name(int number)
69 {
70 int part = number; // The part that still needs to be converted
71 string name; // The return value
72
73 if (part >= 100)
74 {
75 name = digit_name(part / 100) + " hundred"; 76 part = part % 100;
77 }
78
79 if (part >= 20)
80 {
81 name = name + " " + tens_name(part);
82 part = part % 10;
83 }
84 else if (part >= 10)
85 {
86 name = name + " " + teen_name(part);
87 part = 0;
88 }
89
90 if (part > 0)
```

```
91 {
92  name = name + " " +
digit_name(part);
93 }
94
95 return name;
96 }
97
98 int main()
99 {
100 cout << "Please
enter a positive integer: ";
101 int input;
102 cin >> input;
103 cout <<
int_name(input) << endl;
104 return 0;
105 }
```

program run

Please enter a positive integer: 729

seven hundred twenty nine

Nazorat savollari

1. C++da funksiya qanday ishlaydi?
2. funksiyaga kutubxona kerakmi?
3. For operatori funksiyada qanday ishlatiladi?
4. Matematik funksiyalar qanday ishlaydi?
5. Funksiya parametrlar nima?
6. Funksiya qanday chaqiriladi?
7. Funksiya parametrlari orqali nima uzatiladi?
8. If operatorining nechta turi bor?
9. O’zgaruvchilar nima uchun qo’llaniladi?

13-Ma’ruza. Funksiyalarda argument sifatida local, global o’zgaruvchilardan va havolalardan foydalanish.

Ma’ruza rejasi:

13.1 Lokal o’zgaruvchilar

13.2 Global o’zgaruvchilar

13.3 Havolalar tushunchasi

Kalit so’zlar:*, ro’yxat, manzil, nolinchi ko’rchsatkich, tugun, adres olish &, bo’shatish, ko’rsatkich, virtual destruktor, xotira, xotira chiqishi, destruktor, toifani o’zlashtirish, resurslar chiqishi, a’zo destruktori.*

Ko‘rinish sohasi. Lokal va global o‘zgaruvchilar

O‘zgaruvchilar funksiya tanasida yoki undan tashqarida e’lon qilinishi mumkin. Funksiya ichida e’lon qilingan o‘zgaruvchilarga *lokal o‘zgaruvchilar* deyiladi. Bunday o‘zgaruvchilar xotiradagi prog-ramma stekida joylashadi va faqat o‘zi e’lon qilingan funksiya tanasida amal qiladi. Boshqaruv asosiy funksiyaga qaytishi bilan lokal o‘zgaruvchilar uchun ajratilgan xotira bo‘shatiladi (o‘chiriladi).

Har bir o‘zgaruvchi o‘zining amal qilish sohasi va yashash vaqti xususiyatlari bilan xarakterlanadi.

O‘zgaruvchi *amal qilish sohasi* deganda o‘zgaruvchini ishlatish mumkin bo‘lgan programma sohasi (qismi) tushuniladi. Bu tushuncha bilan o‘zgaruvchining *ko‘rinish sohasi* uzviy bog‘langan. O‘zgaruvchi amal qilish sohasidan chiqqanda ko‘rinmay qoladi. Ikkinchi tomondan, o‘zgaruvchi amal qilish sohasida bo‘lishi, lekin ko‘rinmas-ligi mumkin. Bunda ko‘rinish sohasiga ruxsat berish amali «::» yordamida ko‘rinmas o‘zgaruvchiga murojat qilish mumkin bo‘ladi.

O‘zgaruvchining *yashash vaqti* deb, u mavjud bo‘lgan programma bo‘lagining bajarilishiga ketgan vaqt intervaliga aytiladi.

Lokal o‘zgaruvchilar o‘zlari e’lon qilingan funksiya yoki blok chegarasida ko‘rinish sohasiga ega. Blokdagi ichki bloklarda xuddi shu nomdagi o‘zgaruvchi e’lon qilingan bo‘lsa, ichki bloklarda bu lokal o‘zgaruvchi ham amal qilmay qoladi. Lokal o‘zgaruvchi yashash vaqti - blok yoki funksiyani bajarish vaqti bilan aniqlanadi. Bu hol shuni anglatadiki, turli funksiyalarda bir-biriga umuman bog‘liq bo‘lma-gan bir xil nomdagi lokal o‘zgaruvchilarni ishlatish mumkin.

Quyidagi programmada main() va sum() funksiyalarida bir xil nomdagi o‘zgaruvchilarni ishlatish ko‘rsatilgan. Programmada ikkita sonning yig‘indisi hisoblanadi va chop etiladi:

```
#include <iostream.h>
// funksiya prototipi
int sum(int a;int b);
int main()
{
 // lokal o‘zgaruvchilar
 int x=r;
 int y=4;
 cout<<sum(x, y);
 return 0;
}
int sum(int a,int b)
{
 // lokal o‘zgaruvchi
 int x=a+b;
 return x;
}
```

Global o‘zgaruvchilar programma matnida funksiya aniqlanishi-dan tashqarida e’lon qilinadi va e’lon qilingan joyidan boshlab programma oxirigacha amal qiladi.

```
#include <iostream.h>
int f1(); int f2();
int main()
{
 cout<<f1()<<" "<<f2()<<endl;
 return 0;
}
int f1()
{
 return x;// kompilyasiya xatosi ro‘y beradi
}
int x=10; // global o‘zgaruvchi e’loni
int f2(){ return x*x;}
```

YUqorida keltirilgan programmada kompilyasiya xatosi ro‘y beradi, chunki f1() funksiya uchun x o‘zgaruvchisi noma’lum hisob-lanadi.

Programma matnida global o‘zgaruvchilarni ular e’lonidan keyin yozilgan ixtiyoriy funksiyada ishlatish mumkin. SHu sababli, global o‘zgaruvchilar programma matnining boshida yoziladi. Funksiya ichidan global o‘zgaruvchiga murojat qilish uchun funksiyada uning nomi bilan mos tushadigan lokal o‘zgaruvchilar bo‘lmasligi kerak. Agar global o‘zgaruvchi e’lonida unga boshlang‘ich qiymat berilmagan bo‘lsa, ularning qiymati 0 hisoblanadi. Global o‘zgaruvchining amal qilish sohasi uning ko‘rinish sohasi bilan ustma-ust tushadi.

SHuni qayd etish kerakki, tajribali programma tuzuvchilar imkon qadar global o‘zgaruvchilarni ishlatmaslikka harakat qilishadi, chunki bunday o‘zgaruvchilar qiymatini programmaning ixtiyoriy joyidan o‘zgartirish xavfi mavjudligi sababli programma ishlashida mazmunan xatolar yuzaga kelishi mumkin. Bu fikrimizni tasdiqlovchi programmani ko‘raylik.

```
# include <iostream.h>
// global o‘zgaruvchi e’loni
```

```
int test=100;
void Chop_qilish(void );
int main()
{
 //lokal o‘zgaruvchi e’loni
 int test=10;
 //global o‘zgaruvchi chop qilish funksiyasini
chaqirish
 Chop_qilish();
 sout<<”Lokal o’zgaruvchi: ”<<test<<‘\n’;
 return 0;
}
void Chop_qilish(void)
{
 cout<<”Global o’zgaruvchi: ”<<test<<‘\n’;
}
```

Programma boshida test global o‘zgaruvchisi 100 qiymati bilan e’lon qilinadi. Keyinchalik, main() funksiyasida test nomi bilan lokal o‘zgaruvchisi 10 qiymati bilan e’lon qilinadi. Programmada, Chop_qilish() funksiyasiga murojaat qilinganida, asosiy funksiya tanasidan vaqtincha chiqiladi va natijada main() funksiyasida e’lon qilingan barcha lokal o‘zgaruvchilarga murojaat qilish mumkin bo‘lmay qoladi. SHu sababli Chop_qilish() funksiyasida global test o‘zgaruvchisining qiymatini chop etiladi. Asosiy programmaga qaytilgandan keyin, main() funksiyasidagi lokal test o‘zgaruvchisi global test o‘zgaruvchisini «berkitadi» va lokal test o‘zgaruvchini qiymati chop etiladi. Programma ishlashi natijasida ekranga quyidagi natijalar chop etiladi:

Global o‘zgaruvchi: 100

Lokal o‘zgaruvchi: 10

:: amali

YUqorida qayd qilingandek, lokal o‘zgaruvchi e’loni xuddi shu nomdagi global o‘zgaruvchini «berkitadi» va bu joydan global o‘zgaruvchiga murojat qilish imkoni bo‘lmay qoladi. S++ tilida bunday holatlarda ham global o‘zgaruvchiga murojat qilish imko-niyati saqlanib qolingan. Buning uchun «ko‘rinish sohasiga ruxsat berish» amalidan foydalanish mumkin va o‘zgaruvchi oldiga ikkita nuqta - «::» qo‘yish zarur bo‘ladi. Misol tariqasida quyidagi programani keltiramiz:

```
#include <iostream.h >
//global o‘zgaruvchi e’loni
int uzg=5;
int main()
{
//lokal o‘zgaruvchi e’loni
 int uzg=70;
 //lokal o‘zgaruvchini chop etish
 cout<<uzg<<‘/n’;
 //global o‘zgaruvchini chop etish
 cout<<::uzg <<’/n’;
 return 0;
}
```

Programma ishlashi natijasida ekranga oldin 70 va keyin 5 sonlari chop etiladi.

Xotira sinflari

O‘zgaruvchilarning ko‘rinish sohasi va amal qilish vaqtini aniqlovchi o‘zgaruvchi modifikatorlari mavjud (5.1-jadval).

5.1-jadval. O‘zgaruvchi modifikatorlari

Modifikator	**Qo‘llanishi**	**Amal qilish sohasi**	**YAshash davri**

auto	lokal	blok	vaqtincha
register	lokal	blok	vaqtincha
exteru	global	blok	vaqtincha
static	lokal	blok	doimiy
	global	fayl	doimiy
volatile	global	fayl	doimiy

Avtomat o‘zgaruvchilar. auto modifikatori lokal o‘zgaruvchilar e’lonida ishlatiladi. Odatda lokal o‘zgaruvchilar e’lonida bu modifikator kelishuv bo‘yicha qo‘llaniladi va shu sababli amalda uni yozishmaydi:

```
#include <iostream.h>
int main()
{
 auto int X=2; // int X=2; bilan ekvivalent
 cout<<X;
 returu 0;
}
```

auto modifikatori blok ichida e’lon qilingan lokal o‘zgaruvchi-larga qo‘llaniladi. Bu o‘zgaruvchilar blokdan chiqishi bilan avtoma-tik ravishda yo‘q bo‘lib ketadi.

Registr o‘zgaruvchilar. register modifikatori kompilyatorga, ko‘rsatilgan o‘zgaruvchini protsessor registrlariga joylashtirishga harakat qilishni tayinlaydi. Agar bu harakat natija bermasa o‘zga-ruvchi auto turidagi lokal o‘zgaruvchi sifatida amal qiladi.

O‘zgaruvchilarni registrlarda joylashtirish programma kodini bajarish tezligi bo‘yicha optimallashtiradi, chunki protsessor xotiradagi berilganlarga nisbatan registrdagi qiymatlar bilan ancha tez ishlaydi. Lekin registrlar soni cheklanganligi uchun har doim ham o‘zgaruvchilarni registrlarda joylashtirishning iloji bo‘lmaydi.

```
#include < iostream.h >
int main()
```

```
{
 register int Reg;
 ...
 return 0;
}
```

register modifikatori faqat lokal o‘zgaruvchilariga nisbatan qo‘llaniladi, global o‘zgaruvchilarga qo‘llash kompilyasiya xatosiga olib keladi.

Tashqi o‘zgaruvchilar. Agar programma bir nechta moduldan iborat bo‘lsa, ular qandaydir o‘zgaruvchi orqali o‘zaro qiymat alma-shishlari mumkin (fayllar orasida). Buning uchun o‘zgaruvchi birorta modulda global tarzda e’lon qilinadi va u boshqa faylda (modulda) ko‘rinishi uchun u erda extern modifikatori bilan e’lon qilinishi kerak bo‘ladi. extern modifikatori o‘zgaruvchini boshqa faylda e’lon qilinganligini bildiradi. Tashqi o‘zgaruvchilar ishlatilgan prog-rammani ko‘raylik.

```
//Sarlavha.h faylida
void Bayroq_Almashsin(void);
// modul_1.cpp faylida
bool Bayroq;
void Bayroq_Almashsin(void){Bayroq=!Bayroq;}
// masala.cpp faylida
#include < iostream.h>
#include <Sarlavha.h>
#include <modul_1.cpp>
extern bool Bayroq;
int main()
{
 Bayroq_Almashsin();
 if(Bayroq)
 cout<<”Bayroq TRUE”<<endl;
 else cout<<”Bayroq FALSE”<<endl;
```

```
 return 0;
}
```

Oldin sarlavha.h faylida Bayroq_Almashsin( ) funksiya sarlav-hasi e’lon qilinadi, keyin modul_1.srr faylida tashqi o‘zgaruvchi e’lon qilinadi va Bayroq_Almashsin() funksiyasining tanasi aniqla-nadi va nihoyat, masala.cpp faylida Bayroq o‘zgaruvchisi tashqi deb e’lon qilinadi.

Statik o‘zgaruvchilar. Statik o‘zgaruvchilar static modifika-tori bilan e’lon qilinadi va o‘z xususiyatiga ko‘ra global o‘zgaruvchi-larga o‘xshaydi. Agar bu turdagi o‘zgaruvchi global bo‘lsa, uning amal qilish sohasi - e’lon qilingan joydan programma matnining oxirigacha bo‘ladi. Agar statik o‘zgaruvchi funksiya yoki blok ichida e’lon qilinadigan bo‘lsa, u funksiya yoki blokka birinchi kirishda initsializatsiya qilinadi. O‘zgaruvchining bu qiymati funksiya keyingi chaqirilganida yoki blokka qayta kirishda saqlanib qoladi va bu qiymatni o‘zgartirish mumkin. Statik o‘zgaruvchilarni tashqi deb e’lon qilib bo‘lmaydi.

Agar statik o‘zgaruvchi initsializatsiya qilinmagan bo‘lsa, uning birinchi murojatdagi qiymati 0 hisoblanadi.

Misol tariqasida birorta funksiyani necha marotaba chaqiril-ganligini aniqlash masalasini ko‘raylik:

```
#include <iostream.h >
int Sanagich(void);
int main()
{
 int natija;
 for (int i=0; i<30; i++)
 natija=Sanagich();
 cout<<natija;
 return 0;
}
int Sanagich(void)
```

```
{
 static short sanagich=0;
 ...
 sanagich++;
 return sanagich;
}
```

Bu erda asosiy funksiyadan counter statik o‘zgaruvchiga ega Sanagicht() funksiyasi 30 marta chaqiriladi. Funksiya birinchi marta chaqirilganda sanagich o‘zgaruvchiga 0 qiymatini qabul qiladi va uning qiymati birga ortgan holda funksiya qiymati sifatida qaytariladi. Statik o‘zgaruvchilar qiymatlarini funksiyani bir chaqirilishidan ikkinchisiga saqlanib qolinishi sababli, keyingi har bir chaqirishlarda sanagich qiymati bittaga ortib boradi.

Masala. Berilgan ishorasiz butun sonning barcha tub bo‘luv-chilari aniqlansin. Masalani echish algoritmi quyidagi takrorla-nuvchi jarayondan iborat bo‘ladi: berilgan son tub songa (1-qadamda 2 ga) bo‘linadi. Agar qoldiq 0 bo‘lsa, tub son chop qilinadi va bo‘linuv-chi sifatida bo‘linma olinadi, aks holda navbatdagi tub son olinadi. Takrorlash navbatdagi tub son bo‘linuvchiga teng bo‘lguncha davom etadi.

Programma matni:

```
#include<iostream.h>
#include<math.h>
int Navb_tub();
int main()
{
 unsigned int n,p;
 cout<<”\nn qiymatini kiritng: ”;
 cin>>n;
 cout<<"\n1";
 p=Navb_tub();
```

```
 while(n>=p)
 {
  if(n%p==0)
  {
   cout<<"*"<<p;
   n=n/p;
  }
  else p=Navb_tub();
  }
 return 0;
}
int Navb_tub()
{
 static unsigned int tub=1;
 for(;;)
 {
  tub++;
  short int ha_tub=1;
  for(int i=2;i<=tub/2;i++)
   if(tub%i==0)ha_tub=0;
  if(ha_tub)return tub;
 }
 return 0;
}
```

Programmada navbatdagi tub sonni hosil qilish funksiya ko‘ri-nishida amalga oshirilgan. Navb_tub() funksiyasining har chaqirili-shida oxirgi tub sondan keyingi tub son topiladi. Oxirgi tub sonni «eslab» qolish uchun tub o‘zgaruvchisi static qilib aniqlangan.

Programma ishga tushganda klaviaturadan n o‘zgaruvchisining qiymati sifatida 60 soni kiritilsa, ekranga quyidagi ko‘paytma chop etiladi:

1*2*2*3*5

volatile sinfi o‘zgaruvchilari. Agar programmada o‘zgaruvchini birorta tashqi qurilma yoki boshqa programma bilan bog‘lash uchun ishlatish zarur bo‘ladigan bo‘lsa, u volatile modifikatori bilan e’lon qilinadi. Kompilyator bunday modifikatorli o‘zgaruvchini registrga joylashtirishga harakat qilmaydi. Bunday o‘zgaruvchilar e’loniga misol quyida keltirilgan:

```
volatile short port_1;
volatile const int Adress=0x00A2;
```

Misoldan ko‘rinib turibdiki, volatile modifikatorli o‘zgarmas ham e’lon qilinishi mumkin.

onsider this sample program:

```
1   int x;
2   int mystery(int x)
3   {
4    int s = 0;
5    for (int i = 0; i < x; i++)
6    {
7    int x = i + 1;
8    s = s + x;
9    }
10    return x;
11   }
12   int main()
13   {
14    x = 4;
15    int s = mystery(x);
16    cout << s << endl;
```

```
17  }
```

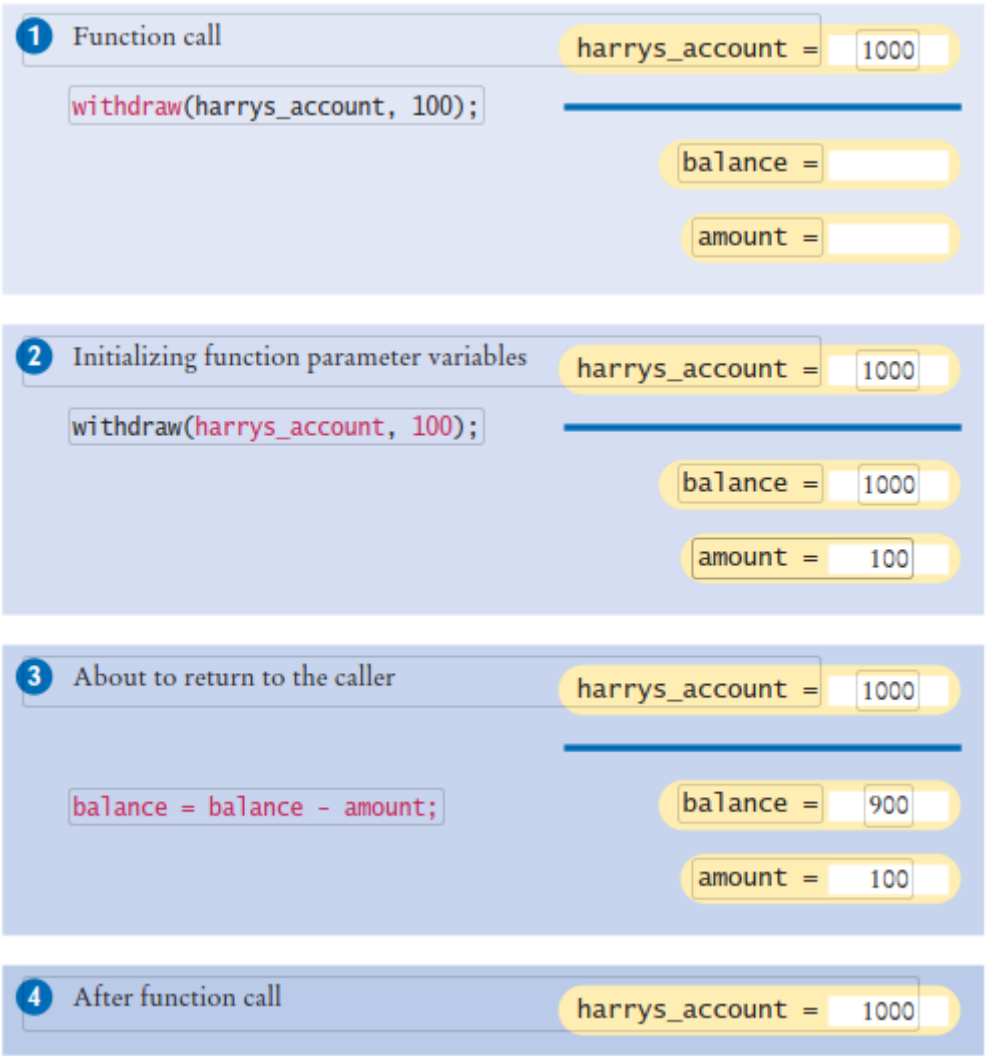
28. Which line defines a global variable?

29. Which lines define local variables named x?

30. Which lines are in the scope of the definition of x in line 2?

31. Which variable is changed by the assignment in line 14?

32. This program defines two variables with the same name whose scopes don't overlap. What are they?



ch05/account.cpp

```
1  #include <iostream>
2
3  using namespace std;
```

```
4 5 /**
6 Withdraws the amount from the given balance, or withdraws
7 a penalty if the balance is insufficient.
8 @param balance the balance from which to make the withdrawal 9 @param amount the amount to withdraw
10 */
11 void withdraw(double& balance, double amount)
12 {
13 const double PENALTY = 10;
14 if (balance >= amount)
15 {
16 balance = balance - amount;
17 }
18 else
19 {
20 balance = balance - PENALTY;
21 }
22 }
23
24 int main()
25 {
26 double harrys_account = 1000;
27 double sallys_account = 500;
28 withdraw(harrys_account, 100);
29 // Now harrys_account is 900
30 withdraw(harrys_account, 1000); // Insufficient funds
31 // Now harrys_account is 890
```

```
32 withdraw(sallys_account, 150);
33 cout << "Harry's account: " << harrys_account << endl;
34 cout << "Sally's account: " << sallys_account << endl; 35
36 return 0;
37 }
```

program run

Harry's account: 890 Sally's account: 350

ch05/intname.cpp

```
1 #include <iostream>
2 #include <string>
3
4 using namespace std;
5
6 /**
7 Turns a digit into its English name.
8 @param digit an integer between 1 and 9
9 @return the name of digit (“one” ... “nine”)
10 */
11 string digit_name(int digit)
12 {
13 if (digit == 1) return "one"; 14 if (digit == 2)
return "two"; 15 if (digit == 3) return "three";
16 if (digit == 4) return "four"; 17 if (digit == 5)
return "five"; 18 if (digit == 6) return "six";
19 if (digit == 7) return "seven"; 20 if (digit == 8)
return "eight"; 21 if (digit == 9) return "nine";
22 return "";
23 }
```

```
24
25 /**
26 Turns a number between 10 and 19 into its English name.
27 @param number an integer between 10 and 19
28 @return the name of the given number (“ten” ... “nineteen”)
29 */
30 string teen_name(int number)
31 {
32 if (number == 10) return "ten";
33 if (number == 11) return "eleven";
34 if (number == 12) return "twelve";
35 if (number == 13) return "thirteen";
36 if (number == 14) return "fourteen";
37 if (number == 15) return "fifteen";
38 if (number == 16) return "sixteen";
39 if (number == 17) return "seventeen";
40 if (number == 18) return "eighteen";
41 if (number == 19) return "nineteen";
42 return "";
43 }
44
45 /**
46 Gives the name of the tens part of a number between 20 and 99.
47 @param number an integer between 20 and 99
48 @return the name of the tens part of the number (“twenty” ... “ninety”) 49 */
```

```
50 string tens_name(int number)
51 {
52 if (number >= 90) return "ninety";
53 if (number >= 80) return "eighty";
54 if (number >= 70) return "seventy";
55 if (number >= 60) return "sixty";
56 if (number >= 50) return "fifty";
57 if (number >= 40) return "forty";
58 if (number >= 30) return "thirty";
59 if (number >= 20) return "twenty";
60 return "";
61 }
62
63 /**
64 Turns a number into its English name.
65 @param number a positive integer < 1,000
66 @return the name of the number (e.g. “two
hundred seventy four”)
67 */
68 string int_name(int number)
69 {
70 int part = number; // The part that still needs
to be converted
71 string name; // The return value
72
73 if (part >= 100)
74 {
75 name = digit_name(part / 100) + " hundred"; 76 part
= part % 100;
```

```
77 }
78
79 if (part >= 20)
80 {
81 name = name + " " + tens_name(part);
82 part = part % 10;
83 }
84 else if (part >= 10)
85 {
86 name = name + " " + teen_name(part);
87 part = 0;
88 }
89
90 if (part > 0)
91 {
92 name = name + " " + digit_name(part);
93 }
94
95 return name;
96 }
97
98 int main()
99 {
100 cout << "Please enter a positive integer: "; 101 int
input;
102 cin >> input;
103 cout << int_name(input) << endl;
104 return 0;
105 }
```

program run

Please enter a positive integer: 729

seven hundred twenty nine

Nazorat savollari

10.C++da funksiya qanday ishlaydi?

11.funksiyaga kutubxona kerakmi?

12.For operatori funksiyada qanday ishlatiladi?

13.Matematik funksiyalar qanday ishlaydi?

14.Funksiya parametrlar nima?

15.Funksiya qanday chaqiriladi?

16.Funksiya parametrlari orqali nima uzatiladi?

17.If operatorining nechta turi bor?

18.O’zgaruvchilar nima uchun qo’llaniladi?

14-Ma’ruza. Rekursiv funksiyalar

Ma’ruza rejasi:

14.1 Rekursiv jarayon nima

14.2 Rekursiv funksilari

14.3 Rekursiv funksiya parametrlari

Kalit so’zlar:*, ro’yxat, manzil, nolinchi ko’rchsatkich, tugun, adres olish &, bo’shatish, ko’rsatkich, virtual destruktor, xotira, xotira chiqishi, destruktor, toifani o’zlashtirish, resurslar chiqishi, a’zo destruktori.*

Joylashtiriladigan (inline) funksiyalar

Kompilyator ishlashi natijasida har bir funksiya mashina kodi ko‘rinishida bo‘ladi. Agar programmada funksiyani chaqirish ko‘rsatmasi bo‘lsa, shu joyda funksiyani adresi bo‘yicha chaqirishning mashina kodi shakllanadi. Odatda funksiyani chaqirish protsessor tomonidan qo‘shimcha vaqt va xotira resurslarini talab qiladi. SHu sababli, agar chaqiriladigan funksiya hajmi unchalik katta bo‘lmagan hollarda, kompilyatorga funksiyani chaqirish kodi o‘rniga funksiya tanasini o‘zini joylashtirishga ko‘rsatma berish mumkin. Bu ish funksiya prototipini inline kalit so‘zi bilan e’lon qilish orqali amalga oshiriladi. Natijada hajmi oshgan, lekin nisbatan tez bajariladigan programma kodi yuzaga keladi.

Funksiya kodi joylashtiriladigan programmaga misol.

```
#include <iostream.h>
inline int Summa(int,int);
int main()
{
 int a=2,b=6,c=3;
 char yangi_qator='\n';
 cout<<Summa(a,b)<<yangi_qator;
```

```
 cout<<Summa(a,c)<<yangi_qator;
 cout<<Summa(b,c)<<yangi_qator;
 return 0;
}
int Summa(int x,int y)
{
 return x+y;
}
```

Keltirilgan programma kodini hosil qilishda Summa() funksiyasi chaqirilgan joylarga uning tanasidagi buyruqlar joylashtiriladi.

Rekursiv funksiyalar

YUqorida qayd qilingandek *rekursiya* deb funksiya tanasida shu funksiyaning o‘zini chaqirishiga aytiladi. Rekursiya ikki xil bo‘ladi:

1) *oddiy* - agar funksiya o‘z tanasida o‘zini chaqirsa;

vositali - agar birinchi funksiya ikkinchi funksiyani chaqirsa, ikkinchisi esa o‘z navbatida birinchi funksiyani chaqirsa.

Odatda rekursiya matematikada keng qo‘llaniladi. CHunki aksariyat matematik formulalar rekursiv aniqlanadi. Misol tariqasida faktorialni hisoblash formulasini

$$n!=\begin{cases}1, & \text{agar } n=0;\\ n*(n-1)!, & \text{agar } n>0,\end{cases}$$

va sonning butun darajasini hisoblashni ko‘rishimiz mumkin:

$$x^n=\begin{cases}1, & \text{agar } n=0;\\ x*x^{n-1}, & \text{agar } n>0.\end{cases}$$

Ko‘rinib turibdiki, navbatdagi qiymatni hisoblash uchun funksiyaning «oldingi qiymati» ma’lum bo‘lishi kerak. S++ tilida rekursiya matematikadagi rekursiyaga o‘xshash. Buni yuqoridagi misollar uchun tuzilgan funksiyalarda ko‘rish mumkin. Faktorial uchun:

```
long F(int n)
```

```
{
 if(!n) return 1;
 else return n*F(n-1);
}
```

Berilgan haqiqiy x soning n- darajasini hisoblash funksiyasi:

```
double Butun_Daraja(double x, int n)
{
 if(!n) return 1;
 else return x*Butun_Daraja(x,n-1);
}
```

Agar faktorial funksiyasiga n>0 qiymat berilsa, quyidagi holat ro‘y beradi: shart operatorining else shoxidagi qiymati (n qiymati) stekda eslab qolinadi. Hozircha qiymati noma’lum n-1 faktorialni hisoblash uchun shu funksiyaning o‘zi n-1 qiymati bilan bilan chaqiriladi. O‘z navbatida, bu qiymat ham eslab qolinadi (stekka joylanadi) va yana funksiya chaqiriladi va hakoza. Funksiya n=0 qiymat bilan chaqirilganda if operatorining sharti (!n) rost bo‘ladi va «return 1;» amali bajarilib, ayni shu chaqirish bo‘yicha 1 qiymati qaytariladi. SHundan keyin «teskari» jarayon boshlanadi - stekda saqlangan qiymatlar ketma-ket olinadi va ko‘paytiriladi: oxirgi qiymat - aniqlangandan keyin (1), u undan oldingi saqlangan qiymatga 1 qiymatiga ko‘paytirib F(1) qiymati hisoblanadi, bu qiymat 2 qiymatiga ko‘paytirish bilan F(2) topiladi va hakoza. Jarayon F(n) qiymatini hisoblashgacha «ko‘tarilib» boradi. Bu jarayonni, n=4 uchun faktorial hisoblash sxemasini 5.2-rasmda ko‘rish mumkin:

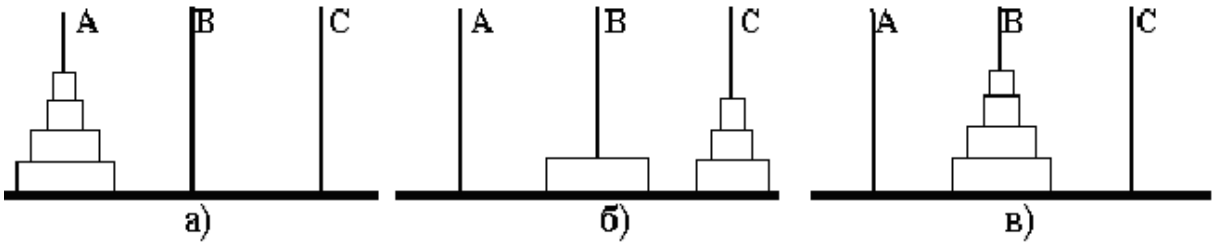
↓	F(4)=4*F(3)	↓	F(4)=4*F(3)	↓	F(4)=4*F(3)	↓	F(4)=4*F(3)	↑	F(4)=4*6
↓	F(3)=3*F(2)	↓	F(3)=3*F(2)	↓	F(3)=3*F(2)	↑	F(3)=3*2		
↓	F(2)=2*F(1)	↓	F(2)=2*F(1)	↑	F(2)=2*1				
↓	F(1)=1*F(0)	↑	F(1)=1*1						
↑	F(0)=1								

14.1-rasm. 4! hisoblash sxemasi

Rekursiv funksiyalarni to‘g‘ri amal qilishi uchun rekursiv chaqirishlarning to‘xtash sharti bo‘lishi kerak. Aks holda rekursiya to‘xtamasligi va o‘z navbatida funksiya ishi tugamasligi mumkin. Faktorial hisoblashida rekursiv tushishlarning to‘xtash sharti funksiya parametri n=0 bo‘lishidir (shart operatorining rost shoxi).
Har bir rekursiv murojaat qo‘shimcha xotira talab qiladi - funksiyalarning lokal ob’ektlari (o‘zgaruvchilari) uchun har bir murojaatda stekdan yangidan joy ajratiladi. Masalan, rekursiv funksiyaga 100 marta murojaat bo‘lsa, jami 100 lokal ob’ektlarning majmuasi uchun joy ajratiladi. Ayrim hollarda, ya’ni rekursiyalar soni etarlicha katta bo‘lganda, stek o‘lchami cheklanganligi sababli (real rejimda 64Kb o‘lchamgacha) u to‘lib ketishi mumkin. Bu holatda programma o‘z ishini «Stek to‘lib ketdi» xabari bilan to‘xtadi.

Quyida, rekursiya bilan samarali echiladigan «Xanoy minorasi» masalasini ko‘raylik.

Masala. Uchta A, B, C qoziq va n-ta har xil o‘lchamli xalqalar mavjud. Xalqalarni o‘lchamlari o‘sish tartibida 1 dan n gacha tartib-langan. Boshda barcha xalqalar A qoziqqa 5.3a -rasmdagidek joylash-tirilgan. A qoziqdagi barcha xalqalarni B qoziqqa, yordamchi S qoziqdan foydalangan holda, quyidagi qoidalarga amal qilgan holda o‘tkazish talab etiladi: xalqalarni bittadan ko‘chirish kerak va katta o‘lchamli xalqani kichik o‘lchamli xalqa ustiga qo‘yish mumkin emas.



14.2-rasm. Xanoy minorasi masalasini echish jarayoni

Amallar ketma-ketligini chop etadigan («Xalqa q dan r ga o‘tkazilsin» ko‘rinishida, bunda q va r - 5.3-rasmdagi A,V yoki S xalqalar). Berilgan n ta xalqa uchun masala echilsin.

Ko‘rsatma: xalqalarni A dan B ga to‘g‘ri o‘tkazishda 5.3b –rasmlar-dagi holat yuzaga keladi, ya’ni n xalqani A dan B o‘tkazish masalasi n-1 xalqani A dan S ga o‘tkazish, hamda bitta xalqani A dan B o‘tkazish masalasiga keladi. Undan keyin S qoziqdagi n-1 xalqali A qoziq yordamida B qoziqqa o‘tkazish masalasi yuzaga keladi va hakoza.

```
#include <iostream.h>
void Hanoy(int n,char a='A',char b='B',char c='C')
{
 if(n)
 {
  Hanoy(n-1,a,s,b);
  cout<<”Xalqa”<< a<<” dan ”<<b<<” ga
o’tkazilsin\n”;
  Hanoy(n-1,c,b,a);
  }
}
int main()
{unsigned int Xalqalar_Soni;
 cout<<”Hanoy minorasi masalasi”<<endl;
 cout<<”Xalqalar sonini kiriting: ”;
 cin>>Xalqalar_Soni;
 Hanoy(Xalqalar_Soni);
 return 0;
}
```

Xalqalar soni 3 bo‘lganda (Xalqalar_Soni=3) programma ekranga xalqalarni ko‘chirish bo‘yicha amallar ketma-ketligini chop etadi:

Xalqa A dan B ga o’tkazilsin

Xalqa A dan C ga o’tkazilsin

Xalqa B dan C ga o’tkazilsin

Xalqa A dan B ga o’tkazilsin

Xalqa C dan A ga o’tkazilsin

Xalqa C dan B ga o’tkazilsin

Xalqa A dan B ga o’tkazilsin

Rekursiya chiroyli, ixcham ko‘ringani bilan xotirani tejash va hisoblash vaqtini qisqartirish nuqtai-nazaridan uni imkon qadar iterativ hisoblash bilan almashtirilgani ma’qul. Masalan, x haqi-qiy sonining n-darajasini hisoblashning quyidagi echim varianti nisbatan kam resurs talab qiladi (n- butun ishorasiz son):

```
double Butun_Daraja(double x, int n)
{
 double p=1;
 for(int i=1; i<=n; i++)p*=x;
 return p;
}
```

Ikkinchi tomondan, shunday masalalar borki, ularni echishda rekursiya juda samarali, hattoki yagona usuldir. Xususan, grammatik tahlil masalalarida rekursiya juda ham o‘ng‘ay hisoblandi.

ch05/account.cpp

```
1 #include <iostream>
2
3 using namespace std;
4 5 /**
6 Withdraws the amount from the given balance, or
withdraws
7 a penalty if the balance is insufficient.
8 @param balance the balance from which to make the
withdrawal 9 @param amount the amount to withdraw
10 */
11 void withdraw(double& balance, double amount)
```

```
12 {
13 const double PENALTY = 10;
14 if (balance >= amount)
15 {
16 balance = balance - amount;
17 }
18 else
19 {
20 balance = balance - PENALTY;
21 }
22 }
23
24 int main()
25 {
26 double harrys_account = 1000;
27 double sallys_account = 500;
28 withdraw(harrys_account, 100);
29 // Now harrys_account is 900
30 withdraw(harrys_account, 1000); // Insufficient
funds
31 // Now harrys_account is 890
32 withdraw(sallys_account, 150);
33 cout << "Harry's account: " << harrys_account << endl;
34 cout << "Sally's account: " << sallys_account << endl; 35
36 return 0;
37 }
```

program run

Harry's account: 890 Sally's account: 350

Nazorat savollari

1. C++da funksiya qanday ishlaydi?
2. funksiyaga kutubxona kerakmi?
3. Rekursiv funksiya nima?
4. Rekursiv funksiya parametrlari.
5. For operatori funksiyada qanday ishlatiladi?
6. Matematik funksiyalar qanday ishlaydi?
7. Funksiya parametrlar nima?
8. Funksiya qanday chaqiriladi?
9. Funksiya parametrlari orqali nima uzatiladi?
10. If operatorining nechta turi bor?
11. O’zgaruvchilar nima uchun qo’llaniladi?

15-Ma’ruza. Massivlar

Ma’ruza rejasi:

15.1 Bir o’lchovli massivlar

15.2 Massivlarni navlarga ajratish

Kalit so’zlar:*, ro’yxat, manzil, nolinchi ko’rchsatkich, tugun, adres olish &, bo’shatish, ko’rsatkich, virtual destruktor, xotira, xotira chiqishi, destruktor, toifani o’zlashtirish, resurslar chiqishi, a’zo destruktori.*

15.1. Bir o’lchovli massivlar

Massiv tushunchasi. Massiv bu bir turli nomerlangan ma’lumotlar jamlanmasidir. Massiv indeksli o’zgaruvchi tushunchasiga mos keladi. Massiv ta’riflanganda turi, nomi va indekslar chegarasi ko’rsatiladi. Masalan, type turidagi length ta elementdan iborat a nomli massiv shunday e’lon qilinadi:

```
type a[length];
```

Bu maxsus a[0], a[1], ..., a[length -1] nomlarga ega bo’lgan type turidagi o’zgaruvchilarning e’lon qilinishiga to’g’ri keladi.

Massivning har bir elementi o’z raqamiga - indeksga ega. Massivning x-nchi elementiga murojaat indekslash operasiyasi yordamida amalga oshiriladi:

```
int x = ...;      //butun sonli indeks
TYPE value = a[x];   //x-nchi elementni o‘qish
a[x] = value;     //x- elementga yozish
```

Indeks sifatida butun tur qiymatini qaytaradigan har qanday ifoda qo'llanishi mumkin: char, short, int, long. C da massiv elementlarining indekslari 0 dan boshlanadi (1 dan emas), length elementdan iborat bo'lgan massivning oxirgi elementining indeksi esa - bu length -1 (length emas) ga teng. Massivning int z[3] shakldagi ta'rifi, int turiga tegishli z[0], z[1], z[2] elementlardan iborat massivni aniqlaydi.

Massiv chegarasidan tashqariga chiqish (ya'ni mavjud bo'lmagan elementni o'qish/yozishga urinish) dastur bajarilishida kutilmagan natijalarga olib kelishi mumkin. Shuni ta'kidlab o'tish lozimki, bu eng ko'p tarqalgan xatolardan biridir.

Agar massiv inisializasiya qilinganda elementlar chegarasi ko'rsatilgan bo'lsa, ro'yxatdagi elementlar soni bu chegaradan kam bo'lishi mumkin, lekin ortiq bo'lishi mumkin emas.

Misol uchun int a[5] = {2,-2}. Bu holda a[0] va a[1] qiymatlari aniqlangan bo'lib, mos holda 2 va –2 ga teng. Agar massiv uzunligiga qaraganda kamroq element berilgan bo'lsa, qolgan elementlar 0 hisoblanadi:

int a10[10] = {1, 2, 3, 4}; //va 6 ta nol

Agar nomlangan massivning tavsifida uning o'lchamlari ko'rsatilmagan bo'lsa, kompilyator tomonidan massiv chegarasi avtomatik aniqlanadi:

int a3[] = {1, 2, 3};

Massivda musbat elementlar soni va summasini hisoblash.

```
#include<stdio.h>
int main()
{
int s = 0,k = 0;
int x[] = {-1,2,5,-4,8,9};
for(int i = 0; i<6; i++)
{
if (x[i]< = 0) continue;
```

```
k++;
s+ = x[i];
};
printf("%d\n",k);
printf("%d\n",s);
return 0;
};
```

Massivning eng katta, eng kichik elementi va o’rta qiymatini aniqlash:

```
#include<stdio.h>
int main()
{
int i,j,n;
float min,max,s = 0;
float a,b,d,x[100];
while(1)
{
printf("\n n = ");
scanf("%d",&n);
if ( n>0 && n< = 100 ) break;
printf("\n Hato 0<n<101 bo'lishi kerak");
}
printf("\n elementlar qiymatlarini kiriting:\n");
for (i = 0;i<n;i++)
{
printf("x[%d] = ",i);
scanf("%f",&x[i]);
}
max = x[0];
```

```
min = x[0];
for(i = 0;i<n;i++)
{
s+ = x[i];
if (max<x[i]) max = x[i];
if (min>x[i]) min = x[i];
};
s/ = n;
printf("\n max = %f",max);
printf("\n min = %f",min);
printf("\n o'rta qiymat = %f",s);
return 0;
};
```

15.2 Massivlarni navlarga ajratish

Navlarga ajratish - bu berilgan ko’plab ob’ektlarni biron-bir belgilangan tartibda qaytadan guruhlash jarayoni.

Massivlarning navlarga ajratilishi tez bajarilishiga ko’ra farqlanadi. Navlarga ajratishning n*n ta qiyoslashni talab qilgan oddiy usuli va n*log(n) ta qiyoslashni talab qilgan tez usuli mavjud. Oddiy usullar navlarga ajratish tamoyillarini tushuntirishda qulay hisoblanadi, chunki sodda va kalta algoritmlarga ega. Murakkablashtirilgan usullar kamroq sonli operasiyalarni talab qiladi, biroq operasiyalarning o’zi murakkabroq, shuning uchun uncha katta bo’lmagan massivlar uchun oddiy usullar ko’proq samara beradi.

Oddiy usullar uchta asosiy kategoriyaga bo’linadi:

- oddiy kiritish usuli bilan navlarga ajratish;
- oddiy ajratish usuli bilan navlarga ajratish;
- oddiy almashtirish usuli bilan navlarga ajratish.

Oddiy kiritish usuli bilan navlarga ajratish

Massiv elementlari avvaldan tayyor berilgan va dastlabki ketma-ketliklarga bo'linadi. i = 2 dan boshlab, har bir qadamda dastlabki ketma-ketlikdan i-nchi element chiqarib olinadi hamda tayyor ketma-ketlikning kerakli o'rniga kiritib qo'yiladi. Keyin i bittaga ko'payadi va h.k.

Kerakli joyni izlash jarayonida, ko'proq o'ngdan bitta pozisiyadan tanlab olingan elementni uzatish amalga oshiriladi, ya'ni tanlab olingan element, j: = i-1 dan boshlab, navlarga ajratib bo'lingan qismning navbatdagi elementi bilan qiyoslanadi. Agar tanlab olingan element a[i] dan katta bo'lsa, uni navlarga ajratish qismiga qo'shadilar, aks holda a[j] bitta pozisiyaga suriladi, tanlab olingan elementni esa navlarga ajratilgan ketma-ketlikning navbatdagi elementi bilan qiyoslaydilar. To'g'ri keladigan joyni qidirish jarayoni ikkita turlicha shart bilan tugallanadi:

agar a[j]>a[i] elementi topilgan bo'lsa;
agar tayyor ketma-ketlikning chap uchiga bo'lsa.

```
int i, j, x;
for(i = 1; i < n; i++)
{
x = [i]; // kiritib qo‘yishimiz lozim bo‘lgan
elementni esda saqlab qolamiz
j = i - 1;
while(x<a[j]&&j> = 0)//to‘g‘ri keladigan joyni
qidirish
}
a[j+1] = a[j]; //o‘ngga surish
j--;
}
a[j+1] = x; //elementni kiritish
}
```

Oddiy tanlash usuli bilan navlarga ajratish

Massivning minimal elementi tanlanadi hamda massivning birinchi elementi bilan joy almashtiriladi. Keyin jarayon qolgan elementlar bilan takrorlanadi va h.k.

```
int i, min, n_min, j;
for(i = 0; i < n-1; i++)
{
min = a[i]; n_min = i; //minimal qiymatni qidirish
for(j = i + 1; j < n; j++)
if(a[j] < min){min = a[j]; n + min = j;}
a[n_min] = a[i]; //almashtirish
a[i] = min;
}
```

Oddiy almashtirish usuli bilan navlarga ajratish

Elementlar juftlari oxirgisidan boshlab qiyoslanadi va o’rin almashinadi. Natijada massivning eng kichik elementi uning eng chapki elementiga aylanadi. Jarayon massivning qolgan elementlari bilan davom ettiriladi.

```
for(int i = 1; i < n; i++)
for(int j = n - 1; j > = i; j—
if(a[j] < a[j-1])
{int r = a[j]; a[j] = a[j-1]; a[j - 1] = r;}
}
```

Bir o’lchamli massivlarni funksiya parametrlari sifatida uzatish. Massivdan funksiya parametri sifatida foydalanganda, funksiyaning birinchi elementiga ko’rsatkich uzatiladi, ya’ni massiv hamma vaqt adres bo’yicha uzatiladi. Bunda massivdagi elementlarning miqdori haqidagi axborot yo’qotiladi, shuning uchun massivning o’lchamlari haqidagi ma’lumotni alohida parametr sifatida uzatish kerak.

Misol:

Massiv barcha elementlari chiqarilsin:

```
#include <stdio.h>
#include <stdlib.h>
int form(int a[100])
{
int n;
printf("\nEnter n:");
scanf("%d",&n);
for(int i = 0;i < n; i++)
a[i] = rand()%100;
return n;
}
void print(int a[100], int n)
{
for(int i = 0;i < n; i++)
printf("%d ", a[i]);
printf("\n");
}
int main()
{
int a[100];
```

```
int n;
n = form(a);
print(a,n);
return 0;
}
```

Funksiyaga massiv boshlanishi uchun ko’rsatkich uzatilgani tufayli (adres bo’yicha uzatish), funksiya tanasining operatorlari hisobiga massiv o’zgarishi mumkin.

Funksiyalarda bir o’lchovli sonli massivlar argument sifatida ishlatilganda ularning chegarasini ko’rsatish shart emas.

Misol. Massivdan barcha juft elementlar chiqarilsin.

```
#include <stdio.h>
#include <stdlib.h>
void form(int a[], int n)
{
for(int i = 0; i < n; i++)
a[i] = rand()%100;
}
void print(int a[],int n)
{
for(int i = 0; i < n; i++)
printf("%d ", a[i]);
printf("\n");
}
int Dell(int a[],int n)
{
```

```
int j = 0, i, b[100];
for(i = 0; i < n; i++)
if(a[i]%2! = 0)
{
b[j] = a[i]; j++;
}
for(i = 0; i < j; i++) a[i] = b[i];
return j;
}
int main()
{
int a[100];
int n;
printf("\nEnter n:");
scanf("%d",&n);
form(a, n);
print(a, n);
n = Dell(a, n);
print(a, n);
system("pause");
return 0;
}
```

Funksiyalarda bir o'lchovli sonli massivlar argument sifatida ishlatilganda ularning chegarasini ko'rsatish shart emas. Misol tariqasida n o'lchovli vektorlar bilan bog'liq funksiyalarni ta'riflashni ko'rib chiqamiz.

Vektorning uzunligini aniqlash funksiyasi:

```
float mod_vec(int n, float x[])
```

```
{
float a = 0;
for (int i = 0; i < n; i++)
 a+ = x[i] * x[i];
return sqrt(double(a));
}
```

Funksiyalarda bir o'lchovli massivlar qaytariluvchi qiymatlar sifatida ham kelishi mumkin. Misol uchun ikki vektor summasini hisoblovchi funksiya prosedurani ko'ramiz:

```
void sum_vec(int n, float a, float b, float c)
{
for(int i = 0;i < n; i++,c[i] = a[i] + b[i]);
}
```

Bu funksiyaga quyidagicha murojaat qilish mumkin:

```
float a[] = {1, -1.5, -2};
b[] = {-5.2, 1.3, -4},c[3];
sum_vec(3, a, b, c);
```

Polinom. Polinom qiymatini hisoblash funksiyasi poly deb nomlanadi.

Prototipi:

double poly(double x, int degree, double coeffs[]);

Algebraik polinom koeffisientlari coeffs[0], coeffs[1], ..., coeffs[degree] massiv elementlarida beriladi.

Misol:

```
#include <stdio.h>
#include <math.h>
```

```
/* polynomial: x**3 - 2x**2 + 5x - 1 */
int main(void)
{
double array[] = { -1.0, 5.0, -2.0, 1.0};
double result;
result = poly(2.0, 3, array);
printf("The polynomial: x**3 - 2.0x**2 + 5x - 1 at
2.0 is %lf\n", result);
return 0;
}
```

16-Ma’ruza. Massivlar uchun umumiy algoritmlar

Ma’ruza rejasi:

16.1 Umumiy massiv algoritmlari

16.1.1 To’ldirish

16.1.2 Nusxa ko’chirish

16.1.3 Yig’indi va O’rtacha Qiymat

Kalit so’zlar:*, ro’yxat, manzil, nolinchi ko’rchsatkich, tugun, adres olish &, bo’shatish, ko’rsatkich, virtual destruktor, xotira, xotira chiqishi, destruktor, toifani o’zlashtirish, resurslar chiqishi, a’zo destruktori.*

16.1 Umumiy massiv algoritmlari

Quyidagi bo’limlarda biz, qiymatlar ketma- ketligini qayta ishlash uchun ba’zi eng keng tarqalgan algoritmlarni muhokama qilamiz.Biz algoritmlarni shunday taqdim etamizki, siz ularni to’liq va qisman to’ldirilagan massivlar hamda (6.7 bo’limda tanishtiradigan)vektorlar yordamida foydalanishingiz mumkin.Qiymatlar miqdori (size of values) ifodasini ishlatganimizda, siz uni massivda elementlar miqdorini anglatuvchi doimiy yoki o’zgaruvchan bilan almashtirishingiz kerak.

(yoki values.size() if values is a vector ifodasi.)

16.1.1 To’ldirish

Bu sikl massivni nollar bilan to’ldiradi:

```
for (int i = 0; i < size of values; i++)
    {
      values[i] = 0;
    }
```

So’ngra, keling kvadratlar massivini 0, 1, 4, 9, 16, va h.z sonlar bilan to’ldiramiz. E’tibor bering, 0 indeksli element 02 ni , 1 indeksli element 12 va h.z ni o’z ichiga oladi.

```
for (int i = 0; i < size of squares; i++)
{
  squares[i] = i * i;
}
```

16.1.2 Nusxa ko’chirish

Ikkita massivni ko’rib chiqamiz:

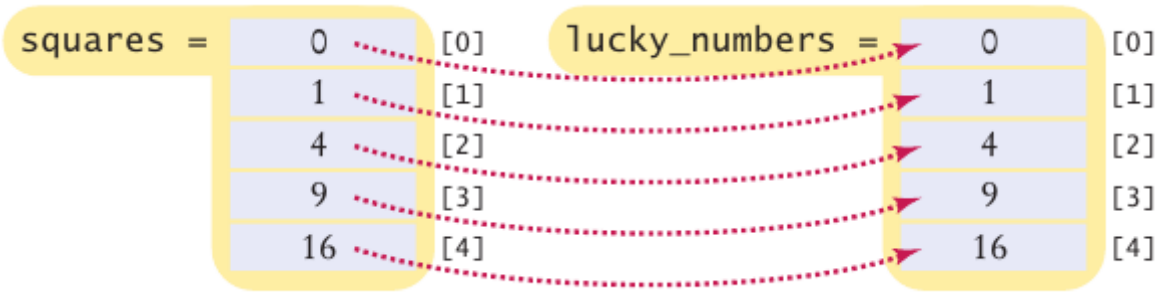
```
int squares[5] = { 0, 1, 4, 9, 16 };
int lucky_numbers[5];
```

Hozir aytaylik, siz birinchi massivdagi barcha qiymatlarni ikkinchisiga ko’chirmoqchisiz. Quyidagi ifoda xato hisoblanadi:

```
lucky_numbers = squares; // Error
```

C++da siz bir massivni ikkinchisiga o’tkaza olmaysiz.Uning o’rniga siz barcha elementlarni ko’chirish uchun sikldan foydalanishingiz kerak:

```
for (int i = 0; i < 5; i++)
{
  lucky_numbers[i] = squares[i];
}
```



16.1 shakl massivndan nusxa olishuchun elementlarni ko’chirish

16.1.3 Yig’indi va O’rtacha Qiymat

Siz allaqachon bu algoritm bilan 4.7.1. bo'limda duch kelgansiz.Quyida massivning barcha elementlarining yig'indisini hisoblash kodi berilgan.

```
double total = 0;
for (int i = 0; i < size of values; i++)
{
  total = total + values[i];
}
```

O'rtacha qiymatni topish uchun elementlar miqdorini bo'ling:

```
double average = total / size of values;
```

miqdor nol emasligini tekshirishni unutmang.

16.1.4 Maksimum va minimum

O'zida eng katta element uchun o'zgaruvchini saqlovchi va massivlarni joriy etishda uchratgan, 4.7.4 - bo'limdagi algoritmni qo'llang:

```
double largest = values[0];
for (int i = 1; i < size of values; i++)
{
  if (values[i] > largest)
  {
  largest = values[i];
  }
}
```

Biz eng katta qiymatni [0] qiymati bilan inisializasiyalashtirganligimiz tufayli, sikl 1 dan boshlanishiga e'tibor qarating.Eng kichik qiymatni hisoblash uchun solishtirishni orqaga qaytaring. Bu algoritmlar massiv kamida bitta elementni o'z ichiga olishni talab qiladi.

16.1.5 Element ajratuvchilari razdeliteli

Siz element to’plamlarini namoyish etganingizda, siz odatda ularni vergul yoki quyidagicha vertikal chiziqlar bilan bo’lishni hohlaysiz:

```
1 | 4 | 9 | 16 | 25
```

E’tibor bering, ajratuvchi sonlarga qaraganda bittaga kam. dastlabki(0 indeksli) dan tashqari har qaysi elementdan oldin ajratuvchini yozing.

```
for (int i = 0; i < size of values; i++)
{
  if (i > 0)
  {
  cout << " | ";
  }
  cout << values[i];
}
```

16.1.6 Bir chiziqli qidiruv

Uni o’rnini almashtirish yoki olib tashlash uchunn siz tez- tez element o’rnini topishingiz kerak bo’ladi. Massivning oxiriga kelmaguncha yoki o’xshashlik topmaguningizcha, barcha elementlarni ko’rib chiqing. Quyida biz 100ga teng bo’lgan birinchi element o’rnini qidiramiz.

```
bool found = false;
while (pos < size of values && !found)
{
  if (values[pos] == 100)
  {
  found = true;
  }
  else
  {
  pos++;
  }
```

```
}
```

Agar topilma to’g’ri bo’lsa, joy birinchi moslikning o’rni hisoblanadi.

6.2.7 Elementni olib tashlash

Joriy miqdori o’zgaruvchan joriy miqdorda saqlanovchi qisman to’ldirilgan mas siv miqdorini ko’rib chiqamiz. Deylik, siz pos indeksli elementni qiymatdan olib tashlamoqchisiz.Agar elementlar hyech qanday aniq tartibda bo’lmasa, bu vazifani amalag oshirish yengil kechadi.Shunchaki, oxirgi elementdan olib tashlangan elementni qaytadan yozing.So’ng, miqdorni kuzatuvchi o’zgaruvchini kamaytiring. (6 - shaklga qarang).

```
values[pos] = values[current_size - 1];
current_size--;
```

Agar elementlar tartibi muhim bo’lsa vaziyat yanada murakkablashadi. Bunday holda siz barcha elementlarni keyingi indeksi pastroq elementga almashtirishingiz kerak.So’ng massivni miqdorini saqlagan holda o’zgaruvchini kamaytirishingiz kerak. (7 - shaklga qarang).

```
for (int i = pos + 1; i < current_size; i++)
{
  values[i - 1] = values[i];
}
current_size--;
```

6-7- shakl. Tartibsiz massivda elementni olib tashlash va Tartibli massivda elementni olib tashlash

6.2.8 Element kiritish

Agar element tartibi muhim bo’lmasa, siz oddiygina miqdorni kuzatuvchi o’zgaruvchini ko’paytirib yangi elementni oxiriga kiritishingiz mumkin.

Qisman to’ldirilgan massiv uchun:

```
if (current_size < CAPACITY)
{
  current_size++;
  values[current_size - 1] = new_element;
}
```

Ketma - ketlikning o’rtasida muayan holatda element kiritish ko’proq ishni talab qiladi. Birinchidan, joriy miqdorni o’zida ushlab turgan o’zgaruvchini ko’paytiring.Keyin, barcha elementlarni qo’shish o’rnidan yuqoriroq indeksga ko’chiring. Nihoyat, yangi elementni joylashtiring.Quyida qisman to’ldirilgan massiv uchun kod berilgan:

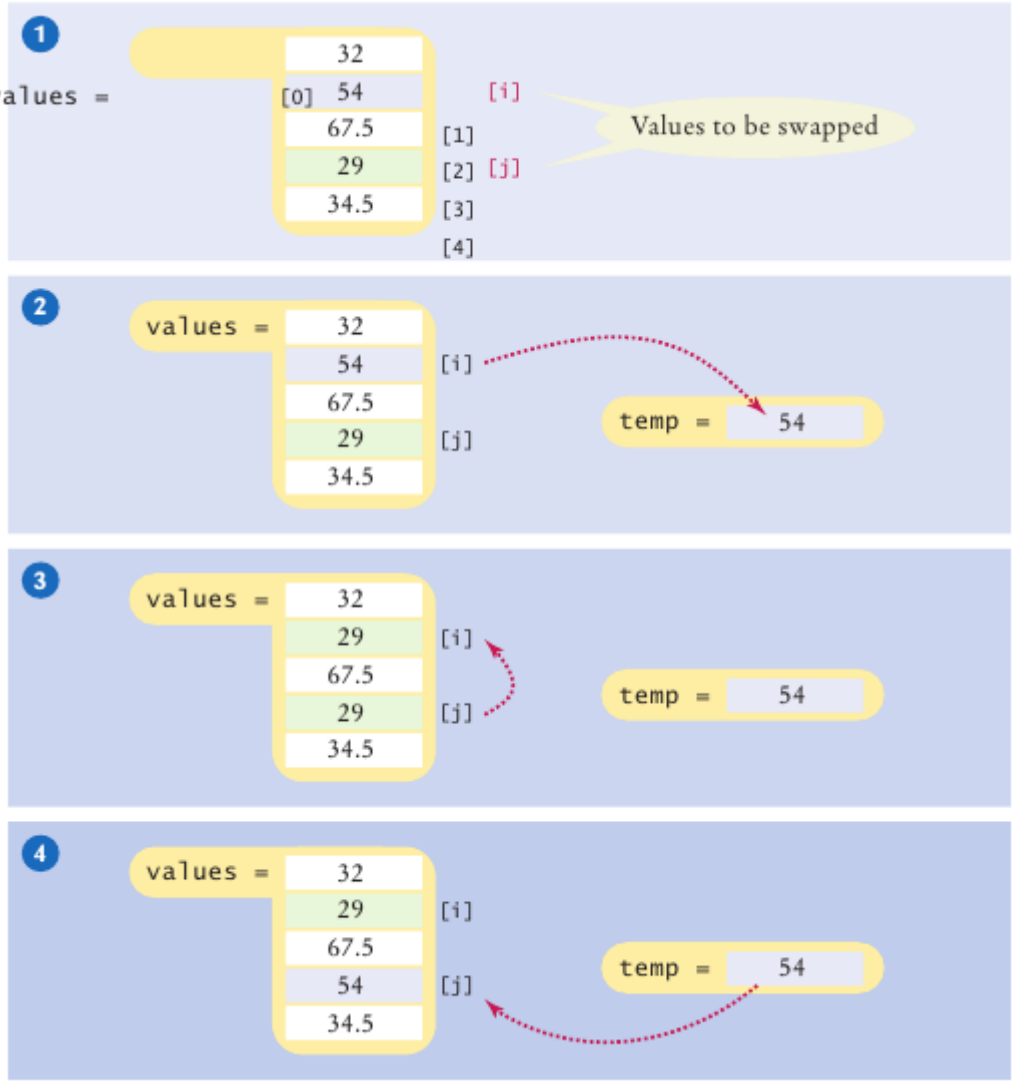
```
if (current_size < CAPACITY)
{
  current_size++;
  for (int i = current_size - 1; i > pos; i--)
  {
  values[i] = values[i - 1];
  }
  values[pos] = new_element;
}
```

Harakatlarning tartibiga e’tibor qarating: Elementni olib tashlaganingizda, birinchi bo’lib keyingi elementni pastroq indeksga o’zgartiring. Keyin esa shu ketma - ketlikda massivni oxiriga yetib borguningizgacha davom ettiring.Yangi element kiritishda siz massivni oxiridan boshlaysiz.Shu elementni yuqoriroq indeksga o’zgartiringg vaa keyinn undann oldingisigaa vaa xakozoo nihoyatt

qo’shishsh joyiga kelguningizgachaa (9- shaklga qarang).

6.2.9 Elementtlar n ailmashtirish

Siz massiv elementlarini tez -tezalmashtirishingiz kerak bo’ladi. Misol uchun, 263 - sahifadagi 6.2 - maxsus mavzudagi saralash algoritmlari elementlarni almashtirish orqali takroran massivni saralaydi.



Massiv qiymatlarining i va j o’rnidagi elementlarni almashtirish vazifasini ko’rib chiqamiz. Biz [i] qiymatlarini [j] qiymatlariga o’rnatishni hoxlar edik, lekin bu ayni paytda [i]qiymatlarida saqlangan qiymatni qaytadan yozganligi sababli biz birinchisini saqlashni istaymiz.

```
double temp = values[i];
```

```
values[i] = values[j];
```

Endi biz [j]qiymatlarini saqlangan qiymatga o’rniatishimiz mumkin.

```
 values[j] = temp;
```

10- shaklda jarayonni ko’rishingiz mumkin.

16.1.10 Kiritishni o’qish

Bilsangiz edi foydalanuvchi qancha kiritish qiymatlari bilan ta’minlashi kerak, shunchaki ularni bitta massivga joylashtirish uchun.

```
double values[NUMBER_OF_INPUTS];
for (i = 0; i < NUMBER_OF_INPUTS; i++)
{
  cin >> values[i];
```

17-Ma’ruza. Massivlar yordamida obyektlarni tadqiq qilish

Ma’ruza rejasi:

17.1 Dinamik massivlar bilan ishlash

17.2 Funksiya va massivlar

Kalit so’zlar:, *ro’yxat, manzil, nolinchi ko’rchsatkich, tugun, adres olish &, bo’shatish, ko’rsatkich, virtual destruktor, xotira, xotira chiqishi, destruktor, toifani o’zlashtirish, resurslar chiqishi, a’zo destruktori.*

Statik massivlarning kamchiliklari shundaki, ularning o‘lchamlari oldindan ma’lum bo‘lishi kerak, bundan tashqari bu o‘lchamlar berilganlarga ajratilgan xotira segmentining o‘lchami bilan chegaralangan. Ikkinchi tomondan, etarlicha katta o‘lchamdagi massiv e’lon qilib, konkret masala echilishida ajratilgan xotira to‘liq ishlatilmasligi mumkin. Bu kamchiliklar dinamik massivlar-dan foydalanish orqali bartaraf etiladi, chunki ular programma ishlashi jarayonida kerak bo‘lgan o‘lchamdagi massivlarni yaratish va zarurat qolmaganda yo‘qotish imkoniyatini beradi.

Dinamik massivlarga xotira ajratish uchun malloc(), calloc() funksiyalaridan yoki new operatoridan foydalanish mumkin. Dina-mik ob’ektga ajratilgan xotirani bo‘shatish uchun free() funksiyasi yoki delete operatori ishlatiladi.

YUqorida qayd qilingan funksiyalar «alloc.h» kutubxonasida joylashgan.

malloc() funksiyasining sintaksisi

```
void * malloc(size_t size);
```

ko‘rinishida bo‘lib, u xotiraning uyum qismidan size bayt o‘lchamidagi uzluksiz sohani ajratadi. Agar xotira ajratish muvaffaqiyatli bo‘lsa, malloc() funksiyasi ajratilgan sohaning boshlanish adresini qaytaradi. Talab qilingan xotirani ajratish muvaffaqiyatsiz bo‘lsa, funksiya NULL qiymatini qaytaradi.

Sintaksisdan ko‘rinib turibdiki, funksiya void turidagi qiymat qaytaradi. Amalda esa konkret turdagi ob’ekt uchun xotira ajratish zarur bo‘ladi. Buning uchun void turini konkret turga keltirish texnologiyasidan foydalaniladi. Masalan, butun turdagi uzunligi 3 ga teng massivga joy ajratishni quyidagicha amalga oshirish mumkin:

```
int * pInt=(int*)malloc(3*sizeof(int));
```

calloc() funksiyasi malloc() funksiyasidan farqli ravishda massiv uchun joy ajratishdan tashqari massiv elementlarini 0 qiymati bilan initsializatsiya qiladi. Bu funksiya sintaksisi

```
void * calloc(size_t num, size_t size);
```

ko‘rinishda bo‘lib, num parametri ajratilgan sohada nechta element borligini, size har bir element o‘lchamini bildiradi.

free() xotirani bo‘shatish funksiyasi o‘chiriladigan xotira bo‘la-giga ko‘rsatkich bo‘lgan yagona parametrga ega bo‘ladi:

```
void free(void * block);
```

free() funksiyasi parametrining void turida bo‘lishi ixtiyoriy turdagi xotira bo‘lagini o‘chirish imkonini beradi.

Quyidagi programmada 10 ta butun sondan iborat dinamik massiv yaratish, unga qiymat berish va o‘chirish amallari bajarilgan.

```
#include <iostream.h>
#include <alloc.h>
int main()
{
 int * pVector;
 if ((pVector=(int*)malloc(10*sizeof(int)))==NULL)
 {
  cout<<”Xotira etarli emas!!!”;
  return 1;
 }
```

```
      // ajratilgan xotira sohasini to‘ldirish
      for(int i=0;i<10;i++) *(pVector+i)=i;
      // vektor elementlarini chop etish
      for(int i=0; i<10; i++) cout<<*(pVector+i)<<endl;
      // ajratilgan xotira bo‘lagini qaytarish
(o‘chirish)
      free(pVector);
      return 0;
     }
```

Keyingi programmada $n \times n$ o‘lchamli haqiqiy sonlar massivi-ning bosh diagonalidan yuqorida joylashgan elementlar yig‘indi-sini hisoblash masalasi echilgan.

```
     #include <iostream.h>
     #include <alloc.h>
     int main()
     {
      int n;
      float * pMatr, s=0;
      cout<<”A(n,n): n=”;
      cin>>n;

if((pMatr=(float*)malloc(n*n*sizeof(float)))==NULL)
      {
       cout<<”Xotira etarli emas!!!”;
       return 1;
      }
      for(int i=0;i<n;i++)
       for(int j=0;j<n;j++)cin>>*(pMatr+i*n+j);
      for(int i=0;i<n;i++)
```

```
 for(int j=i+1;j<n;j++)s+=*(pMatr+i*n+j);
 cout<<"Matritsa bosh diagonalidan yuqoridagi ";
 cout<<"elementlar yig`indisi S="<<s<<endl;
 return 0;
}
```

new operatori yordamida, massivga xotira ajratishda ob'ekt turidan keyin kvadrat qavs ichida ob'ektlar soni ko‘rsatiladi. Masalan, butun turdagi 10 ta sondan iborat massivga joy ajratish uchun

```
pVector=new int[10];
```

ifodasi yozilishi kerak. Bunga qarama-qarshi ravishda, bu usulda ajratilgan xotirani bo‘shatish uchun

```
delete [] pVector;
```

ko‘rsatmasini berish kerak bo‘ladi.

Ikki o‘lchamli dinamik massivni tashkil qilish uchun

```
int **a;
```

ko‘rinishidagi «ko‘rsatkichga ko‘rsatkich» ishlatiladi.

Boshda massiv satrlari soniga qarab ko‘rsatkichlar massiviga dinamik xotiradan joy ajratish kerak:

```
a=new int *[m] // bu erda m massiv satrlari soni
```

Keyin, har bir satr uchun takrorlash operatori yordamida xotira ajratish va ularning boshlang‘ich adreslarini a massiv elementlariga joylashtirish zarur bo‘ladi:

```
for(int i=0;i<m;i++)a[i]=new int[n];//n ustunlar soni
```

SHuni qayd etish kerakki, dinamik massivning har bir satri xotiraning turli joylarida joylashishi mumkin (7.1 va 7.3-rasmlar).

Ikki o‘lchamli massivni o‘chirishda oldin massivning har bir elementi (satri), so‘ngra massivning o‘zi yo‘qotiladi:

```
for(i=0;i<m;i++) delete[]a[i];
delete[]a;
```

Matritsani vektorga ko‘paytirish masalasi uchun dinamik massivlardan foydalanishga misol:

```
void main ()
{
 int n,m;
 int i,j; float s;
 cout<<”\n n=”; cin>>n; // matritsa satrlari soni
 cout<<”\n m=”; cin>>m; // matritsa ustunlari soni
 float *b=new float[m];
 float *c=new float[n];
 // ko‘rsatkichlar massiviga xotira ajratish
 float **a=new float *[n] ;
 for(i=0;i<n;i++)         // har bir satr uchun
 a[i]=new float[m];  //dinamik xotira ajratish
 for(j=0;j<m;j++)cin>>b[j];
 for(i=0;i<n;i++)
  for(j=0;j<m;j++)cin>>a[i][j];
 for(i=0;i<n;i++)
 {
  for(j=0,s=0;j<m;j++)s+=a[i,j]*b[j];
  c[i]=s;
 }
 for(i=0;i<n;i++)cout<<”\t c[”<<i<<”]=”<<c[i];
 delete[]b;
 delete[]c;
 for (i=0;i<n;i++) delete[]a[i];
 delete[]a;
 return;
}
```

Funksiya va massivlar

Funksiyalar massivni parametr sifatida ishlatishi va uni funksiyaning natijasi sifatida qaytarishi mumkin.

Agar massiv parametr orqali funksiyaga uzatilsa, elementlar sonini aniqlash muammosi tug‘iladi, chunki massiv nomidan uning uzunligini aniqlashning iloji yo‘q. Ayrim hollarda, masalan, belgilar massivi sifatida aniqlangan satr (ASCIIZ satrlar) bilan ishlaganda massiv uzunligini aniqlash mumkin, chunki satrlar ‘\0’ belgisi bilan tugaydi.

Misol uchun:

```
#include <iostream.h>
int len(char s[])//massivni parametr sifatida
ishlatish
{
 int m=0;
 while(s[m++]);
 return m-1;
}
void main ()
{
 char z[]=”Ushbu satr uzunligi = ”;
 cout<<z<<len(z);
}
```

Funksiya parametri satr bo‘lmagan hollarda fiksirlangan uzun-likdagi massivlar ishlatiladi. Agar turli uzunlikdagi massivlarni uzatish zarur bo‘lsa, massiv o‘lchamlarini parametr sifatida uzatish mumkin yoki bu maqsadda global o‘zgaruvchidan foydalanishga to‘g‘ri keladi.

Misol:

```
#include <iostream.h>
float sum(int n,float *x) //bu ikkinchi usul
```

```
{
 float s=0;
 for (int i=0;i<n;i++)s+=x[i];
 return s;
}
void main()
{
 float E[]={1.2,2.0,3.0,4.5,-4.0};
 cout<<sum(5,E);
}
```

Massiv nomi ko‘rsatkich bo‘lganligi sababli massiv element-larini funksiyada o‘zgartirish mumkin va bu o‘zgartirishlar funk-siyadan chiqqandan keyin ham saqlanib qoladi.

```
#include <iostream.h>
void vector_01(int n,int*x,int * y) //bu ikkinchi
usul
{
 for (int i=0;i<n;i++)
  y[i]=x[i]>0?1:0;
}
void main()
{
 int a[]=(1,2,-4,3,-5,0,4};
 int c[7];
 vector_01(7,a,c);
 for(int i=0;i<7;i++) cout<<'\t'<<c[i];
}
```

Masala. Butun turdagi va elementlari kamaymaydigan holda tartiblangan bir o‘lchamli ikkita massivlarni yagona massivga, tartiblanish saqlangan holda birlashtirish amalga oshirilsin.

Programma matni:

```
#include <iostream.h>
    \\butun turdagi massivga ko‘rsatkich
qaytaradigan
    \\funksiya
int * massiv_ulash(int,int*,int,int*);
     void main()
{
 int c[]={-1,2,5,10},d[]={1,7,8};
 int * h;
 h=massiv_ulash(5,c,3,d);
 for(int i=0;i<8;i++) cout<<’\t’<<h[i];
 delete[]h;
}
     int * massiv_ulash(int n,int *a ,int m,int
*b);
{
 int * x=new int[n+m];
 int ia=0,ib=0,ix=0;
 while (ia<n && ib<m)
  a[ia]>b[ib]?x[ix++]=b[ib++]:x[ix++]=a[ia++];
 while(ib<m)x[ix++]=b[ib++];
 while(ia<n)x[ix++]=a[ia++];
 return x;
}
```

Ko‘p o‘lchamli massivlar bilan ishlash ma’lum bir murakkablikka ega, chunki massivlar xotirada joylash tartibi turli variantda bo‘lishi mumkin. Masalan, funksiya parametrlar ro‘yxatida $n \times n$ o‘lchamdagi haqiqiy turdagi x[n][n] massivga mos keluvchi parametrni

```
float sum(float x[n][n])
```

ko‘rinishda yozib bo‘lmaydi. Muammo echimi - bu massiv o‘lchamini parametr sifatida uzatish va funksiya sarlavhasini quyidagicha yozish kerak:

```
float sum(int n,float x[][]);
```

Ko‘p o‘lchamli massivlarni parametr sifatida ishlatishda bir nechta usullardan foydalanish mumkin.

1-usul. Massivning ikkinchi o‘lchamini o‘zgarmas ifoda (son) bilan ko‘rsatish:

```
float sum(int n,float x[][10])
{float s=0.0;
 for(int i=0;i<n;i++)
 for(int j=0;j<n;j++)
 s+=x[i][j];
 return s;}
```

2-usul. Ikki o‘lchamli massiv ko‘rsatkichlar massivi ko‘ri-nishida aniqlangan holatlar uchun ko‘rsatkichlar massivini (matritsa satrlar adreslarini) berish orqali:

```
float sum(int n,float *p[])
{
 float s=0.0;
 for(int i=0;i<n;i++)
 for(int j=0;j<n;j++)
 s+=p[i][j];\\”*p[i][j]” emas,chunki massivga
murojat
 return s;
}
```

```
void main()
{
  float x[][4]={{11,-12,13,14},{21,22,23,24},
   {31,32,33,34},{41,42,43,44}};
 float *ptr[4];
 for(int i=0;i<4;i++) ptr[i]=(float *)&x[i];
 cout<<sum(4,ptr)<<endl;
}
```

3-usul. Ko‘rsatkichlarga ko‘rsatkich ko‘rinishida aniqlangan dinamik massivlarni ishlatish bilan:

```
float sum(int n,float **x)
{
 float s=0.0;
 for(int i=0;i<n;i++)for(int
j=0;j<n;j++)s+=x[i][j];   return s;
}
    void main()
    {
     float **ptr;
     int n;
     cin>>n;
     ptr=new float *[n];
     for(int i=0;i<n;i++)
     {
      ptr[i]=new float [n];
      for(int j=0;j<n;j++)
       ptr[i][j]=(float)((i+1)*10+j);
     }
     cout<<sum(n,ptr);
     for(int i=0; i<n;i++) delete ptr[i];
```

```
     delete[]ptr;
    }
```

Navbatdagi programmada funksiya tomonidan natija sifatida ikki o‘lchamli massivni qaytarishiga misol keltirilgan. Massiv elementlarning qiymatlari tasodifiy sonlardan tashkil topadi. Tasoddifiy sonlar «math.h» kutubxonasidagi random() funksiya yordamida hosil qilinadi:

```
#include <iostream.h>
#include <math.h>
int **rmatr(int n,int m)
{
 int ** ptr;
     ptr=new int *[n];
     for(int i=0;i<n;i++)
     {
       ptr[i]=new int[m];
  for(int j=0;j<m;j++) ptr[i][j]=random(100);
     }
     return ptr;
}
int sum(int n,int m,int **ix)
{
 float s=0;
 for(int i=0;i<n;i++)
 for(int j=0;j<m;j++) s+=ix[i][j];
 return s;
}
void main()
{
 int n,m;
```

```
        cin>>n>>m;
        int **matr;
        randomize();
        matr=rmatr(n,m);
        for(int i=0;i<n;i++)
        {
  cout<<endl<<i<<" - satr:"
  for (int j=0;j<m;j++) cout<<'\t'<<matr[i][j];
        }
        cout<<endl<<"Summa="<<sum(n,m,matr);
        for(int i=0;i<n;i++) delete matr[i];
 delete[]matr;
}
```

ch06/medals.cpp

```
1 #include <iostream>
2 #include
<iomanip>yoyoyoy
```

```
11 @param table uch ustunli jadval
12 @param row umulashtirilishi kerak bo'lgan qator
13 @return berilgan qatordagi elementlar majmui
14 */
15 double row_total(int table[][COLUMNS], int row)
16 {
17 int total = 0;
18 for (int j = 0; j < COLUMNS; j++)
19 {
20  total = total + table[row][j];
21 }
22 return total;
23 }
24
25 int main()
26 {
27 const int COUNTRIES = 7;
28 const int MEDALS = 3;
29
30 string countries[] =
31  {
32  "Canada",
33  "China",
34  "Germany",
35  "Korea",
36  "Japan",
37  "Russia",
38  "United States"
39  };
```

```
40
41 int counts[COUNTRIES][MEDALS] =
42 { 43 { 1, 0, 1 }, 44 { 1, 1, 0 }, 45 { 0, 0, 1 },
46 { 1, 0, 0 }, 47 { 0, 1, 1 }, 48 { 0, 1, 1 }, 49 { 1,
1, 0 }
50 };
51
52 cout << " Country Gold Silver Bronze Total" <<
endl; 53
54 // Print countries, counts, and row totals
55 for (int i = 0; i < COUNTRIES; i++)
56 {
57 cout << setw(15) << countries[i];
58 // Process the ith row
59 for (int j = 0; j < MEDALS; j++)
60 {
61 cout << setw(8) << counts[i][j];
62 }
63 int total = row_total(counts, i);
64 cout << setw(8) << total << endl;
65 }
66
67 return 0;
68 }
```

Natija:

```
 M a m l a k a t  O l t i n  K u m u sh  B r o n z a  U m
u m i y
 Kanada  1  0  1  2
 Xitoy  1  1  0  2
```

```
Germaniya 0 0 1 1
Koreya 1 0 0 1
YAponiya 0 1 1 2
Rossiya 0 1 1 2 Qo‘shma SHtatlar 1 1 0 2
```

Nazorat savollari

1. C++da Massiv qanday ishlaydi?
2. Massivga kutubxona kerakmi?
3. Rekursiv funksiya nima?
4. Massiv elementlarini bilan massiv indexlarini farqi nimada?
5. For operatori funksiyada qanday ishlatiladi?
6. Matematik funksiyalar qanday ishlaydi?
7. Massiv elementlarini Funksiya parametrlarida uzatish nima uchun ishlatiladi?
8. Massivlarni qabday turlari mavjud?
9. Funksiya parametrlari orqali nima uzatiladi?

18-Ma’ruza. Vektorlar va ko’p o’lchovli massivlar bilan ishlash

Ma’ruza rejasi:

17.1 Vektor tushunchasi

17.2 Ko’p o’lchovli massivlar

17.3 Ko‘p o‘lchamli statik massivlar

Kalit so’zlar:, *ro’yxat, manzil, nolinchi ko’rchsatkich, tugun, adres olish &, bo’shatish, ko’rsatkich, virtual destruktor, xotira, xotira chiqishi, destruktor, toifani o’zlashtirish, resurslar chiqishi, a’zo destruktori.*

17.1 Vektor tushunchasi

Foydalanuvchi kiritmasidan qiymatlarni yig‘uvchi dasturni yozganingizda nechta qiymatlar bo‘lishini bilmaysiz. Afsuski, *dastur tuzilganda* jadval hajmi aniq bo‘lishi kerak.

Bo‘limda bu muammoni qisman to‘ldirilgan jadvallar qanday yo‘naltirishni ko‘rdingiz. biz quyidagi bo‘limlarda muhokama qiladigan bu vektor tuzilishi qulayroq echimni taklif etadi. Vektor qiymatlar ketma ketligini xuddi jadval qilgandek yig‘adi lekin uning hajmi o‘zgarishi mumkin.

Vektorni aniqlaganingizda burchak kronshteyndagi elementlar turini aniqlashtirasiz. Quyidagicha:

```
vector<double> values;
```

Boshlang‘ich hajmni erkin aniqlashingiz mumkin. Masalan, Bu erda boshlang‘ich hajmi 10 ga teng bo‘lgan vektorning ta’rifi:

```
vector<double> values(10);
```

agar vektorni boshlang‘ich hajmsiz aniqlasangiz uning hajmi 0 ga teng.

Jadvalning 0 hajmini aniqlashda hech qanday nuqta bo‘lmaganda boshlang‘ich hajmi 0 ga teng vektorlarga ega bo‘lish foydali va keyin ular keraklicha

o‘stiriladi. Dasturingizda vektorlardan foydalanish uchun siz vektor sarlavhasini ham kiritishingiz zarur

defining Vectors

```
vector<int> numbers(10);   10 butun vektor
 vector<string> names(3);   3 trosli vektor
vector<double> values;   0 hajmli vektor
vector<double> values();   Xato: Vektorni aniqlamadi.
```

Agar qo‘shimcha elementlar kerak bo‘lsa elementni vektorning oxiriga qo‘shish uchun push_back funksiyasini qo‘llaysiz. bunda uning hajmi 1 ga ortadi. Push_back funksiyas i quyidagi nuqtali notatsiya bilan chaqirishingiz shart bo‘lgan a’zo funksiyasidir:

```
values.push_back(37.5);
```

Bu chaqiruvdan so‘ng 14 shakldagi vektor qiymati 3 hajmga teng va qiymatlar

```
values[2] qiymat 37.5 ga teng.
```

Bo‘sh vektor bilan boshlash va push_back funksiyasi bilan uni to‘ldirish ommalashgan. Masalan,

```
vector<double> values; // Dastlabki bo‘sh
values.push_back(32); // endi qiymatla 1 hajmga va element esa 32 hajmga ega values.push_back(54); // endi qiymatla 2 hajmga va element esa 32,54 hajmga ega values.push_back(37.5); // endi qiymatla 3 hajmga va element esa 32,54,37.5 hajmga ega
```

push_back a’zo funksiyasi uchun boshqa bir foydalanish bu vektorni kiritma qiymatlari bilan to‘ldirish.

```
vector<double> values; // Dastlabki bo‘sh
double input;
while (cin >> input)
```

```
{
 values.push_back(input);
}
```

Bu kiritmassikl 6.2.10 bo‘limdagidan osonroq va oddiyroq ekanini qayd eting. Boshqa a’zo funksiyasi, pop_back, vektorning oxirgi elementini olib tashlash, uning xajmini bittaga kichiklashtirish. (15 shaklga qarang):

```
values.pop_back();
```

Vektorlar va funksiyalar

Siz boshqa qiymatlar kabi vektorlarni funksiya parametri sifatida ishlatishingiz mumkin. Masalan, quyida funksiya flouting member nuqtasi sonlari vektorining umumiy sonini hisoblaydi:

```
double sum(vector<double> values)
{
 double total = 0;
 for (int i = 0; i < values.size(); i++)
 {
  total = total + values[i];
 }
 return total;
}
```

Bu funksiya vektor elementlarini aylanib o‘tadi, lekin ularni o‘zgartirmaydi.

```
void multiply(vector<double>& values, double factor) // Note the &
{
 for (int i = 0; i < values.size(); i++)
 {
  values[i] = values[i] * factor;
```

```
    }
   }
```

Ba’zi programmistlar o‘zgartirilmaydigan vektor parametrlari uchun turg‘un yo‘nalishdan foydalanadilar. (Maxsus mavzu 5.2) masalan:

```
double sum(const vector<double>& values) // const & added for efficiency
```

funksiya vektorni qaytarishi mumkin. YAna vektorlar qolgan qiymatlardan boshqacha emas. funksiyada natijani o‘rnating va uni qaytaring. bu misolda, squares funksiyasi vektorni 0^2 up to $(n-1)^2$ qaytaradi:

```
vector<int> squares(int n)
{
 vector<int> result;
 for (int i = 0; i < n; i++)
 {
  result.push_back(i * i);
 }
 return result;
}
```

18.3 Ko‘p o‘lchamli statik massivlar

S++ tilida massivlar elementining turiga cheklovlar qo‘yil-maydi, lekin bu turlar chekli o‘lchamdagi ob’ektlarning turi bo‘lishi kerak. CHunki kompilyator massivning xotiradan qancha joy (bayt) egallashini hisoblay olishi kerak. Xususan, massiv komponentasi massiv bo‘lishi mumkin («vektorlar-vektori»), natijada *matritsa* deb nomlanuvchi ikki o‘lchamli massiv hosil bo‘ladi.

Agar matritsaning elementi ham vektor bo‘lsa, uch o‘lchamli massivlar - *kub* hosil bo‘ladi. SHu yo‘l bilan echilayotgan masalaga bog‘liq ravishda ixtiyoriy o‘lchamdagi massivlarni yaratish mumkin.

Ikki o‘lchamli massivning sintaksisi quyidagi ko‘rinishda bo‘ladi:

<tur> <nom> [<uzunlik >] [<uzunlik>]

Masalan, 10×20 o‘lchamli haqiqiy sonlar massivining e’loni:

float a[10][20];

E’lon qilingan A matritsani ko‘rinishi 7.2-rasmda keltirilgan.

$$
\begin{array}{cc}
 & \qquad\qquad\qquad\qquad\qquad\qquad\quad j \\
 & a_0: \quad (a_{0\,0}, \quad a_{0\,2} \quad \ldots \quad \ldots \quad a_{0\,18}, \quad a_{0\,19}), \\
 & a_1: \quad (a_{1\,0}, \quad a_{1\,1}, \quad \ldots \quad \ldots \quad a_{1\,18}, \quad a_{1\,19}), \\
 & \ldots \\
\mathbf{i} & a_i: \quad (\ldots, \quad \ldots, \quad \ldots \quad \mathbf{a_{ij}} \quad \ldots \quad \ldots, \quad \ldots), \\
 & \ldots \\
 & a_9: \quad (a_{9\,0}, \quad a_{9\,1}, \quad \ldots \quad \ldots \quad a_{9\,18}, \quad a_{9\,19}).
\end{array}
$$

7.2-rasm. Ikki o‘lchamli massivning xotiradagi joylashuvi

Endi adres nuqtai - nazaridan ko‘p o‘lchamli massiv element-lariga murojaat qilishni ko‘raylik. Quyidagi e’lonlar berilgan bo‘lsin:

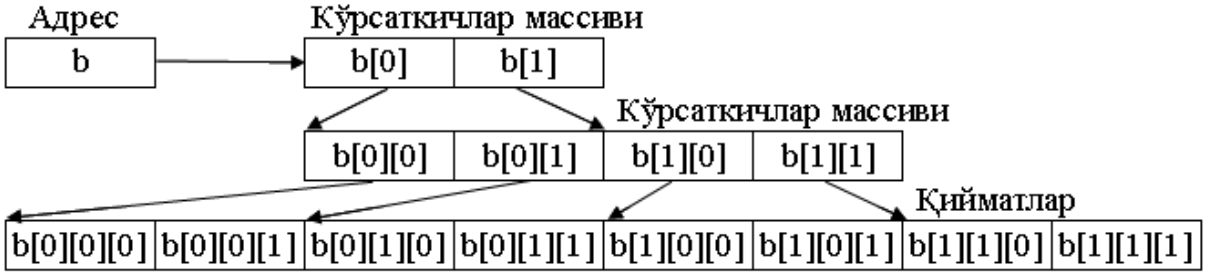
int a[3][2];

float b[2][2][2];

Birinchi e’londa ikki o‘lchamli massiv, ya’ni 2 satr va 3 ustundan iborat matritsa e’lon qilingan, ikkinchisida uch o‘lchamli - 3 ta 2x2 matritsadan iborat bo‘lgan massiv e’lon qilingan. Uning elementlariga murojaat sxemasi:

7.3-rasm. Ikki o‘lchamli massiv elementlariga murojaat

Bu erda a[i] ko‘rsatkichda i-chi satrning boshlang‘ich adresi joyla-shadi, massiv elementiga a[i][j] ko‘rinishidagi asosiy murojaatdan tashqari vositali murojaat qilish mumkin: *(*(a+i)+j) yoki *(a[i]+j).



7.3-rasm. Uch o‘lchamli massivning xotirada tashkil bo‘lishi

Massiv elementlariga murojaat qilish uchun nomdan keyin kvadrat qavsda har bir o‘lcham uchun indeks yozilishi kerak, masalan b[i][j][k]. Bu elementga vositali murojaat ham qilish mumkin va uning variantlari:

Mustaqil nazorat:

1. Besh boshlang‘ich sonni o‘z ichiga oluvchi vektorlar yaxlitligini aniqlang (2, 3, 5, 7, and 11). push_back funksiyasini element qo‘shish uchun ishlating.
2. o‘zingizni tekshiring 35 ga push_back ni ishlatmasdan javob bering.
3. Quyidagi jadvalda vektorlarning tarkibini aniqlang?

```
vector<string> names;
names.push_back("Ann");
names.push_back("Bob");
names.pop_back();
names.push_back("Cal");
```

4. Faraz qiling siz har besh minutda olingan obhavo o‘lchami to‘plamini saqlamoqchisiz. Vektordan foydalanasizmi yoki jadvaldan?
5. faraz qiling siz hafta kunlari nomlarini saqlamoqchisiz. Vektor yoki

7 string jadvalidan foydalanishingiz kerak?

6. Ikki vektorni to‘ldiruvchi, uchinchi vektorni olib keluvchi funksiyaning sarlavhasini toping. Funksiyani o‘zgartirmang.

7. Bir vektor elementini ikkinchisiga qo‘shuvchi, qisman to‘ldirilgan funksiyani faraz qiling.

```
void append(vector<double>__ target, vector<double>__ source)
{
 for (int i = 0; i < source.size(); i++)
 {
  target.push_back(source[i]);
 }
}
```

Nazorat savollari

19. C++da Massiv qanday ishlaydi?
20. Massivga kutubxona kerakmi?
21. Rekursiv funksiya nima?
22. Massiv elementlarini bilan massiv indexlarini farqi nimada?
23. For operatori funksiyada qanday ishlatiladi?
24. Matematik funksiyalar qanday ishlaydi?
25. Massiv elementlarini Funksiya parametrlarida uzatish nima uchun ishlatiladi?
26. Massivlarni qabday turlari mavjud?
27. Funksiya parametrlari orqali nima uzatiladi?
28. Massivning necha hil turi bor?

2-SEMESTR UCHUN

1-Ma’ruza. Ko`rsatkich

Ma’ruza rejasi:

1.1. Ko‘rsatkichga boshlang‘ich qiymat berish;

1.2. Ko‘rsatkich ustida amallar;

1.3. Adresni olish amali;

1.4. Ko‘rsatkichlar va adres oluvchi o‘zgaruvchilar funksiya parametri sifatida;

1.5. O‘zgaruvchan parametrli funksiyalar

Kalit so’zlar:*, ro’yxat, manzil, nolinchi ko’rchsatkich, tugun, adres olish &, bo’shatish, ko’rsatkich, virtual destruktor, xotira, xotira chiqishi, destruktor, toifani o’zlashtirish, resurslar chiqishi, a’zo destruktori.*

Ko‘rsatkichga boshlang‘ich qiymat berish

Programma matnida o‘zgaruvchi e’lon qilinganda, kompilyator o‘zgaruvchiga xotiradan joy ajratadi. Boshqacha aytganda, programma kodi xotiraga yuklanganda berilganlar uchun, ular joylashadigan segmentning boshiga nisbatan siljishini, ya’ni nisbiy adresini aniqlaydi va ob’ekt kod hosil qilishda o‘zgaruvchi uchragan joyga uning adresini joylashtiradi.

Umuman olganda, programmadagi o‘zgarmaslar, o‘zgaruvchilar, funksiyalar va sinf ob’ektlar adreslarini xotiraning alohida joyida saqlash va ular ustidan amallar bajarish mumkin. Qiymat-lari adres bo‘lgan o‘zgaruvchilarga *ko‘rsatkich o‘zgaruvchilar* deyiladi.

Ko‘rsatkich uch xil turda bo‘lishi mumkin:

- birorta ob’ektga, xususan o‘zgaruvchiga ko‘rsatkich;
- funksiyaga ko‘rsatkich;
- void ko‘rsatkich.

Ko‘rsatkichning bu xususiyatlari uning qabul qilishi mumkin bo‘lgan qiymatlarida farqlanadi.

Ko‘rsatkich albatta birorta turga bog‘langan bo‘lishi kerak, ya’ni u ko‘rsatgan adresda qandaydir qiymat joylanishi mumkin va bu qiymatning xotirada qancha joy egallashi oldindan ma’lum bo‘lishi shart.

Funksiyaga ko‘rsatkich. Funksiyaga ko‘rsatkich programma joy-lashgan xotiradagi funksiya kodining boshlang‘ich adresini ko‘rsa-tadi, ya’ni funksiya chaqirilganda boshqaruv ayni shu adresga uzatila-di. Ko‘rsatkich orqali funksiyani oddiy yoki vositali chaqirish amalga oshirish mumkin. Bunda funksiya uning nomi bo‘yicha emas, balki funksiyaga ko‘rsatuvchi o‘zgaruvchi orqali chaqiriladi. Funksiyani boshqa funksiyaga argument sifatida uzatish ham funksiya ko‘rsatkichi orqali bajariladi. Funksiyaga ko‘rsatkichning yozilish sintaksisi quyidagicha:

<tur> (* <nom>) (<parametrlar ro‘yxati>);

Bunda <tur>- funksiya qaytaruvchi qiymat turi; *<nom> - ko‘rsatkich o‘zgaruvchining nomi; <parametrlar ro‘yxati> - funksiya parametr-larining yoki ularning turlarining ro‘yxati.

Masalan:

int (*fun)(float,float);

Bu erda butun son turida qiymat qaytaradigan fun nomidagi funksiyaga ko‘rsatkich e’lon qilingan va u ikkita haqiqiy turdagi parametrlarga ega.

Masala. Berilgan butun n=100 va a,b - haqiqiy sonlar uchun $f_1(x)=5\sin(3x)+x$, $f_2(x)=\cos(x)$ va $f_3(x)=x^2+1$ funksiyalar uchun $\int_a^b f(x)dx$ integralini to‘g‘ri to‘rtburchaklar formulasi bilan taqriban hisoblansin:

$$\int_a^b f(x)dx \approx h[f(x_1)+f(x_2)+...+f(x_n)],$$

bu erda $h=\frac{b-a}{n}$, $x_i=a+ih-h/2, i=1..n$.

Programma bosh funksiya, integral hisoblash va ikkita matematik funksiyalar - $f_1(x)$ va $f_3(x)$ uchun aniqlangan funksiyalardan tashkil topadi, $f_2(x)=\cos(x)$ funksiyaning adresi «math.h» sarlavha faylidan olinadi. Integral hisoblash funksiyasiga ko‘rsatkich orqali integrali hisoblanadigan funksiya adresi, a va b - integral chegaralari qiymatlari uzatiladi. Oraliqni bo‘lishlar soni - n global o‘zgarmas qilib e’lon qilinadi.

```
#include <iostream.h>
#include <math.h>
const int n=100;
double f1(double x){return 5*sin(3*x)+x;}
double f3(double x){return x*x+1;}
double Integral(double(*f)(double),double a,double
b)
{
 double x,s=0;
 double h=(b-a)/n;
 x=a-h/2;
 for(int i=1;i<=n; i++) s+=f(x+=h);
 s*=h;
 return s;
}
int main()
{
 double a,b;
 int menu;
 while(1)
 {
  cout<<"\nIsh regimini tanlang:\n";
  cout<<"1:f1(x)=5*sin(3*x)+x integralini\
```

```
        hisoblash\n";
        cout<<"2:f2(x)=cos(x) integralini hisoblash\n";
        cout<<"3:f3(x)=x^2+1 integralini hisoblash\n";
        cout<<"0:Programmadan chiqish\n";
        do
        {
         cout<<" Ish regimi-> ";
         cin>>menu;
        }
         while (menu<0 || menu>3);
        if(!menu)break;
        cout<<"Integral oralig'ining quyi chegarasi a=";
        cin>>a;
        cout<<"Integral oralig'ining yuqori chegarasi
b=";
        cin>>b;
        cout<<"Funksiya integrali S=";
        switch (menu)
        {
         case 1 : cout<<Integral(f1,a,b)<<endl; break;
         case 2 : cout<<Integral(cos,a,b)<<endl; break;
         case 3 : cout<<Integral(f3,a,b)<<endl;
        }
       }
       return 0;
      }
```

Programmaning ishi cheksiz takrorlash operatori tanasini bajarishdan iborat. Takrorlash tanasida foydalanuvchiga ish reji-mini tanlash bo‘yicha menyu taklif qilinadi:

Ish regimini tanlang:

1: f1(x)=5*sin(3*x)+x integralini hisoblash

2: f2(x)=cos(x) integralini hisoblash

3: f3(x)=x^2+1 integralini hisoblash

0: Programmadan chiqish

Ish regimi->

Ko‘rsatkich ustida amallar

Foydalanuvchi 0 va 3 oralig‘idagi butun sonni kiritishi kerak. Agar kiritilgan son (menu o‘zgaruvchi qiymati) 0 bo‘lsa, break opera-tori yordamida takrorlashdan, keyin programmadan chiqiladi. Agar menu qiymati 1 va 3 oralig‘ida bo‘lsa, integralning quyi va yuqori chegaralarini kiritish so‘raladi, hamda Integral() funksiyasi mos funksiya adresi bilan chaqiriladi va natija chop etiladi. SHunga e’tibor berish kerakki, integral chegaralarining qiymatlarini to‘g‘ri kiritilishiga foydalanuvchi javobgar.

Ob’ektga ko‘rsatkich. Biror ob’ektga ko‘rsatkich (shu jumladan o‘zgaruvchiga). Bunday ko‘rsatkichda ma’lum turdagi (tayanch yoki hosila-viy turdagi) berilganlarning xotiradagi adresi joylashadi. Ob’ektga ko‘rsatkich quyidagicha e’lon qilinadi:

<tur> *<nom>;

Bu erda <tur> - ko‘rsatkich aniqlaydigan adresdagi qiymatning turi, <nom> - ob’ekt nomi (identifikator). Agar bir turda bir nechta ko‘rsatkichlar e’lon qilinadigan bo‘lsa, har bir ko‘rsatkich uchun ‘*’ belgisi qo‘yilishi shart:

```
int *i, j,*k;
float x,*y,*z;
```

Keltirilgan misolda i va k - butun turdagi ko‘rsatkichlar va j - butun turdagi o‘zgaruvchi, ikkinchi operatorda x - haqiqiy o‘zgaruvchi va y,z - haqiqiy turdagi ko‘rsatkichlar e’lon qilingan.

void ko‘rsatkich. Bu ko‘rsatkich ob’ekt turi oldindan noma’lum bo‘lganda ishlatiladi. void ko‘rsatkichining muhim afzalliklaridan biri - unga har qanday turdagi ko‘rsatkich qiymatini yuklash mumkin-ligidir. void ko‘rsatkich adresidagi qiymatni ishlatishdan oldin, uni aniq bir turga oshkor ravishda keltirish kerak bo‘ladi. void ko‘rsatkichni e’lon qilish kuyidagicha bo‘ladi:

void *<nom>;

Ko‘rsatkichning o‘zi o‘zgarmas yoki o‘zgaruvchan bo‘lishi va o‘zgarmas yoki o‘zgaruvchilar adresiga ko‘rsatishi mumkin, masalan:

```
int i; // butun o‘zgaruvchi
const int ci=1; // butun o‘zgarmas
int * pi; // butun o‘zgaruvchiga ko‘rsatkich
const int *pci; // butun o‘zgarmasga ko‘rsatkich
int *const cp=&i;//butun o‘zgaruvchiga o‘zgarmas
                 //ko‘rsatkich
const int*const cpc=&ci; // butun o‘zgarmasga
o‘zgarmas
// ko‘rsatkich
```

Misollardan ko‘rinib turibdiki, ‘*’ va ko‘rsatkich nomi ora-sida turgan const modifikatori faqat ko‘rsatkichning o‘ziga tegishli hisoblanadi va uni o‘zgartirish mumkin emasligini bildiradi, ‘*’ belgisidan chapda turgan const esa ko‘rsatilgan adresdagi qiymat o‘zgarmas ekanligini bildiradi.

Ko‘rsatkichga qiymatni berish uchun ‘&’ - adresni olish amali ishlatiladi.

Ko‘rsatkich o‘zgaruvchilarining amal qilish sohasi, yashash davri va ko‘rinish sohasi umumiy qoidalarga bo‘ysunadi.

Ko‘rsatkichga boshlang‘ich qiymat berish

Ko‘rsatkichlar ko‘pincha dinamik xotira (boshqacha nomi «uyum» yoki «heap») bilan bog‘liq holda ishlatiladi. Xotiraning dinamik deyilishiga sabab, bu sohadagi bo‘sh xotira programma ishlash jarayonida, kerakli paytida ajratib olinadi va zarurat qolmaganida qaytariladi (bo‘shatiladi). Keyinchalik, bu xotira bo‘lagi

programma tomonidan boshqa maqsadda yana ishlatilishi mumkin. Dinamik xotiraga faqat ko‘rsatkichlar yordamida murojaat qilish mumkin. Bunday o‘zgaruvchilar *dinamik o‘zgaruvchilar* deyiladi va ularni yashash vaqti yaratilgan nuqtadan boshlab programma oxirigacha yoki oshkor ravishda yo‘qotilgan (bog‘langan xotira bo‘shatilgan) joygacha bo‘ladi.

Ko‘rsatkichlarni e’lon qilishda unga boshlang‘ich qiymatlar berish mumkin. Boshlang‘ich qiymat (initsializator) ko‘rsatkich nomi-dan so‘ng yoki qavs ichida yoki ‘=’ belgidan keyin beriladi. Boshlang‘ich qiymatlar quyidagi usullar bilan berilishi mumkin:

I. Ko‘rsatkichga mavjud bo‘lgan ob’ektning adresini berish:

a) adresni olish amal orqali:

```
int i=5,k=4; // butun o‘zgaruvchilar
int *p=&i;  // p ko‘rsatkichga i o‘zgaruvchining
// adresi yoziladi
int *p1(&k); // p1 ko‘rsatkichga k o‘zgaruvchining
// adresi yoziladi
```

b) boshqa, initsializatsiyalangan ko‘rsatkich qiymatini berish:

```
int * r=p; // p oldin e’lon qilingan va qiymatga
ega
                // bo‘lgan ko‘rsatkich
```

v) massiv yoki funksiya nomini berish:

```
int b[10];  // massivni e’lon qilish
int *t=b;  // massivning boshlang‘ich adresini
berish
void f(int a){/* … */} // funksiyani aniqlash
void (*pf)(int);// funksiyaga ko‘rsatkichni e’lon
qilish
pf=f; // funksiya adresini ko‘rsatkichga berish
```

II. Oshkor ravishda xotiraning absolyut adresini berish:

```
char *vp = (char *)0xB8000000;
```

Bunda 0xB8000000 - o‘n oltilik o‘zgarmas son va (char*) - turga keltirish amali bo‘lib, u vp o‘zgaruvchisini xotiraning absolyut adresidagi baytlarni char sifatida qayta ishlovchi ko‘rsatkich turiga aylantirilishini anglatadi.

III. Bo‘sh qiymat berish:

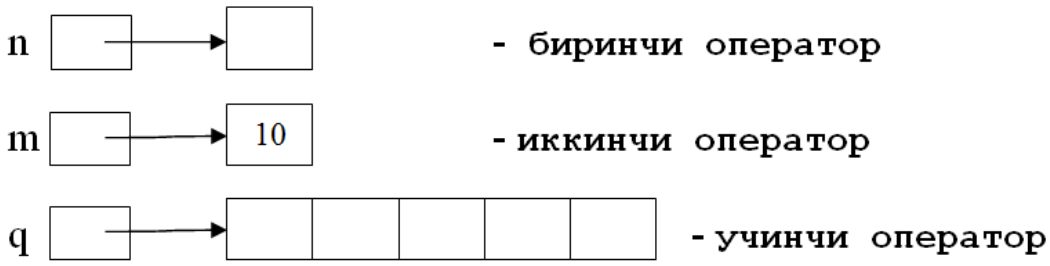
```
int *suxx=NULL;
int *r=0;
```

Birinchi satrda maxsus NULL o‘zgarmasi ishlatilgan, ikkinchi satrda 0 qiymat ishlatilgan. Ikkala holda ham ko‘rsatkich hech qanday ob’ektga murojaat qilmaydi. Bo‘sh ko‘rsatkich asosan ko‘rsatkichni aniq bir ob’ektga ko‘rsatayotgan yoki yo‘qligini aniqlash uchun ishla-tiladi.

IV. Dinamik xotirada new amali bilan joy ajratish va uni adresini ko‘rsatkichga berish:

```
int * n=new int;          // birinchi operator
int * m=new int(10); // ikkinchi operator
int * q=new int[5];  // uchinchi operator
```

Birinchi operatorda new amali yordamida dinamik xotirada int uchun etarli joy ajratib olinib, uning adresi n ko‘rsatkichga yuklanadi. Ko‘rsatkichning o‘zi uchun joy kompilyasiya vaqtida ajra-tiladi.



6.1-rasm. Dinamik xotiradan joy ajratish

Ikkinchi operatorda joy ajratishdan tashqari m adresiga boshlang‘ich qiymat - 10 sonini joylashtiradi.

Uchinchi operatorda int turidagi 5 element uchun joy ajra-tilgan va uning boshlang‘ich adresi q ko‘rsatkichga berilayapti.

Xotira new amali bilan ajratilgan bo‘lsa, u delete amali bilan bo‘shatilishi kerak. YUqoridagi dinamik o‘zgaruvchilar bilan bog‘lan-gan xotira quyidagicha bo‘shatiladi:

```
delete n; delete m; delete[]q;
```

Agarda xotira new[] amali bilan ajratilgan bo‘lsa, uni bo‘shatish uchun delete [] amalini o‘lchovi ko‘rsatilmagan holda qo‘llash kerak.

Xotira bo‘shatilganligiga qaramasdan ko‘rsatkichni o‘zini keyinchalik qayta ishlatish mumkin.

Ko‘rsatkich ustida amallar

Ko‘rsatkich ustida quyidagi amallar bajarilishi mumkin:

1. ob’ektga vositali murojaat qilish amali;
2. qiymat berish amali;
3. ko‘rsatkichga o‘zgarmas qiymatni qo‘shish amali;
4. ayirish amali;
5. inkrement va dekrement amallari;
6. solishtirish amali;
7. turga keltirish amali.

Vositali murojaat qilish amali ko‘rsatkichdagi adres bo‘yicha joylashgan qiymatni olish yoki qiymat berish uchun ishlatiladi:

```
shar a;           // char turidagi o‘zgaruvchi
e’loni.
shar *p=new char; // Ko‘rsatkichni e’lon qilib,unga
// dinamik xotiradan ajratilgan
// xotiraning adresini berish
*p=’b’;  // p adresiga qiymat joylashtirish
a=*p; // a o‘zgaruvchisiga p adresidagi qiymatni
berish
```

SHuni qayd qilib o‘tish kerakki, xotiraning aniq bir joyidagi adresni bir paytning o‘zida bir nechta va har xil turdagi ko‘rsatkich-larga berish mumkin va ular orqali murojaat qilinganda berilgan-ning har xil turdagi qiymatlarini olish mumkin:

```
unsigned long int A=0Xcc77ffaa;
unsigned short int * pint=(unsigned short int*)&A;
unsigned char* pchar=(unsigned char*)&A;
cout<<hex<<A<<' '<<hex<<*pint<<'
'<<hex<<(int)*pchar;
```

Ekranga har xil qiymatlar chop etiladi:

cc77ffaa ffaa aa

O‘zgaruvchilar bitta adresda joylashgan holda yaxlit qiymatning turli bo‘laklarini o‘zlashtiradi. Bunda, bir baytdan katta joy egal-lagan son qiymatining xotirada «teskari» joylashishi inobatga olinishi kerak.

Agar har xil turdagi ko‘rsatkichlarga qiymatlar berilsa, albat-ta turga keltirish amalidan foydalanish kerak:

```
int n=5;
float x=1.0;
int *pi=&n;
float *px=&x;
void *p;
int *r,*r1;
px=(float *)&n;
p=px;
r=(int *)p;
r1=pi;
```

Ko‘rsatkich turini void turiga keltirish amalda ma’noga ega emas. Xuddi shunday, turlari bir xil bo‘lgan ko‘rsatkichlar uchun turni keltirish amalini bajarishga hojat yo‘q.

Ko‘rsatkich ustidan bajariladigan arifmetik amallarda avto-matik ravishda turlarning o‘lchami hisobga olinadi.

Arifmetik amallar faqat bir xil turdagi ko‘rsatkichlar ustidan bajariladi va ular asosan, massiv tuzilmalariga ko‘rsatkich-lar ustida bajariladi.

Inkrement amali ko‘rsatkichni massivning keyingi elementiga, dekrement esa aksincha, bitta oldingi elementining adresiga ko‘chiradi. Bunda ko‘rsatkichning qiymati sizeof(<massiv elementi-ning turi>) qiymatiga o‘zgaradi. Agar ko‘rsatkich k o‘zgarmas qiymatga oshirilsa yoki kamaytirilsa, uning qiymati k*sizeof(<massiv elementining turi>) kattalikka o‘zgaradi.

Masalan:

```
short int *p=new short[5];
long * q=new long [5];
p++;     // p qiymati 2 oshadi
q++;     // q qiymati 4 ga oshadi
q+=3;    // q qiymati 3*4=12 oshadi
```

Ko‘rsatkichlarning ayirmasi deb, ular ayirmasining tur o‘lcha-miga bo‘linishiga aytiladi. Ko‘rsatkichlarni o‘zaro qo‘shish mumkin emas.

Adresni olish amali

Adresni olish quyidagicha e’lon qilinadi:

<tur> & <nom>;

Bu erda <tur> - adresi olinadigan qiymatning turi, <nom>- adres oluvchi o‘zgaruvchi nomi. O‘rtadagi ‘&’ belgisiga *adresni olish amali* deyiladi.

Bu ko‘rinishda e’lon qilingan o‘zgaruvchi shu turdagi o‘zgaruvchining sinonimi deb qaraladi. Adresni olish amali orqali bitta o‘zgaruvchiga har xil nom bilan murojaat qilish mumkin bo‘ladi.

Misol:

```
int kol;
```

```
int & pal=kol;   // pal murojaati, u kol
// o‘zgaruvchisining alьternativ nomi
const char & cr=’\n’; // cr - o‘zgarmasga murojaat
```

Adresni olish amalini ishlatishda quyidagi qoidalarga rioya qilish kerak: adres oluvchi o‘zgaruvchi funksiya parametri sifatida ishlatilgan yoki extern bilan tavsiflangan yoki sinf maydoniga murojaat qilingandan holatlardan tashqari barcha holatlarda boshlang‘ich qiymatga ega bo‘lishi kerak.

Adresni olish amali asosan funksiyalarda adres orqali uzatiluvchi parametrlar sifatida ishlatiladi.

Adres oluvchi o‘zgaruvchining ko‘rsatkichdan farqi shundaki, u alohida xotirani egallamaydi va faqat o‘z qiymati bo‘lgan o‘zgaruvchining boshqa nomi sifatida ishlatiladi.

Ko‘rsatkichlar va adres oluvchi o‘zgaruvchilar funksiya parametri sifatida

Funksiya prototipida yoki aniqlanish sarlavhasida ko‘rsatilgan parametrlar *formal parametrlar* deyiladi, funksiya chaqirishida ko‘rsatilgan argumentlarga *faktik parametrlar* deyiladi.

Funksiya chaqirilishida faktik parametrning turi mos o‘rindagi formal parametr turiga to‘g‘ri kelmasa yoki shu turga keltirishning iloji bo‘lmasa kompilyasiya xatosi ro‘y beradi.

Faktik parametrlarni funksiyaga ikki xil usul bilan uzatish mumkin: *qiymati* yoki *adresi* bilan.

Funksiya chaqirilishida argument qiymat bilan uzatilganda, argument yoki uning o‘rnidagi kelgan ifoda qiymati va boshqa argu-mentlarning nusxasi (qiymatlari) stek xotirasiga yoziladi. Funksiya faqat shu nusxalar bilan amal qiladi, kerak bo‘lsa bu nusxalarga o‘zgartirishlar qilinishi mumkin, lekin bu o‘zgarishlar argument-ning o‘ziga ta’sir qilmaydi, chunki funksiya o‘z ishini tugatishi bilan nusxalar o‘chiriladi (stek tozalanadi).

Agar parametr adres bilan uzatilsa, stekka adres nusxasi yoziladi va xuddi shu adres bo‘yicha qiymatlar o‘qiladi (yoziladi). Funksiya o‘z ishini tugatgandan keyin

shu adres bo‘yicha qilingan o‘zgarishlar saqlanib qolinadi va bu qiymatlarni boshqa funksiyalar ishlatishi mumkin.

Argument qiymat bilan uzatilishi uchun mos formal parametr sifatida o‘zgaruvchini turi va nomi yoziladi. Funksiya chaqirilishida mos argument sifatida o‘zgaruvchining nomi yoki ifoda bo‘lishi mumkin.

Faktik parametr adres bilan uzatilganda unga mos keluvchi formal parametrni ikki xil usul bilan yozish mumkin: *ko‘rsatkich orqali* yoki *adresni oluvchi parametrlar orqali*. Ko‘rsatkich orqali yozilganda formal parametr turidan keyin ‘*’ belgisi yoziladi, mos argumentda esa o‘zgaruvchining adresi (& amal orqali) yoki massiv nomi, yoki funksiya nomi bo‘lishi mumkin. Adresni olish amali orqali parametr uzatishda formal parametrda turidan keyin ‘&’ belgisi yoziladi va funksiya chaqirilishida mos argument sifatida o‘zgaruvchi nomi keladi.

Misol:

```
#include <iostream.h>
void f(int,int*,int &)
void main()
{
 int i=1,j=2,k=3;
 cout<<i<<” “<<j<<” “<<k;
 f(i,&j,k);
 cout<<i<<” “<<j<<” “<<k;
}
void f(int i;int *j;int &k)
{
 i++;
 (*j)++;
 k++;
 *j=i+k;
 k=*j+i;
```

```
}
```

Programma ishlashi natijasida ekranga quyidagi qiymatlar chop qilinadi:

1 2 3

1 6 8

Bu misolda birinchi parametr i qiymat bilan uzatiladi (“int i"). Uning qiymati funksiya ichida o‘zgaradi, lekin tashqaridagi i o‘zga-ruvchisining qiymati o‘zgarmaydi. Ikkinchi parametrni ko‘rsatkich orqali adresi bilan uzatilishi talab qilinadi (“int *j”), adresni uzatish uchun ‘&’- adresni olish amali ishlatilgan (“&j”). Funksiya tanasida argument adresidan qiymat olish uchun ‘*’- qiymat olish amali qo‘llanilgan. Uchinchi parametrda murojaat orqali (“&k”) argumentning adresi uzatish ko‘zda tutilgan. Bu holda funksiya chaqirilishida mos argument o‘rnida o‘zgaruvchi nomi turadi, funksiya ichida esa qiymat olish amalini ishlatishning hojati yo‘q. Funksiya ishlash natijasidagi qiymatlarni argumentlar ro‘yxati orqali olish qulay va tushunarli usul hisoblanadi.

Agar funksiya ichida adres bilan uzatiladigan parametr qiymati o‘zgarmasdan qolishi zarur bo‘lsa, bu parametr const modifikator bilan yozilishi kerak:

fun(int n,const char*str);

Agarda funksiyani chaqirishda argumentlar faqat nomlari bilan berilgan bo‘lsa, kelishuv bo‘yicha massivlar va funksiyalar adresi bilan, qolgan turdagi parametrlar qiymatlari bilan uzatilgan deb hisoblanadi.

Misol tariqasida diskriminantni hisoblash usuli yordamida $ax^2+bx+c=0$ ko‘rinishidagi kvadrat tenglama ildizlarini funksiya parametrlari vositasida olish masalasini ko‘raylik.

```
    #include <iostream.h>
#include <math.h>
    int Kvadrat_Ildiz(float a,float b,float c,
 float & x1, float & x2)
    {
     float D;
```

```
          D=b*b-4*a*c;
          if(D<0) return 0;
          if(D==0)
          {
            x1=x2=-b/(2*a);
           return 1;
          }
          else
          {
       x1=(-b+sqrt(D))/(2*a);
            x2=(-b-sqrt(D))/(2*a);
            return 2;
          }
         }
    int main()
         {
          float a,b,c,D,x1,x2;
          cout<<"ax^2+bx+c=0 tenglama ildizini topish.
";
          cout<<"\n a - koeffisiyentni kiriting: ";
cin>>a;
          cout<<"\n b - koeffisiyentni kiriting: ";
cin>>b;
          cout<<"\n c - koeffisiyentni kiriting: ";
cin>>c;
          switch (Kvadrat_Ildiz(a,b,c,x1,x2))
          {
      case 0: cout<<"Tenglama haqiqiy ildizga ega
emas!";
                  break;
```

```
            case 1: cout <<"Tenglama yagona ildizga ega:
";
                    cout<<"\n x= "<<x1;
                    break;
            default:cout<<"Tenglama ikkita ildizga ega:
";
                    cout<<"\nx1= "<<x1;
                    cout<<"\nx2= "<<x2;
          }
          return 0;
      }
```

Programmadagi Kvadrat_Ildiz() funksiyasi kvadrat tenglama ildizini hisoblaydi. Uning qaytaradigan qiymati tenglamaning nechta ildizi borligini anglatadi. Agar tenglamaning haqiqiy ildizi mavjud bo‘lmasa (D<0), funksiya 0 qiymatini qaytaradi. Agar D=0 bo‘lsa, funksiya 1 qiymatini qaytaradi. Agar D>0 bo‘lsa funksiya 2 qiymatini qaytaradi. Mavjud ildizlar - x1 va x2 adres oluvchi parametrlarda qaytariladi.

1.4. Ko‘rsatkichlar va adres oluvchi o‘zgaruvchilar funksiya parametri sifatida

S++ tilida parametrlar soni noma’lum bo‘lgan funksiyalarni ham ishlatish mumkin. Bundan tashqari ularning turlari ham noma’-lum bo‘lishi mumkin. Parametrlar soni va turi funksiyani chaqirish-dagi argumentlar soni va ularning turiga qarab aniqlanadi. Bunday funksiyalar sarlavhasi quyidagi formatda yoziladi:

<funksiya turi> <funksiya nomi> (<oshkor parametrlar ro‘yxati>, …)

Bu erda <oshkor parametrlar ro‘yxati> - oshkor ravishda yozilgan parametrlar nomi va turi. Bu parametrlar *majburiy parametrlar* deyiladi. Bunday parametrlardan kamida bittasi bo‘lishi shart. Qolgan parametrlar soni va turi noma’lum hisoblanadi. Ularni aniqlash va ishlatish to‘la ravishda programma tuzuvchi zimmasiga yuklanadi.

O‘zgaruvchan sondagi parametrlarni tashkil qilish usuli umuman olganda ikkita:

1-usul. Parametrlar ro‘yxati oxirida yana bir maxsus parametr yoziladi va uning qiymati parametrlar tugaganligini bildiradi. Kompilyator tomonidan funksiya tanasida parametrlar birma-bir aniqlashtiriladi. Barcha parametrlar turi oxirgi maxsus parametr turi bilan ustma-ust tushadi deb hisob-lanadi;

2-usul. Birorta maxsus parametr sifatida noma’lum para-metrlar soni kiritiladi va unga qarab parametrlar soni aniq-lanadi.

Ikkala usulda ham parametrlarga murojaat qilish uchun ko‘rsat-kichlar ishlatiladi. Misollar keltiramiz.

1 - usul:

```
#include <iostream.h>
float Sonlar_kupaytmasi(float arg,...)
{
 float p=1.0;
float *ptr=&arg;
if(*prt==0.0) return 0.0;
for(;*prt;prt++)p*=*prt;
return p;
}
void main()
{
cout<<Sonlar_kupaytmasi(2e0,3e0,4e0,0e0)<<’\n’;
cout<<Sonlar_kupaytmasi(1.0,2.0,3.0,10.0,8.0,0.0);
}
```

Natija:

24

480

2 - usul:

```
#include <iostream.h>
int Yigindi(int,...);
void main()
{
 cout<<"\nYigindi(2,6,4)="<<Yigindi(2,6,4);
 cout<<"\nYigindi(6,1,2,3,4,5,6)="
 cout<<Yigindi(6,1,2,3,4,5,6);
}
int Yigindi(int k,...)
{
 int *ptr=&k
 int s=0;
 for(;k;k--)s+=*(++ptr);
 return s;
}
```

Natija:

Yigindi(2,6,4)=10

Yigindi(6,1,2,3,4,5,6)=21

Ikkala misolda ham noma'lum parametrlar berilgan maxsus parametr turini qabul qilgan. Har xil turdagi parametrlarni ishla-tish uchun turni aniqlaydigan parametr kiritish kerak:

```
#include <iostream.h>
float Summa(char,int,…);
void main()
{
 cout<<Summa('i',3,10,20,30);
 cout<<Summa('f',3,10.0,20.0,5.0);
 cout<<Summa('d',3,10,20,30);
}
```

```
int Summa(char z,int k,…)
{
 switch(z)
 {
  case ’i’:
  {
   int *ptr=&k+1; int s=0;
   for (;k--;ptr++) s+=*(ptr);
   return (float)s;
   }
  case ’f’:
  {
   float*ptr=(float *)(&k+1); float s=0.0;
   for (;k--;ptr++) s+=*(ptr);
   return s;
  }
  default:
  {
   cout<<”\n parametr hato berilgan”;
   return 9999999.0;
       }
}
}
```

1.5. O‘zgaruvchan parametrli funksiyalar

YUqorida keltirilgan misolda noma’lum parametrlarni turini aniqlash masalasi kompilyator tomonidan emas, balki prog-ramma tuzuvchisi tomonidan hal qilingan.

Quyidagi dastur ko’rsatkichning xatti harakatini aniqlab beradi. Biz bir hil qaytarishni amalga oshiramiz ikki marta, lekin hisob ko’rsatkichi uchun turi xil qiymatlar bilan bir hil hisob o’zgartiriladi.

ch07/accounts.cpp

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main()
6 {
7 double harrys_account = 0;
8 double joint_account = 2000;
9 double* account_pointer = &harrys_account;
10
11 *account_pointer = 1000; // Dastlabki jamg'arma
12
13 *account_pointer = *account_pointer - 100; //
qaytarish $100 ga 14 cout << "Balance: " <<
*account_pointer <<endl;//chiquvchi balans 15
16 / /  korsatkich qiymatini o'zgartirish
17 account_pointer = &joint_account;
18
19 / /  bitte yozuv boshqa boshqa hisob raqamlarini
ifodalaydi
20 *account_pointer = *account_pointer - 100; //
qaytish $100
21 cout << "Balance: " <<
*account_pointer<<endl;//chiquvchi balans 15
23 return 0;
```

```
24  }
```

Natija:

Balance: 900

Balance: 1900

1. Bu to'plam operatorlarini ko'rib chiqamiz.

```
int a = 1;
 int b = 2;
 int* p = &a;
 cout << *p << endl;
 p = &b;
 cout << *p << endl;
```

2. Bu to'plam operatorlarini ko'rib chiqamiz.

```
int a = 1;
      int b = 2;
```

Ma'lumotning qaysidir ko'rsatgichlari uchun inkor ko'rsatgichlari Ko'rsatgichning xotira manzillari — xotirada joylashgan qator qiymatlarini aytadi. Bu xulosaning umumiy xatosi bo'ladi

```
double* account_pointer = &joint_account;
account_pointer = 1000; // Error
```

Belgilash bayonotida 1000 hisobot balansi belgilanmagan. Buning o'rniga, faqat o'zgaruvchan qo'shma hisob ko'rsatgichlarini tasvirlab programma zimmasiga ishora markerini sozlaydi va faqatgina 1000 manzilni emas . Ko'p kompilyatorlar bu topshiriq uchun xato xisobot beradi.

Aslida o'zgaruvchi, foydalanishi uchun*account_pointer ishlatiladi:

```
*account_pointer = 1000; // OK
```

Xar bir ko'rsatkich o'zgaruvchilari uchun alohida tarifi bilan ishlatish

Bu kabi bir nechta o'zgaruvchilarni aniqlashda C++ huquqiy hisoblanadi:

```
int i = 0, j = 1;
```

Bu usulda ko'rsatgichlardan foydalanish, chalkashliklarga olib keladi:

```
double* p, q;
```

* faqat boshlang'ich o'zgaruvchilar bilan bog'laydi. Yani, p qo'shaloq * qiymat bo'lib, q

unga qo'shaloq qiymat bo'ladi. Kelib chiqishi mumkin bo'lgan tartibsizlikni oldini olish uchun ko'rsatgichlarni alohida aniqlash kerak:

```
double* p; double* q;
```

Nazorat savollari

1. Ko’rsatgichlar nima?
2. Nimaga ko’rsatgichlar qulay emas?
3. Kompyuter o’yinlari yaratish uchun dasturchi qay darajada malakali bo’lishi kerak deb o’ylaysiz?
4. Joriy bajarilayotgan ko’rsatgichlar qayerda saqlanadi?
5. Kompyuterni arifmetik amallarni bajaradi?
6. kompilyator apparat ta’minotmi yoki dasturiymi?
7. Qasyi jihati uchun C++ foydalanuvchilarga juda muhim

2-Ma’ruza. Ko`rsatkich va massiv

Ma’ruza rejasi:

2.1 Massivlar ko'rsatkichlar o'rnida;

2.2 Arifmetik ko'rsatkich;

2.3 Ko'rsatkichlar - massivning o'zgaruvchanligi.

Kalit so’zlar:*, ro’yxat, manzil, nolinchi ko’rchsatkich, tugun, adres olish &, bo’shatish, ko’rsatkich, virtual destruktor, xotira, xotira chiqishi, destruktor, toifani o’zlashtirish, resurslar chiqishi, a’zo destruktori.*

2.1. Massivlar ko'rsatkichlar o'rnida

Ko'rsatkichlar massivlarning o'ziga xosligini tushunish uchun zarur. Keyingi bo'limlarda massiv va ko'rsatkichlar C++ da bir biriga qanday bog'liqligini ko'rsatamiz

Massivning bayonotini oling:

```
int a[10];
```

Bilganingizdek, a[3] massivni bildiradi. Qavssiz massivning nomi

ko'rsatkichning boshlang'ich elementini bildiradi (3-tasvirnga qarang). Ko'rsatkichni o'zgaruvchining ichida tasvirga olishingiz mumkin: int* p = a; // Now p points to a[0]

Shuningde, ko'rsatkichning o'rniga massivning nomini ishlatishingiz mumkin.

```
cout << *a;
```

Bu tasdiqlash pastdagi bilan bir xil ta'sirga ega

```
cout << a[0];
```

2.2 Arifmetik ko'rsatkich

Massivlarga o'zgargan ko'rsatkichlar arifmetik ko'rsatkichlarni qo'llashadi. Boshqa massiv joylashuvini topish uchun ko'rsatkichga butun son qo'shishingiz mumkin. Masalan, p massivning boshlanishini ko'rsatadi:

```
double a[10];
```

double* p = a;

So'ng

p + 3 ifodasi

3 indeksli massivning ko'rsatkichidir, va

```
*(p + 3)
```

massiv 3 - elementidir.

Oldingi bo'limda ko'rganingizdek, massiv nomini ko'rsatkichday ishlatishimiz mumkin. Bu degani, a + 3 3 indeksli massivning ko'rsatkichidir , va *(a + 3) a[3] bilan bir xil ma'noga ega.

Umuman olganda, har bir butun son n ga

a[n] *(a + n) bilan bir xil.

Bu bog'liqlik massiv/ko'rsatkichning ikki taraflama qonunidir.

Bu qonun C++ da hamma massivlar nimaga 0 indeks bilan boshlanishini

tushuntiradi. Ko'rsakich a yoki (a + 0) massivning boshlanish elementini ko'rsatadi. Shuning uchun bu element a[0]

bo'lishi kerak. Arifmetik ko'rsatkichni yaxshiroq tushunish uchun, haqiqiy xotira manzilini bilamiz deb tasavvur qilaylik. Tasavvur qiling, a array 20300 manzilida boshlanadi. Massiv double turidagi 10 ta baxoni o'z ichiga oladi. Double baxosi 8

bayt xotirani egallaydi. Shuning uchun massiv 80 bayt xotirani egallaydi, 20300 dan 20379 gacha. Boshlang'ich baxo 20300 manzilida joylashgan, keyingisi 20308 manzilda, va h.k.(3-tasvirni ko'ring).Masalan, a + 3ning baxosi 20300 + 3 □ 8 = 20324.

317-betdagi 3-jadval arifmetik ko'rsatkich va massiv/ko'rsatkichning ikki

taraflama bog'lanishini shu misol orqali ko'rsatadi.

Agar massiv va ko'rsatkichlar o'rtasidagi bog'liqlikni tushungan bo'lsangiz, massiv parametrini boshqa turdagi parametrlardan farqini bilasiz.Misol tarzida, massivning hamma baxolarini yig'indisini hisoblaydigan funksiyani oling:

```
double sum(double a[], int size)
{
 double total = 0;
 for (int i = 0; i < size; i++)
 {
 total = total + a[i];
 }
 return total;
}
```

Bu yerda funksiya berilgan(4-tasvirni ko'ring):

```
double data[10];
... // Initialize data
double s = sum(data, 10);
```

Baxoning ma'lumoti funksiyaning yig'indisida berilgan. u odatda double* turidagi ko'rsatkich, massivning boshlang'ich elementini ko'rsatadi. Shuning uchun funksiya shu tarzda bo'ladi deb kutiladi:

double sum(double* a, int size)

Biroq, funksiyaga yaxshilab qarasangiz,parametrning o'zgaruvchanligi massivning

bo'sh chegaralari bilan e'lon qilinishini ko'rasiz

double sum(double a[], int size)

C++ ning tuzuvchisiga ko'ra, bu parametrlar mutlaqo bir xil. [] bu belgi -

ko'rsatkichni e'lon qiladigan "sintaktik shakar". (Kompyuter olimlari bu terminni odamlar o'qishi uchun oson va mukammal tafsilotlarni yashirish uchun ishlatishadi.)

massiv funksiyaga berilgandek tuyilishi mumkin, lekin aslida funksiya massivning faqat boshlang'ich manzilini qabul qiladi.

Endi funksiyani asosiga shu ifodani oling:

total = total + a[i];

C++ ning tuzuvchisi a ni massiv emas, ko'rsatkich qilib ko'rsatgan. a[i] ifodasi *(a + i) uchun sinkratik shakl hisoblanadi. Bu ifoda hotira joylashuvining i elementlarini bildiradi, a o'zgaruvchining manzilini emas. 4-tasvirda i 3 ga teng bo'lganida a[i] qanday qabul qilinishini ko'ramiz.

Massivning kattaligini doim belgilash nega muxumligini endi tushundiz. Funksiya massiv qayerda boshlanishini aytib turuvchi a manzilni qabul qiladi. Xotira manzili yordamida funksiya massivning baxolarini joylashtiradi.Lekin funksiya qayerda to'xtashni ham bilishi kerak.

6–9 savollarda o'zingizni sinang,3-tasvidigidaka massiv va ko'rsatkich chizing. Boshlang'ich manzilni belgilang (20300ni olsangiz qulay), va har bir baxo 4 bayt olishini belgilang.

6. Quyidagi ifodalardan keyin massiv a qanday bo'ladi? int a[] = { 2, 3, 5 };

```
int* p = a;
p++;
*p = 0;
```

7. Quyidagi ifodalardan keyin massiv a qanday bo'ladi?

```
int a[] = { 2, 3, 5 };
int* p = a + 1;
*(p + 1) = 0;
```

8. Quyidagi ifodalardan keyin massiv qanday bo'ladi?

```
int a[] = { 2, 3, 5 };
int* p = a;
int* q = a + 2;
p++;
q--;
*p = *q;
```

9. Quyidagi ifodalar nimani muhrlaydi? int a[] = { 2, 3, 5 };

```
cout << *a + 2 << " ";
```

```
cout << *(a + 2) << endl;
```

10. 6-bobda, biz bo'sh kvadrat funksiyani aniqladik (int n, int result[])

2.3 Ko'rsatkichlar - massivning o'zgaruvchanligi.

Ko'rsatkichdan foydalanib o'zgaruvchan parametrni aniqlang.

Qo'llang: Endi bobning ohiridagi shu mashqlarni bajaring: R7.7, R7.8, P7.4.

massivdan o'tish uchun ko'rsatkichni ishlatish

7.2.3 bo'limdagi funksiya yig'indisini oling. Endi bilganingizday, funksiya yig'indising birinchi parametri-bu ko'rsatkich, funksiyani biroz boshqacha tarzda ifoda qilishingiz mumkin. Butun son indeksini ko'paytirish o'rniga, massiv elementlarini ko'rish uchun ko'rsatkich o'zgaruvchanligini ko'paytirishingiz mumkin:

```
double sum(double* a, int size)
{
 double total = 0;
 double* p = a; // p starts at the beginning of the array
 for (int i = 0; i < size; i++)
 {
 total = total + *p; // Add the value to which p points
 p++; // Advance p to the next array element
 }
 return total;
}
```

Aslida, p ko'rsatkichi a[0] elementini ko'rsatadi.

p++ ning ko'payishi;

uni keyingi elementga olib boradi (5-tasvirni ko'ring).

Massiv elementini a[i] qilib qabul qilgandan ko'ra, uni ko'paytirgan maqul. Shu sababdan, ayrim programmistlar massiv elementlariga yetish uchun indekslardan emas, ko'rsatkichlardan foydalanishadi.

Aniq programmalashtiring

Ayrim programmistlar instruksiyalar sonini kamaytirishga urunishadi, natija kodi tushunish uchun qiyin bo'lsa ham. Masalan, bu yerda funksiya yig'indisining to'g'ri joriy etilishi berilgan:

```
double sum(double* a, int size)
{
 double total = 0;
 while (size-- > 0) // Jarayon davomiyligi
 {
 total = total + *a++; // ; increment a
 }
 return total;
}
```

Bu 2 yo'l bilan amalga oshirildi. Birinchisi, funksiyaning parametri va o'lchami o'zgaruvchan, qiymatlarni o'zgartirish-qonuniy. Ifodalarning o'lchami -- va ++ o'zgaruvchanning ko'payishi yoki kamayishi va eski baxosiga qaytishini anglatadi. Boshqa so'z bilan aytganda

size-- > 0 ifodasi

2 ta vazifani bajaradi: o'lchamni kamaytirish va undan oldin uni pozitivligini tekshirish. Huddi shunday

*a++ ifodasi

ko'rsatkichni keyingi elementga ko'paytiradi, va elementni ko'payishdan oldingi xoliga

qaytaradi.

Iltimos bunday programmalash uslubini ishlatmang. Sizni ishingiz o'z aqlingiz bilan

boshqalarni ko'zini qamashtirish emas,balki tushunish oson bo'lgan kodni yozish.

Ko'rsatkichni o'z joyiga qaytarish

2 ta elementli massivga ko'rsatkichni qaytarmoqchi bo'lgan funksiyani oling, massivning birinchi va so'nggi baxolari:

```
double* firstlast(double a[], int size)
{
 double result[2];
 result[0] = a[0];
 result[1] = a[size - 1];
 return result; // Error!
}
```

Funksiya ko'rsatkichni massivning boshlang'ich elementiga qaytardi. Biroq, bu massiv birinchi funksiyaning o'zgaruvchanidir. Funksiya chiqib ketsa o'zgaruvchan ham ketadi.Yaqinda ular boshqa funksiyalar bilan to'ldiriladi.

Massivni kesib o'tib mu masalani yechishingiz mumkin:

```
void firstlast(const double a[], int size, double[]
result)
{
 result[0] = a[0];
 result[1] = a[size - 1];
}
```

Massivni joylashtirish chaqiruvchining zimmasida.

Doimiy Ko'rsatkichlar

Quyidagi ifoda doimiy ko'rsatkichni aniqlaydi:

```
const double* p = &balance;
```

P ko'rsatkichdagi baxoni o'lachayolmaysiz. Quyidagi ifoda noto'g'ri:

```
*p = 0; // Error
```

Albatta, baxoni qisangiz bo'ladi:

```
cout << *p; // OK
```

Doimiy massivning parametr o'zgaruvchanligi doimiy ko'rsatkichga teng. Masalan:

```
double sum(const double[] values, int size)
```

7.2.3 bo'limdan baxo-ko'rsatkichligini eslang. Funksiya shunday aniqlanishi mumkin:

```
double sum(const double* values, int size)
```

Massiv elementlarini o'qish uchun funksiya ko'rsatkich baxolarini ishlatadi, ammo ularni aniqlayolmaydi.

Nazorat savollari

1. C++ dasturida nechta funksiya bo’ladi?
2. Kutubxonalr nima?
3. Nomlangan nomalar sohasi nima?
4. Ko’rsatgichlar nima?
5. Nimaga ko’rsatgichlar qulay emas?
6. Kompyuter o’yinlari yaratish uchun dasturchi qay darajada malakali bo’lishi kerak deb o’ylaysiz?
7. Joriy bajarilayotgan ko’rsatgichlar qayerda saqlanadi?

3-Ma’ruza. Ko`rsatkich va struktura

Ma’ruza rejasi:

3.1 Strukturalar;

3.2 Structuralarga ko'rsatkichlar;

3.3 Structuralar bilan ko'rsatgich a'zolar.

Kalit so’zlar:*, ro’yxat, manzil, nolinchi ko’rchsatkich, tugun, adres olish &, bo’shatish, ko’rsatkich, virtual destruktor, xotira, xotira chiqishi, destruktor, toifani o’zlashtirish, resurslar chiqishi, a’zo destruktori.*

3.1 Strukturalar

Structura nomi ko’cha manzili bu yolg’iz mohiyatga ikki ahamiyatni birlashtirish

uchun tariflanishi mumkin. C++da, biz struct odamovi so’zli structurani tariflaymiz:

```
struct ko’cha manzili
{
 int uy_raqami
 string ko’cha_ismi;
};
```

bu tariflash maydonlarisiz o’zgaruvchilarni elon qilish uchun foydalanadigan ko’cha manzilining yangi turi;

ko’cha manzili oq uy oq uy ozgaruvchisida azolar deb ataladigasan ikki qismi bor, uy_raqami va

ko’cha_nomi. Siz bu nuqtalardan har bir azolarning kirishida foydalanasiz, quyidagiga o’xshagan:

oq_uy,uy_raqami = 1600;

oq_uy,ko’cha_nomi = "Pennsylvania Avenue";

3.2 Structuralarga ko'rsatkichlar

Bu jo’shqin structura ahamiyatini ajratmasi uchun odatlangan, yangi operatordan foydalanish ko'rsatgich manzili * manzil_ko’rsatgichi yangi ko'rsatgich manzili. Faraz qiling manzil ko’rsatgich ko’rsatgichiga structuraning uy raqamini o’rnatishni xoxlaysiz manzil_ko’rsatgich,uy_raqami=1600;//XATOLIK

Baxtga qarshi,bu sintaksis xato. Bu nuqta operatori *operatoridan ko’ra ustunroq. Bu siz anglagan kompliyator o’ylari

(manzil_ko’rsatkich,uy_raqami)=1600;//XATOLIK

Manzil ko’rsatgichi ko’rsatgich bo’lishiga qaramay u structura emas. Siz operator ko’rsatgichiga va kompliyator xatolar haqida bayon qiladigan nuqtalarga etibor qila olmaysiz O’rniga siz birinchi*operatoriga keyin nuqtaga aniq etibor berishingiz kerak (manzil_ko’rsatgich,uy_raqami)=1600;//XATOLIK.Chunki bu shunchalik odatiy holatki C+ +ning loyihachilari.

3.3 Structuralar bilan ko'rsatgich a'zolar

Structuraning azosi ko'rsatgich bo'lishi mumkin. Bu holat odatda malumot structura qiymatlari orasida bolinganda vujudga keladi . Quyidagi misolni hisoblangTashkilotlarning ko'p sonli offislarida har bir ishchida ism va ish joylari bor.

```
struct Employee
{
 string name;
 StreetAddress* office;
}
```

Bu yerda biz ikki ishchining hisoblash idoralarini tariflaymiz:

StreetAddress accounting;

accounting.house_number = "1729";

accounting.street_name = "Park Avenue";

Employee harry;

harry.name = "Smith, Harry";

harry.office = &accounting;

Ko'cha manzili turi va viloyat: 2201 C ko'cha manzilli dastlabki o'zgaruvchilarni elon qilish.

32. Ko'cha manzili turi va viloyat: 2201 C Ko'cha NW. manzilidan tashkil topgan dastlabki o'zgaruvchilarni ko'rsatish.

33. Struktura turi sanasini 4-iyul yoki 31-dekabr kabi ko'rsatib tariflash.

34. Siz tariflagan 33 o'zgaruvchi turidan mustaqillik kuni va dastlabki 4-iyul kabi foydalanish.

35 Harryning offis manzilini qanday chop qilasiz?

amaliyot Hozir siz bu mashqlarni R7.23, R7.24, P7.16.boblar oxirida ko'rishingiz mumkin

ko'rsatgich ozgaruvchilaridan foydalanish va tariflash

Ko'rsatgich xotirada o'zgaruvchining turgan joyini bildiradi.

• T*turiko'rsatgich o'zgaruvchisining T turini bildiradi.

• & operator maydoni o'zgaruvchining turgan joyidir.

• * operator ko'rsatgich ko'rsatadigan o'zgaruvchining kirishiga ruxsat beradi

• Tajribasiz ko'rsatgich foydalanadigan xatolik.

• NULL ko'rsatgich biror obyektni ko'rsatmaydi

C++da nayzalar va ko'rsatgichlar orasidagi aloqani tushunish.

• Nayza o'zgaruvchisining nomi nayza elementing boshlanishiga ko'rsatgichdir.

• Ko'rsatgich arfimetigi berilgan elementlar raqamidan sakrab o'tadigan qo'shilgan butun son o'rnini qoplovchi nayza ko'rsatgichi yani maydonlash ko'rsatgichini anglatadi.

• Nayza/butun son o'rnini to'ldiradigan a[n] aniqlovchisi bo'lgan *(a+n) past holat ikkilamchi ko'rsatgichini ko'rsatadi.

• Nayzani funcsiyaga bosganda faqat manzil ko'rsatiladi.

C++string obyektlarini nayza xarakteridagi funksiyalar bilan ishlatish.

• Char turining ahamiyati shaxsiy xarakterga beriladi. Tom manolardagi xarakter-

lar birlikka asoslanib yopiladi.

• Tom manodagi string (juftlikka asoslanadi) u nol termini bilan char nayzasining ahamiyati

• Char tipining funksiyalari ko'plab

ko'rsatgichlardan foydalanadi.

• Char maydoni funksiya azosi string obyektini ko'rsatadi.

• Siz C++ozgaruvchilari bilan C stringlarni birinchi belgilab qoyishingiz mumkin • C++string obyekti bilan [] operator xarakterlariga birinchi kirishingiz mumkin.

Run vaqtigacha aniqlanmagan talabnoma xotirasi programma xotirasidan joy ajratadi va o'rnini aniqlaydi.

• Ajratilgan dinamic xotiradan foydalanish agar siz o'zingizga qancha qiymat ker-akligini bilmasangiz.

• To'dadan yangi operator ajratish.

• Dinamik ajratilgan obyekt bilan o'chirish yoki o'chirilgan obyekt bilan qaytarib olishingiz kerak.

• Erkin ko'rsatgichdan foydalanish

MISOLLARNI KO'RIB CHIQISH

r7.1 Quyidagi koddan nusxa olish .Saqlangan a va b 20300va20308 ni taxmin qilish. Sizning bnusxangiz a b va p uchun bo'lishi kerak.

```
double a=1000; double b =2000; double*p=&a;
*p = 3000;
p = &b;
a = *p * 2;
```

r7.2 Quyidagi koddan nusxa olish .Saqlangan a va b 20300va20308 ni taxmin qilish. Sizning bnusxangiz a b p va q uchun bo'lishi kerak.

```
double a = 1000; double b =  2000; double* p = &a;
double* q = &b;
*p = *q;
p = q;
*p = 3000;
```

r7.3 Quyidagi kodni chop etish nima?

```
double a = 1000; double b =  2000; double* p = &a;
double* q = p;
b = *q;
p = &b;
a = *p + *q;
cout << a << " " << b << endl;
```

r7.4 Quyidagi kodda xatoliklarni tushuntirish.Hamma xatolikdan tashkil topgan qator-larni emas.Har bir qator uni boshqaruvini amalga oshirishga bog'liq.

```
1  double a = 1000;
2  double* p = a;
3  int* p = &a;
4  double* q;
5  *q = 2000;
6  int* r = NULL;
7  *r = 3000;
```

r7.5 Faraz qiling biror kalit sifatidagi stringdan hatto bo'sh sttringdan ruxsat berilgan sis-temada foydalanishuppose shunga qaramay foydalanuvchi sistemaga birinchi marta

kodsiz bog'lanadi

String o'zgaruvchisi va NULLko'rsatgichini bo'sh kodlar orasidan ajratishni qan-day tasvirlaysiz.

r7.6 Tarif berish

double primes[] = { 2, 3, 5, 7, 11, 13 };

double* p = primes + 3;

quyidagi ifodalanishning manosini tushuntiradigan diagramma chizish:

a. primes[1]

b. primes + 1

c. *(primes + 1)

d. p[1]

e. p + 1

r7.7 Asosiy nayzani tariflash

double primes[] = { 2, 3, 5, 7, 11, 13 };

Xotirada joyni 20300 da boshlash

a. primes

b. *primes

c. primes + 4

d. *(primes + 4)

e. primes[4]

f. &primes[4]

r7.8 Asosiy nayzani tariflash

double primes[] = { 2, 3, 5, 7, 11, 13 };

7,2,3 sohadagi muhokama qilingan funksiyalarni yig'indisini hisoblash

b. sum(primes, 4);

c. sum(primes + 2, 4);

d. sum(primes, 0);

e. sum(NULL, 4);

r7.9 Asosiy nayzani tariflash

double primes[] = { 2, 3, 5, 7, 11, 13 };

Xotira joyini 20300dan boshlash funksiya asosini sum(primes, 4), deb nomlash 318

betdagi tariflangan 7.2 maxsus turdan foydalanish sizning asosiy stolinggiz a, size, total, p, and i ko'rsatadi.

r7.10 Ko'rsatgichlar raqamli o'zgaruvchisi bo'lganlarni adreslaydi siz o'zgaruvchilarni cout << (unsigned)(p) sifatida chop etishingiz mumkin natijalarnitushuntirish int* and q is a double* bilan p, p + 1, q, va q + 1 solishtiradigan dastur yozish

r7.11 Bir xil o'zgaruvchilarni ko'rsatgichidan tashkil topgan o'zgaruvchilarni ko'rsatish NULL yoki random qiymatiga o'zgaruvchilarni ko'rsatish a,b,c o'zgaruvchilarini yaratish va o'rnatilgan 4 korsatgichli kod yozish va bularning imkoni borini ko'rsa-tish.

r7.12 Funksiya butun son nayzasi qo'lga kiritilgan birinchi va oxirgi qiymatlarni bajar-ish va nayzadagi natijani zaxiraga saqlash.

r7.13 Quyidagi misollarni manosini tushuntirish:

a. "Harry" + 1

b. *("Harry" + 2)

c. "Harry"[3]

d. [4]"Harry"

r7.14 Quyidagi o'zgaruvchilar tariflari orasidagi farq nimada?

a. char a[6] = "Hello";

b. char* b = "Hello";

r7.15 Quyidagi 3ta o'zgaruvchilar tariflari orasidagi farq nimada?

a. char* p = NULL;

b. char* q = "";

c. char r[1] = { '\0' };

r7.16 Dastur segmentini hisoblash:

```
char a[] = "Mary had a little lamb";
char* p = a;
int count = 0;
```

```
while (*p != '\0')
{
 count++;
 while (*p != ' ' && *p != '\0') { p++; }
 while (*p == ' ') { p++; }
}
```

r7.17 Quyidagi C++ da 3 marta qaytaradigan kodni hisoblash.

```
string a = "Hello";
string b = a + a + a;
```

Faraz qiling C string char t[100];dek elon qilinadi

C string uchun qaytaradigan ekvivalent kod yozish.

r7.18 Quyidagi C++ huquqlarini ajratish

```
void f(int p[])
{
 int* q;
 const int* r;
 int s[10];
 p = q;
 p = r;
 p = s;
 q = p;
 q = r;
 q = s;
 r =  p;
r = q;
 r = s;
 s = p;
 s = q;
 s = r;
```

}

r7.19 Agar siz heapdan qo'lga kiritgan obyektlarni o'chirib tashlashni unutgan bo'lsangiz nima sodir boladi?Agar siz ikki marta o'chirsangiz nima sodir bo'ladi?

r7.20 Tajribasiz va NULL ko'rsatgichini o'chirib tashlaydigan dastur yozisg. Dastur ishlaganda nima sodir bo'ladi?

r7.21 Quyidagi koddan xatoliklarni topish Hamma qator xatodan tashkil topmagan har bir qator bitta xatoga bog'liq tajribasiz NULL ko'rsatgich uchun xatolarni ko'rib chiqish.

```
1   int* p = new int;
2   p = 5;
3   *p = *p + 5;
4   string s = "Harry";
5   *s = "Sally";
6   delete &s;
7   int* a = new int[10];
8   *a = 5;
9   a[10] = 5;
10   delete a;
11   int* q;
12   *q = 5;
13   q = p;
14   delete q;  15   delete p;
```

r7.22 Ko'rsatgich va to'plam bo'lmaganda uchburchaksimon ikki o'lchamli to'plamdan foydalanishni qanday aniqlaymiz?

r7.23 To'plamdagi birlashtirilgan ko'cha manzili uchun sohalarni qayta yozish

7.24 Shaxsni ota onasi uchun ko'rsatmalar va shaxsni o'z ichiga olgan shaxs tipi struc-

turasini loyihasini tur ko'rsatgich bo'glanishlarni to'g'ri asos solinishi sizning ota onanggiz va siz uchun structura qiymatini aniqlaydiganholatlarni yozing (ota onanggizni ota onasi uchun NULL dan foydalaning)

r7.25 7.4 mashqdagi maximum deb ataladigan funksiyani natijasini ko'rsatadigan shakl chizish

r7.26 Bir qancha chiziqlarni o'qishdan keyin P7.13mashqdagi chiziqlar to'plamida ko'rsatmalarni ko'rsatish funksiyasini chizish.

r7.27 Praragrifda tekshirgan foydalanuvchi uchun har bir pargirifda 7.6 soha das-tur yoritilgan foydalanuvchiga tekshirib biladigan bu paragrifni bilishda qiyinchilik tug'diradi bu muommoni yechish uchun ko'rsatgich char ning to'plamini ko'rsa-tadigan loaned_items. loaned_items[i] parallel tekshirib bilgan to'plami bor agar bir qancha paragrifni foydalanuvchi tekshirib bilmasa loaned_items. loaned_items[i] nolga teng bo'ladi.

P7.4 Massivni maksimum qiymatini topadigan rekursiv funksiya tuzing.

double* maximum(double* a, int size)

Agar hajim 0 bo'lsa, return NULL.

P7.5 :qarama qarshi funksiya ichida double tipida va int tipida uning o'lchami

kiritiladi.

void reverse(double* a, int size)

P7.6 strncpy funksiyasi standart kutubxonadan olinadi.

P7.7 int strncpy(const char s[], const char t[]) U s ni elementlarini t ga nusxalaydi.

P7.8 Funksiyani yozing :

void reverse(char s[])

U s massivni teskari tartibda joylashtiradi.Masalan "Harry"so'zi "yrraH" ga aylanadi.

P7.9 Strncpy va strncat funksiyalari:

void safe_concat(const char a[], const char b[], char result[],

int result_maxlength)

P7.10 Foydalanuvchi Q kiritmaguncha cin orqali ma'lumot o'quvchi int* read_data(int& size) funksiyasini yozing. Funksiya no'merli kiruvchi ma;lumotlar hajmini o'rnatadi va bu bazaga ko'rsatkich qaytariladi.

Bu masssiv elementlarining aniq hajmi bo'lishi shart. Albatta, siz foydalanuvchi kiritishi mumlik bo'lgan ma'lumotlar hajmini oldindan bilmaysiz.

10 ta elementli massivdan boshlang va massiv to'lmaguncha hajim ikki marta oshirib boriladi. Oxirida, joriya hajimli massiv yaratiladi va barcha ma'lumotlar bu massivga kiritiladi. Hosil qilingan oraliq massivlar o'chiriladi.

P7.11 Eng pastki Hisoblagich bar grafigi chop tomonidan Galton kengashi simülasyonunu oshirish.

1 18 64 149 239 265 186 61 15 2

* * * * * * * *

* * * * * *

* * * * * *

* * * *

* * * *

* * * *

* * * *

* * *

* * *

* *

* *

* *

*

P7.12 334 betdagi 7.1 nusxa holatini tekshirish sizning dasturinggiz 1-bo'sh menyuni ko'rsatishi kerak Foydalanuvchi adminstrator dasturini berkitish.Foydalanuvchi

uchun nusxaning nomeri bo'sh menuga qaytish va ID uchun kechiktirmay bajarish. Adminsrrator uchun bu menyuni hisoblash.

Master Project balansidagi tanlovda bosh hisobga uni bog'lash va ID foydalanuvchisi uchun kechiktirmay reja tanlovida bosh menyuga qaytish.

P7.13 Car bufferi [1000] ga ularni birlashtirish a matn chiziqlarini o'qiydigan dastur

yozish 1000 xarakterini o'qigandan keyin to'xtash Matnni o'qiganinggizda o bilan n

atamalar xususiyatini hammasini yangi chiziqda qayta joylash. Matndagi chiziqlarni boshlanish to'plamini ko'rsatadigan ko'rsatgichlar bo'lganligi uchun [100] xarakter to'plamini chop etish Agar kiritilgan malumot ko'proq bo'lsa malumotlarni faqat 100 tasini ko'rib chiqing eng so'ngi kiritilgan malumot liniyasi bilan boshlash zid turdagi liniyani ko'rsatadi.

P7.14 100 ta chiziq yoki 1000 ta xarakterning faqat kiritilgan malumotlarini nazo-rat qiladigan fact orqali chegaralangan tushunganinggizda bu chegarani olib tashlash bir uzun tizimli narsada keltirilgan malumotni seriyalarga ajrating .

P7 15Tizimning xususiyat bufferidagi charni qo'lga kiritish uchun c string funksiyasidan foydalaningf char vectoridagidek chiziqning bajarilgan to'plarining vector classlari va tizimdan qanday foydalanish P7.14 mashqda ko'rsatilgan bu mashqda siz qo'l qobilyatida bajarishingiz kerak to'plamdan (new char[1000]) bufferining 1000 xususiyatini bajaring.

Nazorat savollari

1. 1

   2
2. 2 2
3. 30
4. 25

5. 15 o'zgaruvchan a xotira tomonidan taqibdir. Manzili keyingi bir dasturda farq qilishi mumkin.

6. 2 0 5

7. 2 3 0

8. 2 3 5

9. 4 5

10. void squares(int n, int* result)

11. 15: 'H"e"l"l"o"," "W"o"r"l"d"!"\n"\0'

12. 14. Null teminator sanalmaydi, va '\n' bitta belgi sifatida koriladi.

13. char* p = "Hello";

4-Ma’ruza. Havolalar

Ma’ruza rejasi:

4.1 Birlashmalar va ular ustida amallar.

4.2 Foydalanuvchi tomonidan aniqlangan berilganlar turi.

4.3 Makroslarni aniqlash va joylashtirish.

4.4 Makroslarda ishlatiladigan amallar.

Kalit so’zlar:*, ro’yxat, manzil, nolinchi ko’rchsatkich, tugun, adres olish &, bo’shatish, ko’rsatkich, virtual destruktor, xotira, xotira chiqishi, destruktor, toifani o’zlashtirish, resurslar chiqishi, a’zo destruktori.*

4.1 Birlashmalar va ular ustida amallar

Birlashmalar xotiraning bitta sohasida (bitta adres bo‘yicha) har xil turdagi bir nechta berilganlarni saqlash imkonini beradi.

Birlashma e’loni union kalit so‘zi, undan keyin identifikator va blok ichida har xil turdagi elementlar e’lonidan iborat bo‘ladi, masalan:

```
union Birlashma
 {
 int n;
 unsigned long N;
 char Satr[10];
 };
```

Birlashmaning bu e’lonida kompilyator tomonidan Birlashma uchun uning ichidagi eng ko‘p joy egallovchi elementning - Satr satrining o‘lchamida, ya’ni 10 bayt joy ajratiladi. Vaqtning har bir momentida birlashmada, e’lon qilingan maydonlarning faqat bittasining turidagi berilgan mavjud deb hisoblanadi. YUqoridagi misol-da Birlashma ustida amal bajarilishida uning uchun ajratilgan xotirada yoki int turidagi n yoki unsigned long turidagi N yoki Satr satr qiymati joylashgan deb hisoblanadi.

Birlashma maydonlariga xuddi struktura maydonlariga muro-jaat qilgandek ‘.’ orqali murojaat qilinadi.

Strukturalardan farqli ravishda birlashma e’lonida faqat uning birinchi elementiga boshlang‘ich qiymat berish mumkin:

```
union Birlashma
{
 int n;
 unsigned long N;
 char Satr[10];
}
 birlashma={25};
```

Bu misolda birlashma birlashmasining n maydoni boshlang‘ich qiymat olgan hisoblanadi.

Birlashma elementi sifatida strukturalar kelishi mumkin va ular odatda berilganni «bo‘laklarga» ajratish yoki «bo‘laklardan» yaxlit berilganni hosil qilish uchun xizmat qiladi. Misol uchun so‘zni baytlarga, baytlarni tetradalarga (4 bitga) ajratish va qaytadan birlashtirish mumkin.

Quyida baytni katta va kichik yarim baytlarga ajratishda birlashma va strukturadan foydalanilgan programmani matni kelti-rilgan.

```
#include <iostream.h>
union BCD
     {unsigned char bayt;
      struct
       {
        unsigned char lo:4;
        unsigned char hi:4;
   } bin;
     } bcd;
int main()
```

```
{
        bcd.bayt=127;
        cout<<"\n Katta yarim bayt :
"<<(int)bcd.bin.hi;
        cout<<"\n Kichik yarim bayt:
"<<(int)bcd.bin.lo;
        return 0;
       }
```

Programma bosh funksiyasida bcd birlashmasining bayt o‘lchamida bayt maydoniga 127 qiymati beriladi va uning katta va kichik yarim baytlari chop etiladi.

Programma ishlashi natijasida ekranga quyidagi natijalar chiqadi:

Katta yarim bayt : 7

Kichik yarim bayt: 15

Masala. Haqiqiy turdagi sonning kompьyuter xotirasidagi ichki ko‘rinishini chop qilish. Haqiqiy son float turida deb hisoblanadi va u xotirada 4 bayt joy egallaydi (1-ilovaga qarang). Qo‘yilgan masalani echish uchun birlashma xususiyatdan foydalaniladi, ya’ni xotiraning bitta adresiga haqiqiy son va belgilar massivi joylashtiriladi. Haqiqiy son xotiraga o‘qilib, belgilar massivining har bir elementining (baytining) ikkilik ko‘rinishi chop etiladi.

Programma matni:

```
#include <iostream.h>
const unsigned char bitlar_soni=7;
const unsigned char format=sizeof(float);
void Belgi_2kodi(unsigned char blg);
union Son_va_Belgi
{
 float son;
 unsigned char belgi[format];
};
```

```
int main()
{
 Son_va_Belgi son_va_belgi;
 cin>>son_va_belgi.son;
 for(int b=format-1; b>=0; b--)
  Belgi_2kodi(son_va_belgi.belgi[b]);
 return 0;
}
void Belgi_2kodi(unsigned char blg)
{
 unsigned char l0000000=128;
 for(int i=0;i<=bitlar_soni;i++)
 {
  if(blg&l0000000)cout<<'1';
  else cout<<'0';
  blg=blg<<1;
 }
 cout<<' ';
}
```

Programmada Son_va_Belgi birlashmasini e’lon qilish orqali float turidagi x o‘zgaruvchisini va float turi formatining baytlardagi uzunligidagi belgilardan iborat belgi massivini xotiraning bitta joyiga joylashuviga erishiladi. Bosh funksiyada birlashma turidagi son_va_belgi o‘zgaruvchisi e’lon qilinadi va uning x maydoniga klaviaturadan haqiqiy son o‘qiladi. Keyin belgilar massividagi har bir elementning ikkilik kodi chop etiladi. Ikkilik kodni chop etish 8 marta baytni 7-razryadidagi sonni chop etish va bayt razryadlarini bittaga chapga surish orqali amalga oshiriladi. SHunga e’tibor berish kerakki, belgilar massividagi elementlarning ikkilik kodlarini chop qilish o‘ngdan chap tomonga bajarilgan. Bunga sabab, son ichki formatidagi baytlarning xotirada «kichik bayt - kichik adresda» qoidasiga ko‘ra joylashuvidir.

Programmaga -8.5 soni kiritilsa, ekranda

11000001 00001000 00000000 00000000

ko‘rinishidagi ikkilik sonlari ketma-ketligi paydo bo‘ladi.

4.2 Foydalanuvchi tomonidan aniqlangan berilganlar turi

C++ tilida foydalanuvchi tomonidan nafaqat struktura yoki birlashma turlari, balki ayni paytda mavjud (aniqlangan) turlar asosida yangi turlarni yaratishi mumkin.

Foydalanuvchi tomonidan aniqlanadigan tur typedef kalit so‘zi bilan boshlanadi, undan keyin mavjud tur ko‘rsatiladi va identi-fikator yoziladi. Oxirida yozilgan identifikator - yangi yaratilgan turning nomi hisoblanadi. Masalan,

typedef unsigned char byte;

ifodasi byte deb nomlanuvchi yangi turni yaratadi va o‘z mazmuniga ko‘ra unsigned char turi bilan ekvivalent bo‘ladi. Keyinchalik, programmada xotiradan bir bayt joy egallaydigan va [0..255] oraliqdagi qiymatlarni qabul qiladigan byte turidagi o‘zgaruvchi (o‘zgarmaslarni) e’lon qilish mumkin:

```
byte c=65;
byte Byte=0xFF;
```

Massiv ko‘rinishidagi foydalanuvchi tomonidan aniqlanuvchi tur e’loni quyidagicha bo‘ladi:

```
typedef char Ism[30];
Ism ism;
```

Ism turidagi ism o‘zgaruvchisi e’loni - bu 30 belgidan iborat massiv (satr) e’lonidir.

Odatda echilayotgan masalaning predmet sohasi terminlarida ishlash uchun strukturalar qayta nomlanadi. Natijada murakkab tuzi-lishga ega bo‘lgan va zarur xususiyatlarni o‘ziga jamlagan yangi turlarni yaratishga muvofiq bo‘linadi.

Masalan, kompleks son haqidagi ma’lumotlarni o‘z ichiga oluvchi Complex turi quyidagicha aniqlanadi:

```
typedef struct
{
 double re; double im;
    } Complex;
```

Endi kompleks son e’lonini

Complex KSon;

yozish mumkin va uning maydonlariga murojaat qilish mumkin:

KSon.re=5.64;

KSon.im=2.3;

4.3 Makroslarni aniqlash va joylashtirish

Makros - bu programma (kod) bo‘lagi bo‘lib, ko‘rinishi va ishlashi xuddi funksiyadek. Biroq u funksiya emas. Funksiyalar va makroslar o‘rtasida bir nechta farqlar mavjud:

programma matnida uchragan makros ifodasi o‘z aniqlanishi (tanasi bilan) bilan preprotsessor ishlash paytida, ya’ni programma kompilyasiyasidan oldin almashtiriladi. SHu sababli makros funk-siyani chaqirish bilan bog‘liq qo‘shimcha vaqt sarfini talab qilmaydi;

makroslardan foydalanish programmaning boshlang‘ich kodi (matnini) kattalashuviga olib keladi. Bunga qarama-qarshi holda funksiya kodi yagona nusxada bo‘ladi va u programma kodini qisqa-rishiga olib keladi. Lekin funksiyani chaqirish uchun qo‘shimcha resurslar sarflanadi;

kompilyator makrosdagi turlar mosligini tekshirmaydi. SHu sababli, makrosga argument jo‘natishda turlarning mosligi yoki argu-mentlar sonining to‘g‘ri kelishi yoki kelmasligi haqidagi xatolik xabarlari berilmaydi;

makros boshlang‘ich kodga programma bo‘lagini qo‘yish vositasi bo‘lganligi va bunday bo‘laklar matnning turli joy-lariga qo‘yish mumkinligi sababli makroslar

bilan bog‘liq fiksirlangan, yagona adreslar bo‘lmaydi. SHu sababli makroslarda ko‘rsatkichlar e’lon qilish yoki makros adreslarini ishlatish imkoniyati yo‘q.

Makroslarni aniqlash uchun #define direktivasidan foydalaniladi. Funksiyaga o‘xshab makroslar ham parametrlarga ega bo‘lishi mumkin. Misol uchun ikkita sonni ko‘paytmasini hisoblovchi makros quyidagicha aniqlanadi:

```
#include <iostream.h>
#define KUPAYTMA(x,y)((x)+(y))
int main()
{
 int a=2, b=3;
 c=KUPAYTMA(a,b);
 cout<<c;
 return 0;
}
```

Misoldan ko‘rinib turibdiki, tashqi ko‘rinishi bo‘yicha mak-roslardan foydalanish funksiyalardan foydalanishga o‘xshash. SHuning uchun ularni ayrim hollarda ularga *psevdofunksiyalar* deb atashadi. Makroslar aniqlanishining yana bir o‘ziga xos tomoni shundaki, S++ tilida ularning nomlarini katta harflar bilan yozishga kelishilgan.

YUqoridagi misolning o‘ziga xos ko‘rinishidan biri bu makros parametrlarini qavs ichida yozilishidir. Aks holda makros aniqlani-shini (tanasini) matnga qo‘yishda mazmunan xatolik yuzaga kelishi mumkin. Masalan,

#define KVADRAT(x) x*x

Programma matnida ushbu makros ishlatilgan satr mavjud bo‘lcin:

int y=KVADRAT(2);

u holda, makros aniqlanishini matnga qo‘yish natijasida programma matnida yuqoridagi satr quyidagi ko‘rinishga keladi:

int y=2*2;

Lekin, programmada makrosni ishlatish

int y=KVADRAT(x+1);

koʻrinishida boʻlsa, makros aniqlanishini matnga qoʻyish natijasida ushbu satr

int y=x+1*x+1;

koʻrsatmasi bilan almashtiriladiki, bu albatta kutilgan almash-tirish emas. SHu sababli, makros aniqlanishida umumiy qoida sifatida parametrlarni qavsga olish tavsiya etiladi:

#define KVADRAT(x)(x)*(x)

Agar makros chaqirilishida turga keltirish operatoridan foydalangan holat boʻlsa, makros tanasini toʻliqligicha qavsga olish talab qilinadi. Misol uchun programma matnida makrosga murojaat quyidagicha boʻlsin:

double x=(double)KVADRAT(x+1);

Bu holda makros aniqlanishi

#define KVADRAT(x)((x)*(x))

koʻrinishi toʻgʻri hisoblanadi.

Makros aniqlanishida oxirgi eslatma sifatida shuni qayd etish kerakki, ortiqcha probellar makrosdan foydalanishda xatolik-larga olib kelishi mumkin. Masalan

#define CHOP_QILISH (x)cout<<x

makros aniqlanishida makros nomi CHOP_QILISH va parametrlar roʻyxati (x) oʻrtasida ortiqcha probel qoʻyilgan. Preprotsessor bu mak-rosni parametrsiz makros deb qabul qiladi, hamda “(x)cout<<x” satr ostini makros tanasi deb hisoblaydi va makros almashtirishlarda shu satrni programma matniga qoʻyilada. Natijada kompilyasiya xatosi roʻy beradi. Xatoni tuzatish uchun makros nomi va parametrlar roʻyxati oʻrtasidagi probelni olib tashlash etarli:

#define CHOP_QILISH(x)cout<<x

Agar makros aniqlanishi bitta satrga sigʻmasa, shu satr oxiriga ‘\’ belgisini qoʻyish orqali keyingi satrda davom ettirish mumkin:

#define BURCHAK3(a,b,c)(unsigned int)a+(unsigned int)b\

>(unsigned int)c &&(unsigned int)a+(unsigned int)s>\ (unsigned int)b &&(unsigned int)b+(unsigned int)s>\ (unsigned int)a

Makros aniqlanishida boshqa makroslar ishtirok etishi mumkin. Quyidagi misolda ichma-ich joylashgan makros aniqlanishi ko‘rsatilgan.

#define PI 3.14159

#define KVADRAT(x) ((x)*(x))

#define AYLANA_YUZI(r)(PI* KVADRAT(r))

Foydalanishga zarurati qolmagan makrosni programma matnining ixtiyoriy joyida #undef direktivasi bilan bekor qilish mumkin, ya’ni shu satrdan keyin makros preprotsessor uchun noaniq hisoblanadi. Quyida aylana yuzasini hisoblaydigan programma matni keltirilgan.

```
#include <iostream.h>
#define PI 3.14159
#define KVADRAT(x) ((x)*(x))
#define AYLANA_YUZI(r)(PI* KVADRAT(r))
int main()
{
 double r1=5,r2=10;
 double c1,c2;
 c1=AYLANA_YUZI(r1);
 #undef AYLANA_YUZI
 c2=AYLANA_YUZI(r2);
 cout<<c1<<endl;
 cout<<c2<<endl;
 return 0;
}
```

Programma kompilyasiyasida “c1=AYLANA_YUZI(r1);” satr normal qayti ishlangan holda “c2=AYLANA_YUZI(r2);” satri uchun AYLANA_YUZI funksiyasi aniqlanmaganligi haqida xatolik xabari chop etiladi.

4.4 Makroslarda ishlatiladigan amallar

Makroslar ishlatilishi mumkin bo‘lgan ikkita amal mavjud: ‘#’- satrni joylashtirish va ” ##” - satrni ulash amallari.

Agar makros parametri oldida ‘#’- satrni joylashtirish amali qo‘yilgan bo‘lsa, makros aniqlanishini matnga qo‘yish paytida shu o‘ringa mos argumentning (o‘zgaruvchining) nomi qo‘yiladi. Buni quyi-dagi misolda ko‘rish mumkin:

```
#include <iostream.h>
#define UZG_NOMI(uzg) cout<<#uzg<<'='<<uzg;
int main()
{
 int x=10;
 UZG_NOMI(x);
 return 0;
}
```

Programma ishlashi natijasida ekranda

```
x=10
```

satri paydo bo‘ladi.

Satr ulash amali ikkita satrni bittaga birlashtirish uchun xizmat qiladi. Satrlarni birlashtirishdan oldin ularni ajratib turgan probellar o‘chiriladi. Agar hosil bo‘lgan satr nomidagi makros mavjud bo‘lsa, preprotsessor shu makros tanasini chaqiruv bo‘lgan joyga joylashtiradi.

Misol uchun,

```
#include <iostream.h>
#define MACRO_BIR cout<<"MACRO_1";
#define MACRO_IKKI cout<<"MACRO_2";
#define MACRO_BIRLASHMA(n) MACRO_##n
int main(int argc, char* argv[])
{
```

```
 int x=10;
 MACRO_BIRLASHMA(BIR);
 cin>>x;
 return 0;
}
```

programmmasi preprotsessor tomonidan qayta ishlangandan keyin uning oraliq matni kuyidagi ko‘rinishda bo‘ladi:

```
int main(int argc, char* argv[])
{
 int x=10;
 cout<<"MACRO_1";
 cin>>x;
 return 0;
}
```

Satrlarni ulash amalidan yangi o‘zgaruvchilarni hosil qilish uchun foydalanish mumkin.

```
#define UZG_ELONI(i) int var ## i
...
UZG_ELONI(1);
...
```

YUqoridagi misolda makros o‘z aniqlanishi bilan almashtirish natijasida “UZG_ELONI(1);” satri o‘rnida

int var1;

ko‘rsatmasi paydo bo‘ladi.

Nazorat savollari

1. C++ dasturida nechta funksiya bo’ladi?
2. Kutubxonalr nima?
3. Nomlangan nomalar sohasi nima?

4. O’zgaruvchilar nima uchun qo’llaniladi?
5. Qanday o’zgaruvchi turlari bor?
6. O’zgaruvchilar qanday qiymatlar qabul qiladi?
7. Kiritish operatori qanday ishlaydi?
8. Chiqarish operatori sintaksisi qnaqa?

5-Ma’ruza. Mantli fayllar bilan ishlash operatorlari

Ma’ruza rejasi:

5.1 Matn fayllarini o'qish va yozish.

5.2 Oqimni ochish.

5.3 Fayl dan o'qish.

5.4 Faylga yozish.

5.5 Fayl ma'lumot almashishga misollar.

Kalit so’zlar:*, ro’yxat, manzil, nolinchi ko’rchsatkich, tugun, adres olish &, bo’shatish, ko’rsatkich, virtual destruktor, xotira, xotira chiqishi, destruktor, toifani o’zlashtirish, resurslar chiqishi, a’zo destruktori.*

5.1 Matn fayllarini o'qish va yozish

The C++ input/output library oqimlar tushunchasiga bog'langan. *input stream* bu ma'lumotlar manbayi, va *output stream* ma'lumotlar uchun joy. Ma'lumotlar uchun eng keng tarqalgan manbalar va belgilangan joylar sizning qattiq diskingizdagi fayllardir.

Data faqat bir ko'chirish tasmasini, bir vaqtning
o'zida bir-bandlari kabi kiritish oqimini yetib
kelishi ko'rsatilgan

F*aylga kirish uchun siz* file stream dan foydalanasz. Fayl stream ning asosan 3 turi mavjud: ifstream

(input uchun), ofstream (output uchun), va fstream (ikkala input va output uchun). Qachonki siz bu fayllardan birontasidan foydalanganingizda u <fstream> header ni o'z ichiga oladi.

Quyidagi bo'limlarda, siz fayllarni ma'lumotlarni qanday qayta ishlashni o'rganasiz. Bu fayllar katta ma'lumotlar silsilasini tahlil qilishda juda keng tarqalgan, chunki Fayl ishlash bir qancha fanlar juda foydali qobiliyat hisoblanadi.

5.2 Oqimni ochish

File stream dan biror narsani o'qish uchun uni ochish kerak.Stream ni ochganingizda diskda saqlangan fayl nomini berasiz. Aytaylik siz programmadek bir xil katalogda joylashgan input deb nomlangan fayldan ma'lumot o'qimoqchisiz. So'ngra fayni ochish uchun quyidagi funksiyalarni bajarasiz:

```
in_file.open("input.dat");
```

Bu bayonot input.dat nomli fayl bilan turli fayllar orasida bog'lanad.

Barcha stream lar obyektlardir, va siz ularni manipulyatsiya qilish uchun dot notation funksiyasidan foydalanishingiz mumkin.

Yozish uchun faylni ochishda ofstream dan foydalaniladi. O'qish va yozish uchun faylni ochishda fstream dan foydalanasiz.

Fayl nomlari katalog yo'nalishi ma'lumotlarni o;z ichiga oladi, masalan;

~/homework/input.dat (UNIX)

c:\homework\input.dat (Windows)

Qachonki fayl nomini so'zma so'z belgilaganingizda , va nom backslash xossasini o'z ichiga oladi.(xuddi Win dows filename dek),siz har bir backslashni ikki marta ishlatishni ta'minlashingiz kerak:

```
in_file.open("c:\\homework\\input.dat");
```

Recall that a single backslash tom ma'nodagi so'zma so'z yolg'iz backslash

bayonoti bu **qochish xarakterida,** qaysiki boshqa xususiyat bo'lgan maxsus ma'no shakli bilan bog'langan, masalan: \n yangi satr uchun. \\ kombinatsiysi yolg'iz backslash ma'nosini anglatadi.

Agarda string variable da saqlangan nomni o'tqazishni xohlasangiz, C++ string ni C string ga o'zgartirish uchun c_str funksiyasidan foydalaning:

```
cout << "Please enter the file name:";
string filename;
cin >> filename;
ifstream in_file;
in_file.open(filename.c_str());
```

Qachonki programma tugallanganda, siz ochgan barcha stream lar avtomatik

tarzda yopiladi. Bundan tashqari, qo'lda yaqin komponenti funksiyasi bilan stream ni yopish mumkin:

in_file.close();

Agar stream variable dan boshqa fayl bilan ma'lumot almashish uchun yana

foydalanishni xohlasangiz Manual yopish aynan zarur.

5.3 Fayl dan o'qish

File stream dan ma'lumot o'qish bu to'liq to'g'rilik: Siz shunchaki har doim cin

dan o'qish uchun foydalangan funksiyalarni ishlatasiz:

```
string name;
double value;
in_file >> name >> value;
```

Muvaffaqiyatsizlikka uchragan funkisayalardan ma'lum bo'ladiki input ham qulagan. siz allaqachon cin bilan bu funksiyadan foydalangansiz, konsol kiritish xatolar uchun tekshirish. File stream larga ham shu tarzda yondashiladi. Qachonki fayldan son o'qishga harakat qilsangiz, va keyingi ma'lumotb birligi bu prop-erly formatidagi son emas, so'ngra stream omadsizlikka uchraydi. Ma'lumotni o'qigandan so'ng, qayta ishlash muvaffaqiyati uchun oldin sinash kerak:

```
if (!in_file.fail())
{
 Process input.
}
```

shu bilan bir qatorda, siz bir faktdan foydalanishingiz mumkin >> operator “not failed” sharoitiga aylanadi, ya'ni, input bayonoti va test sinovi imkoniyatini bergan holda:

```
if (in_file >> name >> value)
{
```

Process input.

}

qachonki fayldan input ni oqiganingizda, soni formatidagi xatolar

Bundan tashqari, agarda faylni ochsangiz va nom mantiqan izchil emas, yoki bu nomdagi fayl mavjud emas, keyin bu file stream ham omadsizzlikka uchragan maqomiga ega bo'ladi. Mag'lubiyatlarni kechiktirmasdan sinab ko'rish bu yaxshi fikr.

5.4 Faylga yozish

Faylga yozish uchun, siz ofstream yoki fstream variable ni aniqlang va uni oching.. keyin output file ga ma'lumot jo'nating, bir xil operatsiyalardan foydalangan holda, qaysiki cout lar bilan:

```
ofstream out_file;
out_file.open("output.txt");
out_file << name << " " << value << endl;
```

5.5 Fayl ma'lumot almashishga misollar

Quyida fayldagi ma'lumotlarni almashtirishning oddiy namunasi keltirilgan. ijtimoiy xavfsizlik boshqarmasi eng mashxur chaqaloqlar ro'yhatini o'zlarining web saytlarida e'lon qilishdi:their ularning web saytlaridan, http://www.ssa. gov/ OACT/babynames/. Agar berilgan o'n yil davomida 1000 eng mashhur nomlar so'rov bo'lsa, brauzer ekranida natija ko'rsatadi (1-rasmga qarang).

Ma'lumotlarni matndek saqlash, shunchaki uni tanlang va faylga natija joylashtirish. Bu kitoblarning hamroh kodi babynames.txt deb nomlangan faylni,

ya'ni 1990 yillardagi ma'lumotlarni qamragan, o'z ichiga olgan. Fayldagi har bir satr 7 ta kirisgdan tashkil topgan;

- Unvon (1 dan 1,000 gacha)
- Ism, tezlik, va erkaklar va ayollar ismlarining foizlari
- Ism, tezlik, va erkaklar va ayollar ismlaring foiz hisobi

Masalan, bu qatordagi

10 Joseph 260365 1.2681 Megan 160312 0.8168

Ko'rinadiki 260,365 ta tug'ilishdan har 10- bolaning ismi Josef bo'lgan yoki osha davrdagi barcha tug'ilishlarning 1.2681 foizi. Har 10-chi qizning eng keng tarqalgan ismi Megan bo'lgan. Nimaga bu yerda Joseflar Meganlardan ko'proq ? Ota-onalar, ularning qiz nomlari kengroq

tarqalgan majmuyidan foydalanishgan ko'rinadi.

Keling, ro'yxatda o'g'il-qizlar yuqori 50 foizgacha berilgan

nomlarini aniqlash tomonidan, deb gumonda sinov qilaylik.

Har bir chiziqni qayta ishlash uchun, biz birinchi navbatda o'qib:

```
int rank;
in_file >> rank;
```

Biz keyin bola ning ismi uch qadriyatlar majmuyi

o'qidik

ism:

```
string name;
int count;
double percent;
in_file >> name >> count >> percent;
```

keyin biz qizlar uchun bosqichni takrorlaymiz.

Chunki harakatlar o'ziga xos, Biz buning uchun yordamchi funktsiya process_name ni ta'minlaymiz. 50 foizga yetgandan so'ng jarayonni to'xtatish uchun biz tezlikni qo'shishimiz mumkin va 50 foizga yetgandan so'ng to'xtatamiz. Shunga qaramasdan, u ancha oddiyroq bo'lib chiqdi, boshlang'ich jami 50 va chastotalarni olib tashlash. Biz o'g'il bolalar va qizlar uchun alohida jamuljamlikni ajratishimiz kerak. qachonki jami 0 dan pastga tushish pasayganda, chop etishni to'xtatamiz.

Quyidagi kod process_name funktsiyasi in-file parametr o'zgaruvchan mos yozuvlar parametr ekanligini unutmang. O'qish va yozish o'zgartiruvchilari stream

variable dir. Stream variable monitorlari qancha xususiyatlar o'qilgan yoki yozilgan. har bir o'qish va yozish operatsiyalari ma'lumotlarni o'zgartiradi. Shu sababdan, siz har doim stream parametr o'zgaruvchilar mos yozuvlar parametrlarini aniqlash kerak.

To'liq programma pastda keltirilgan. Ko'rib turganingizdek, bir fayl o'qish o'qish klaviatura kiritish kabi osondir.

Dastur ishlab chiqarishga qarang. Hayratlanarlisi, atigi 69 o'g'il bolalar ismi va qizlar ismining 153 tasi jami tug'ilishlarining yarmini tashkil etgan. Bu shaxsni ko'rsatuvchi biznes bilan shug'ullanuvchilar uchun yaxshi yangilik. 8-betdagi 10-mashqsizdan bu taqsimlash oxirgi yillarda qanday o'zgarganini o'qishni talab qiladi.

ch08/babynames.cpp

```
1 #include <iostream>
2 #include <fstream>
3 #include <string>
4
5 using namespace std;
6
7 /**
8 Nom ma'lumotlarini o'qing, agar nom jami  >= 0 bolsa chop qiling, va umumiyni o'zgartiring.
9 @param in_file  Kiruvchi stream
10 @param total hali qayta ishlanishi lozim umumiy ulush
11 */
12 void process_name(ifstream& in_file, double& total)
13 {
14 string name;
15 int count;
16 double percent;
```

```
17 in_file >> name >> count >> percent;
18
19 if (in_file.fail()) { return; } // Har bir
kiritishdan keyin muvaffaqiyatsizlikni tekshiring
20 if (total > 0) { cout << name << " "; }
21 total = total - percent;
22 }
23
24 int main()
25 {
26 ifstream in_file;
27 in_file.open("babynames.txt");
28 if (in_file.fail()) { return 0; } // Ochgandan so'ng
muvaffaqiyatsizlikni tekshirish
29
30 double boy_total = 50;
31 double girl_total = 50;
32
33 while (boy_total > 0 || girl_total > 0)
34 {
35 int rank;
36 in_file >> rank;
37 if (in_file.fail()) { return 0; }
38
39 cout << rank << " ";
40
41 process_name(in_file, boy_total);
42 process_name(in_file, girl_total);
43
```

```
44  cout << endl; 45 }
46
47 return 0;
48 }
```

Natija

1 Michael Jessica

2 Christopher Ashley 3 Matthew Emily

4 Joshua Sarah

5 Jacob Samantha

6 Nicholas Amanda

7 Andrew Brittany

8 Daniel Elizabeth

9 Tyler Taylor

10 Joseph Megan

...

68 Dustin Gabrielle 69 Noah Katie

70 Caitlin

71 Lindsey

...

150 Hayley

151 Rebekah

152 Jocelyn

153 Cassidy

Nazorat savollari

1. Agar in_file.open("")ni belgilasangiz nima sodir bo;ladi?
2. Quyidagi kod bilan nima xatolik bor?

ifstream out_file;

out_file.open("output.txt");

out_file << "Hello, World!" << endl;

3. Quyidagi funksiya bilan nima xatolik bor? double sum(ifstream in)

```
{
 double total = 0;
 double input;
 while (in >> input) { total = total + input; }
 return total;
}
```

4. Qanday qilib babynames.cpp dasturnini o'zgartirasiz, shuningdek siz aholi orasida keng tarqalgan ismlarni bilib olasiz?

5. QAnday qilib babynames.cpp dasturini ozgartirasiz shuningdek dastur output 00 bo'lib saqlanadimi

6-Ma’ruza. Binary fayllar bilan ishlash operatorlari

Ma’ruza rejasi:

6.1 Matn va binar fayllar;

6.2 O‘qish-yozish oqimlari. Standart oqimlar;

6.3 Belgilarni o‘qish-yozish funksiyalari;

6.4 Satrlarni o‘qish - yozish funksiyalari;

6.5 Fayldan o‘qish-yozish funksiyalari;

6.6 Formatli o‘qish va yozish funksiyalari;

6.7 Fayl ko‘rsatkichini boshqarish funksiyalari

Kalit so’zlar:*, ro’yxat, manzil, nolinchi ko’rchsatkich, tugun, adres olish &, bo’shatish, ko’rsatkich, virtual destruktor, xotira, xotira chiqishi, destruktor, toifani o’zlashtirish, resurslar chiqishi, a’zo destruktori.*

6.1. Matn va binar fayllar

C++ tili S tilidan o‘qish-yozish amalini bajaruvchi standart funksiyalar kutubxonasini vorislik bo‘yicha olgan. Bu funksiyalar <stdio.h> sarlavha faylida e’lon qilingan. O‘qish-yozish amallari fayllar bilan bajariladi. Fayl matn yoki binar (ikkilik) bo‘lishi mumkin.

Matn fayl - ASCII kodidagi belgilar bilan berilganlar maj-muasi. Belgilar ketma-ketligi satrlarga bo‘lingan bo‘ladi va satr-ning tugash alomati sifatida CR (karetkani qaytarish yoki ‘\r’) LF (satrni o‘tkazish yoki ‘\n’) belgilar juftligi hisoblanadi. Matn fayldan berilganlarni o‘qishda bu belgilar juftligi bitta CR belgisi bilan almashtiriladi va aksincha, yozishda CR belgisi ikkita CR va LF belgilariga almashtiriladi. Fayl oxiri #26 (^Z) belgisi bilan belgilanadi.

Matn faylga boshqacha ta’rif berish ham mumkin. Agar faylni matn tahririda ekranga chiqarish va o‘qish mumkin bo‘lsa, bu matn fayl. Klaviatura ham компьyuterga faqat matnlarni jo‘natadi. Boshqacha aytganda programma

tomonidan ekranga chiqariladigan barcha ma’lumotlarni stdout nomidagi matn fayliga chiqarilmoqda deb qarash mumkin. Xuddi shunday klaviaturadan o‘qilayotgan har qanday berilganlarni matn faylidan o‘qilmoqda deb hisoblanadi.

Matn fayllarining komponentalari *satrlar* deb nomlanadi. Satrlar uzluksiz joylashib, turli uzunlikda va bo‘sh bo‘lishi mumkin. Faraz qilaylik, T matn fayli 4 satrdan iborat bo‘lsin:

1-satr#13#10	2- satr uzunroq #13#10	#13#10	4-satr#13#10#26

12.3-rasm. To‘rtta satrdan tashkil topgan matn fayli

Matnni ekranga chiqarishda satr oxiridagi #13#10 boshqaruv belgilari juftligi kursorni keyingi qatorga tushiradi va uni satr boshiga olib keladi. Bu matn fayl ekranga chop etilsa, uning ko‘rinishi quyidagicha bo‘ladi:

1- satr[13][10]

2- satr uzunroq[13][10]

[13][10]

4- satr[13][10]

[26]

Matndagi [n] - n- kodli boshqaruv belgisini bildiradi. Odatda matn tahrirlari bu belgilarni ko‘rsatmaydi.

Binar fayllar - bu oddiygina baytlar ketma-ketligi. Odatda binar fayllardan berilganlarni foydalanuvchi tomonidan bevosita «ko‘rish» zarur bo‘lmagan hollarda ishlatiladi. Binar fayllardan o‘qish-yozishda baytlar ustida hech qanday konvertatsiya amallari baja-rilmaydi.

6.2. O‘qish-yozish oqimlari. Standart oqimlar

Oqim tushunchasi berilganlarni faylga o‘qish-yozishda ularni belgilar ketma-ketligi yoki oqimi ko‘rinishida tasavvur qilishdan kelib chiqqan. Oqim ustida quyidagi amallarni bajarish mumkin:

- oqimdan berilganlar blokini operativ xotiraga o‘qish;
- operativ xotiradagi berilganlar blokini oqimga chiqarish;

- oqimdagi berilganlar blokini yangilash;
- oqimdan yozuvni o‘qish;
- oqimga yozuvni chiqarish.

Oqim bilan ishlaydigan barcha funksiyalar buferli, format-lashgan yoki formatlashmagan o‘qish-yozishni ta’minlaydi.

Programma ishga tushganda o‘qish-yozishning quyidagi standart oqimlar ochiladi:

stdin - o‘qishning standart vositasi;

stdout - yozishning standart vositasi;

stderr - xatolik haqida xabar berishning standart vositasi;

stdprn - qog‘ozga chop qilishning standart vositasi;

stdaux - standart yordamchi qurilma.

Kelishuv bo‘yicha stdin - foydalanuvchi klaviaturasi, stdout va stderr - terminal (ekran), stdprn - printer bilan, hamda stdaux - компьyuter yordamchi portlariga bog‘langan hisoblanadi. Berilgan-larni o‘qish-yozishda stderr va stdaux oqimidan boshqa oqimlar bufer-lanadi, ya’ni belgilar ketma-ketligi operativ xotiraning bufer deb nomlanuvchi sohasida vaqtincha jamlanadi. Masalan, belgilarni tashqi qurilmaga chiqarishda belgilar ketma-ketligi buferda jamlanadi va bufer to‘lgandan keyingina tashqi qurilmaga chiqariladi.

Hozirdagi operatsion sistemalarda klaviatura va displeylar matn fayllari sifatida qaraladi. Haqiqatdan ham berilganlarni klaviaturadan programmaga kiritish (o‘qish) mumkin, ekranga esa chiqarish (yozish) mumkin. Programma ishga tushganda standart o‘qish va yozish oqimlari o‘rniga matn fayllarni tayinlash orqali bu oqim-larni qayta aniqlash mumkin. Bu holatni *o‘qishni (yozishni) qayta adreslash ro‘y berdi* deyiladi. O‘qish uchun qayta adreslashda ‘<’ belgisidan, yozish uchun esa ‘>’ belgisidan foydalaniladi. Misol uchun gauss.exe bajariluvchi programma berilganlarni o‘qishni klaviaturadan emas, balki massiv.txt faylidan amalga oshirish zarur bo‘lsa, u buyruq satrida quyidagi ko‘rinishda yuklanishi zarur bo‘ladi:

gauss.exe < massiv.txt

Agar programma natijasini natija.txt fayliga chiqarish zarur bo‘lsa

gauss.exe > natija.txt

satri yoziladi.

Va nihoyat, agar berilganlarni massiv.txt faylidan o‘qish va natijani natija.txt fayliga yozish uchun

gauss.exe < massiv.txt > natija.txt

buyruq satri teriladi.

Umuman olganda, bir programmaning chiqish oqimini ikkinchi programmaning kirish oqimi bilan bog‘lash mumkin. Buni *konveyrli jo‘natish* deyiladi. Agar ikkita junat.exe programmasi qabul.exe programmasiga berilganlarni jo‘natishi kerak bo‘lsa, u holda ular o‘rtasiga ‘|’ belgi qo‘yib yoziladi:

junat.exe | qabul.exe

Bu ko‘rinishdagi programmalar o‘rtasidagi konveyrli jo‘na-tishni operatsion sistemaning o‘zi ta’minlaydi.

Belgilarni o‘qish-yozish funksiyalari

Belgilarni o‘qish-yozish funksiyalari makros ko‘rinishida amalga oshirilgan.

getc() makrosi tayinlangan oqimdan navbatdagi belgini qayta-radi va kirish oqimi ko‘rsatkichini keyingi belgini o‘qishga mos-lagan holda oshiradi. Agar o‘qish muvaffaqiyatli bo‘lsa getc() funksiyasi ishorasiz int ko‘rinishidagi qiymatni, aks holda EOF qaytaradi. Ushbu funksiya prototipi quyidagicha:

int getc(FILE * stream)

EOF identifikator makrosi

#define EOF(-1)

ko‘rinishida aniqlangan va o‘qish-yozish amallarida fayl oxirini belgilash uchun xizmat qiladi. EOF qiymati ishorali char turida deb hisoblanadi. SHu sababli o‘qish-yozish jarayonida unsigned char turidagi belgilar ishlatilsa, EOF makrosini ishlatib bo‘lmaydi.

Navbatdagi misol getc() makrosini ishlatishni namoyon qiladi.

```
#include <iostream.h>
#include <stdio.h>
int main()
{
 char ch;
 cout<<”Belgini kiriting: “;
 ch=getc(stdin);
 cout<<”Siz ”<<ch<<” belgisini kiritdingiz.\n”;
 return 0;
}
```

getc() makrosi aksariyat holatlarda stdin oqimi bilan ishlatilganligi sababli, uning getc(stdin) ko‘rinishiga ekvivalent bo‘lgan int getchar() makrosi aniqlangan. YUqoridagi misolda «ch=getc(stdin);» qatorini «ch=getchar();» qatori bilan almashtirish mumkin.

Belgini oqimga chiqarish uchun putc() makrosi aniqlangan va uning prototipi

int putc(int c, FILE*stream)

ko‘rinishida aniqlangan. putc() funksiyasi stream nomi bilan berilgan oqimga s belgini chiqaradi. Funksiya qaytaruvchi qiymati sifatida int turiga aylantirilgan s belgi bo‘ladi. Agar belgini chiqarishda xatolik ro‘y bersa EOF qaytariladi.

putc() funksiyasini standart stdout oqimi bilan bog‘langan holati - putc(c,strout) uchun putchar(c) makrosi aniqlangan.

Satrlarni o‘qish - yozish funksiyalari

Oqimdan satrni o‘qishga mo‘ljallangan gets() funksiyasining prototipi

char * gets(char *s);

ko‘rinishida aniqlangan. gets() funksiyasi standart oqimdan satrni o‘qiydi va uni s o‘zgaruvchisiga joylashtiradi. Joylashtirish payti-da oqimdagi ‘\n’ belgisi ‘\0’ belgisi bilan almashtiriladi. Bu funksiyani ishlatishda o‘qilayotgan satrning uzunligi s satr uchun ajratilgan joy uzunligidan oshib ketmasligini nazorat qilish kerak bo‘ladi.

puts() funksiyasi

int puts(const char *s);

ko‘rinishida bo‘lib, u standrat oqimga argumentda ko‘rsatilgan satrni chiqaradi. Bunda satr oxiriga yangi satrga o‘tish belgisi ‘\n’ qo‘shiladi. Agar satrni oqimga chiqarish muvaffaqiyatli bo‘lsa puts() funksiyasi manfiy bo‘lmagan sonni, aks holda EOF qaytaradi.

Satrni o‘qish-yozish funksiyalarini ishlatishga misol tariqa-sida qo‘yidagi programmani keltirish mumkin.

```
#include <stdio.h>
int main()
{
 char *s;
 puts(”Satrni kiriting: “);
 gets(s);
 puts(”Kiritilgan satr: ”);
 puts(s);
 return 0;
}
```

Formatli o‘qish va yozish funksiyalari

Formatli o‘qish va yozish funksiyalari - scanf() va printf() funksiyalari S tilidan vorislik bilan olingan. Bu funksiyalarni ishlatish uchun «stdio.h» sarlavha faylini programmaga qo‘shish kerak bo‘ladi.

Formatli o‘qish funksiyasi scanf() quyidagi prototipga ega:

int scanf(const char * <format>[<adres>,...])

Bu funksiya standart oqimdan berilganlarni formatli o‘qishni amalga oshiradi. Funksiya, kirish oqimidagi maydonlar ketma-ket-ligi ko‘rinishidagi belgilarni birma-bir o‘qiydi va har bir maydonni <format> satrida keltirilgan format aniqlashtiruvchi-siga mos ravishda formatlaydi. Oqimdagi har bir maydonga format aniqlashtiruvchisi va natija joylashadigan o‘zgaruvchining adresi bo‘lishi shart.

Boshqacha aytganda, oqimdagi maydon (ajratilgan belgilar ketma-ketligi) ko‘rsatilgan formatdagi qiymatga akslanti-riladi va o‘zgaruvchi bilan nomlangan xotira bo‘lagiga joylashti-riladi (saqlanadi). Funksiya oqimdan berilganlarni o‘qish jarayo-nini «to‘ldiruvchi belgini» uchratganda yoki oqim tugashi natijasida to‘xtatishi mumkin. Oqimdan berilganlarni o‘qish muvafaqqiyatli bo‘lsa, funksiya muvafaqqiyatli aylantirilgan va xotiraga saqlangan maydonlar sonini qaytaradi. Agar hech bir maydonni saqlash imkoni bo‘lmagan bo‘lsa, funksiya 0 qiymatini qaytaradi. Oqim oxiriga kelib qolganda (fayl yoki satr oxiriga) o‘qishga harakat bo‘lsa, funksiya EOF qiymatini qaytaradi.

Formatlash satri - <format> belgilar satri bo‘lib, u uchta toifaga bo‘linadi:

- to‘ldiruvchi belgilar;
- to‘ldiruvchi belgilardan farqli belgilar;
- format aniqlashtiruvchilari.

To‘ldiruvchi-belgilar – bu probel, ‘\t’, ‘\n’ belgilari. Bu belgilar formatlash satridan o‘qiladi, lekin saqlanmaydi.

To‘ldiruvchi belgilardan farqli belgilar – bu qolgan barcha ASCII belgilari, ‘%’ belgisilan tashqari. Bu belgilar formatlash satridan o‘qiladi, lekin saqlanmaydi.

12.1–jadval. Format aniqlashtiruvchilari va ularning vazifasi

Komponenta	**Bo‘lishi shart yoki yo‘q**	**Vazifasi**
[*]	Yo‘q	Navbatdagi ko‘rib chiqilayotgan maydon qiymatini o‘zgaruvchiga o‘zlashtirmaslik belgisi. Kirish oqimidagi maydon ko‘rib chiqiladi, lekin o‘zgaruvchida saqlanmaydi.
[<kenglik>]	Yo‘q	Maydon kengligini aniqlashtiruvchisi. O‘qiladigan belgilarning maksimal sonini aniqlaydi. Agar oqimda to‘ldiruv–chi belgi yoki almashtirilmaydigan belgi

		uchrasi funksiya nisbatan kam sondagi belgilarni o‘qishi mumkin.
[F|N]	Yo‘q	O‘zgaruvchi ko‘rsatkichining (adresining) modifikatori: F– far pointer; N- near pointer
[h|l|L]	Yo‘q	Argument turining modifikatori. <tur belgisi> bilan aniqlangan o‘zgaruvchining qisqa (short - h) yoki uzun (long –l,L) ko‘ri–nishini aniqlaydi.
<tur belgisi>	Ha	Oqimdagi belgilarni almashtiriladigan tur belgisi

Format aniqlashtiruvchilari – oqim maydonidagi belgilarni ko‘rib chiqish, o‘qish va adresi bilan berilgan o‘zgaruvchilar turiga mos ravishda almashtirish jarayonini boshqaradi. Har bir format aniqlashtiruvchisiga bitta o‘zgaruvchi adresi mos kelishi kerak. Agar format aniqlashtiruvchilari soni o‘zgaruvchilardan ko‘p bo‘lsa, nati-ja nima bo‘lishini oldindan aytib bo‘lmaydi. Aks holda, ya’ni o‘zgaruvchilar soni ko‘p bo‘lsa, ortiqcha o‘zgaruvchilar inobatga olinmaydi.

Format aniqlashtiruvchisi quyidagi ko‘rinishga ega:

% [*][<kenglik>] [F|N] [h|l|L] <tur belgisi>

Format aniqlashtiruvchisi ‘%’ belgisidan boshlanadi va undan keyin 12.1–jadvalda keltirilgan shart yoki shart bo‘lmagan komponen-talar keladi.

12.2–jadval. Almashtiriladigan tur alomati belgilari

Tur alomati	**Kutilayotgan qiymat**	**Argument turi**
Son turidagi argument		
d, D	O‘nlik butun	int * arg yoki long * arg
E,e	Suzuvchi nuqtali son	float * arg
F	Suzuvchi nuqtali son	float * arg
G,g	Suzuvchi nuqtali son	float * arg
O	Sakkizlik son	int * arg

O	Sakkizlik son	long * arg
I	O‘nlik, sakkizlik va o‘n oltilik butun son	int * arg
I	O‘nlik, sakkizlik va o‘n oltilik butun son	long * arg
U	Ishorasiz o‘nlik son	Unsigned int * arg
U	Ishorasiz o‘nlik son	Unsigned long * arg
X	O‘n oltilik son	int * arg
X	O‘n oltilik son	int * arg
Belgilar		
S	Satr	char * arg (belgilar massivi)
C	Belgi	char * arg (belgi uchun maydon kengligi berilishi mumkin (masalan, %4s). N belgidan tashkil topgan belgilar massiviga ko‘rsatkich: char arg[N])
%	‘%’ belgisi	Hech qanday almashtirishlar baja–rilmaydi, ‘%’ belgisi saqlanadi.
Ko‘rsatkichlar		
N	int * arg	%n argumentigacha muvaffaqiyatli o‘qilgan belgilar soni, aynan shu int ko‘rsatkichi bo‘yicha adresda saqlanadi.
P	YYYY:ZZZZ yoki ZZZZ ko‘rinishidagi o‘n oltilik	Ob’ektga ko‘rsatkich (far* yoki near*).

Oqimdagi belgilar bo‘lagini almashtiriladigan tur alomati-ning qabul qilishi mumkin bo‘lgan belgilar 12.2-jadvalda keltirilgan.

12.3–jadval. Format aniqlashtiruvchilari va ularning vazifasi

Komponenta	**Bo‘lishi shart yoki yo‘q**	**Vazifasi**

[bayroq]	Yo‘q	Bayroq belgilari. CHiqarilayotgan qiy-matni chapga yoki o‘nga tekislashni, sonning ishorasini, o‘nlik kasr nuqta-sini, oxirdagi nollarni, sakkizlik va o‘n oltilik sonlarning alomatlarni chop etishni boshqaradi. Masalan, ‘-‘ bayrog‘i qiymatni ajratilgan o‘ringa nisbatan chapdan boshlab chiqarishni va kerak bo‘lsa o‘ngdan probel bilan to‘ldirishni bildiradi, aks holda chap tomondan probellar bilan to‘ldiradi va davomiga qiymat chiqariladi.
[<kenglik>]	Yo‘q	Maydon kengligini aniqlashtiruvchisi. CHiqariladigan belgilarning minimal sonini aniqlaydi. Zarur bo‘lsa qiymat yozilishidan ortgan joylar probel bilan to‘ldiriladi.
[.<xona>]	Yo‘q	Aniqlik. CHiqariladigan belgilarning maksimal sonini ko‘rsatadi. Sondagi raqamlarning minimal sonini.
[F|N|h|l|L]	Yo‘q	O‘lcham modifikatori. Argumentning qisqa (short - h) yoki uzun (long –l,L) ko‘ri-nishini, adres turini aniqlaydi.
<tur belgisi>	Ha	Argument qiymati almashtiriladigan tur alomati belgisi

Formatli yozish funksiyasi printf() quyidagi prototipga ega:

int printf(const char * <format>[,<argument>,...])

Bu funksiya standart oqimga formatlashgan chiqarishni amalga oshiradi. Funksiya argumentlar ketma-ketligidagi har bir argument qiymatini qabul qiladi va unga <format> satridagi mos format aniqlashtiruvchisini qo‘llaydi va oqimga chiqaradi.

12.4–jadval. printf() funksiyasining almashtiriladigan tur belgilari

Tur alomati	**Kutilayotgan qiymat**	**CHiqish formati**
Son qiymatlari		
D	Butun son	Ishorali o‘nlik butun son

I	Butun son	Ishorali o‘nlik butun son
O	Butun son	Ishorasiz sakkizlik butun son
U	Butun son	Ishorasiz o‘nlik butun son
X	Butun son	Ishorasiz o‘n oltilik butun son (a,b,c,d,e,f belgilari ishlatiladi)
X	Butun son	Ishorasiz o‘n oltilik butun son (A,B,C,D,E,F belgilari ishlatiladi)
F	Suzuvchi nuqtali son	[-]dddd.dddd ko‘rinishidagi suzuvchi nuqtali son
E	Suzuvchi nuqtali son	[-]d.dddd yoki e[+/-]ddd ko‘rini–shidagi suzuvchi nuqtali son
G	Suzuvchi nuqtali son	Ko‘rsatilgan aniqlikka mos e yoki f shaklidagi suzuvchi nuqtali son
E, G	Suzuvchi nuqtali son	Ko‘rsatilgan aniqlikka mos e yoki f shaklidagi suzuvchi nuqtali son. e format uchun ‘E’ chop etiladi.
Belgilar		
S	Satrga ko‘rsatkich	0-belgisi uchramaguncha yoki ko‘rsa-tilgan aniqlikka erishilmaguncha belgilar oqimga chiqariladi.
C	Belgi	Bitta belgi chiqariladi
%	Hech nima	‘%’ belgisi oqimga chiqariladi.
Ko‘rsatkichlar		
N	int ko‘rsatkich (int* arg)	%n argumentigacha muvaffaqiyatli chiqarilgan belgilar soni, aynan shu int ko‘rsatkichi bo‘yicha adresda saqlanadi.
P	Ko‘rsatkich	Argumentni YYYY:ZZZZ yoki ZZZZ ko‘rinishidagi o‘n oltilik songa aylantirib oqimga chiqaradi.

Har bir format aniqlashtiruvchisiga bitta o‘zgaruvchi adresi mos kelishi kerak. Agar format aniqlashtiruvchilari soni o‘zgaruvchi-lardan ko‘p bo‘lsa, natijada nima

bo‘lishini oldindan aytib bo‘lmaydi. Aks holda, ya’ni o‘zgaruvchilar soni ko‘p bo‘lsa, ortiqcha o‘zgaruvchilar inobatga olinmaydi. Agar oqimga chiqarish muvaf-faqiyatli bo‘lsa, funksiya chiqarilgan baytlar sonini qaytaradi, aks holda EOF.

printf() funksiyasining <format> satri argumentlarni almash-tirish, formatlash va berilganlarni oqimga chiqarish jarayonini boshqaradi va u ikki turdagi ob’ektlardan tashkil topadi:

oqimga o‘zgarishsiz chiqariladigan oddiy belgilar;

argumentlar ro‘yxatidagi tanlanadigan argumentga qo‘llani-ladigan format aniqlashtiruvchilari.

Format aniqlashtiruvchisi quyidagi ko‘rinishga ega:

% [<bayroq>][<.kenglik>] [.<xona>][F|N|h|l|L] <tur belgisi>

Format aniqlashtiruvchisi ‘%’ belgisidan boshlanadi va undan keyin 12.3–jadvalda keltirilgan shart yoki shart bo‘lmagan kompo-nentalar keladi.

Almashtiriladigan tur belgisining qabul qilishi mumkin bo‘lgan belgilar 12.4-jadvalda keltirilgan.

Berilganlar qiymatlarini oqimdan o‘qish va oqimga chiqarishda scanf() va prinf() funksiyalaridan foydalanishga misol:

```
#include <stdio.h>
int main()
{
 int bson, natija;
 float hson;
 char blg, satr[81];
 printf("\nButun va suzuvchi nuqtali sonlarni,");
 printf("\nbelgi hamda satrni kiriting\n");
 natija=scanf("%d %f %c %s", &bson,
&hson,&blg,satr);
 printf("\nOqimdan %d ta qiymat o'qildi ",natija);
 printf("va ular quyidagilar:");
```

```
 printf("\n %d %f %c %s \n",bson, hson, blg, satr);
 return 0;
}
```

Programma foydalanuvchidan butun va suzuvchi nuqtali sonlarni, belgi va satrni kiritishni so‘raydi. Bunga javoban foydalanuvchi tomonidan

10 12.35 A Satr

qiymatlari kiritilsa, ekranga

Oqimdan 4 ta qiymat o'qildi va ular quyidagilar:

10 12.35 A Satr

satrlari chop etiladi.

Fayldan o‘qish-yozish funksiyalari

Fayl oqimi bilan o‘qish-yozish amalini bajarish uchun fayl oqimini ochish zarur. Bu ishni, prototipi

FILE * fopen(const char* filename, const char *mode);

ko‘rinishida aniqlangan fopen() funksiyasi orqali amalga oshiri-ladi. Funksiya filename nomi bilan faylni ochadi, u bilan oqimni bog‘laydi va oqimni identifikatsiya qiluvchi ko‘rsatkichni javob tariqasida qaytaradi. Faylni ochish muvaffaqiyatsiz bo‘lganligini fopen() funksiyasining NULL qiymatli javobi bildiradi.

Parametrlar ro‘yxatidagi ikkinchi - mode parametri faylni ochish rejimini aniqlaydi. U qabul qilishi mumkin bo‘lgan qiymat-lar 12.5- jadvalda keltirilgan.

12.5-jadval. Fayl ochish rejimlari

Mode qiymati	**Fayl ochilish holati tavsifi**
R	Fayl faqat o‘qish uchun ochiladi
W	Fayl yozish uchun ochiladi. Agar bunday fayl mavjud bo‘lsa, u qaytadan yoziladi (yangilanadi).
A	Faylga yozuvni qo‘shish rejimi. Agar fayl mavjud bo‘lsa, fayl uning oxiriga yozuvni yozish uchun ochiladi, aks holda yangi fayl yaratiladi va yozish rejimida ochiladi.

r+	Mavjud fayl o‘zgartirish (o‘qish va yozish) uchun ochiladi.
w+	YAngi fayl yaratilib, o‘zgartirish (o‘qish va yozish) uchun ochiladi. Agar fayl mavjud bo‘lsa, undagi oldingi yozuvlar o‘chiriladi va u qayta yozishga tayyorlanadi.
a+	Faylga yozuvni qo‘shish rejimi. Agar fayl mavjud bo‘lsa, uning oxiriga (EOF alomatidan keyin) yozuvni yozish (o‘qish) uchun ochiladi, aks holda yangi fayl yaratiladi va yozish rejimida ochiladi.

Matn fayli ochilayotganligini bildirish uchun fayl ochilish rejimi satriga ‘t’ belgisini qo‘shib yozish zarur bo‘ladi. Masalan, matn fayl o‘zgartirish (o‘qish va yozish) uchun ochilayotganligini bildirish uchun “rt+” satri yozish kerak bo‘ladi. Xuddi shunday binar fayllar ustida ishlash uchun ‘b’ belgisini ishlatish kerak. Misol uchun fayl ochilishining “wb+” rejimi binar fayl yangilanishini bildiradi.

Fayl o‘zgartirish (o‘qish-yozish) uchun ochilganda, berilganlarni oqimdan o‘qish, hamda oqimga yozish mumkin. Biroq yozish amalidan keyin darhol o‘qib bo‘lmaydi, buning uchun o‘qish amalidan oldin fseek() yoki rewind() funksiyalari chaqirilishi shart.

Faraz qilaylik «C:\\USER\\TALABA\\iat1kuz.txt» nomli matn faylni o‘qish uchun ochish zarur bo‘lsin. Bu talab

FILE *f=fopen(”C:\\USER\\TALABA\\iat1kuz.txt”,”r+”);

ifodasini yozish orqali amalga oshiraladi. Natijada diskda mavjud bo‘lgan fayl programmada f o‘zgaruvchisi nomi bilan aynan bir narsa deb tushuniladi. Boshqacha aytganda, programmada keyinchalik f ustida bajarilgan barcha amallar, diskdagi «iat1kuz.txt» fayli ustida ro‘y beradi.

Fayl oqimi bilan ishlash tugagandan keyin u albatta yopilishi kerak. Buning uchun fclose() funksiyasidan foydalaniladi. Funksiya prototipi quyidagi ko‘rinishga ega:

int fclose(FILE * stream);

fclose() funksiyasi oqim bilan bog‘liq buferlarni tozalaydi (masalan, faylga yozish ko‘rsatmalari berilishi natijasida buferda yig‘ilgan berilganlarni diskdagi

faylga ko‘chiradi) va faylni yopadi. Agar faylni yopish xatolikka olib kelsa, funksiya EOF qiymatini, normal holatda 0 qiymatini qaytaradi.

fgetc() funksiyasi prototipi

int fgetc(FILE *stream);

ko‘rinishida aniqlangan bo‘lib, fayl oqimidan belgini o‘qishni amalga oshiradi. Agar o‘qish muvafffaqiyatli bo‘lsa, funksiya o‘qilgan belgini int turidagi ishorasiz butun songa aylantiradi. Agar fayl oxirini o‘qishga harakat qilinsa yoki xatolik ro‘y bersa, funksiya EOF qiymatini qaytaradi.

Ko‘rinib turibdiki, getc() va fgetc() funksiyalari deyarli bir xil ishni bajaradi, farqi shundaki, getc() funksiyasi belgini standart oqimdan o‘qiydi. Boshqacha aytganda, getc() funksiyasi, fayl oqimi standart qurilma bo‘lgan fgetc() funksiyasi bilan aniqlangan makrosdir.

fputc() funksiyasi

int fputc(int c, FILE *stream);

prototipi bilan aniqlangan. fputc() funksiyasi fayl oqimiga argu-mentda ko‘rsatilgan belgini yozadi (chiqaradi) va u amal qilishida putc() funksiyasi bilan bir xil.

Fayl oqimidan satr o‘qish uchun

char * fgets(char * s, int n, FILE *stream)

prototipi bilan fgets() aniqlangan. fgets() funksiyasi fayl oqimidan belgilar ketma-ketligini s satriga o‘qiydi. Funksiya o‘qish jarayoni-ni oqimdan n-1 belgi o‘qilgandan keyin yoki keyingi satrga o‘tish belgisi (‘\n’) uchraganda to‘xtatadi. Oxirgi holda ‘\n’ belgisi ham s satrga qo‘shiladi. Belgilarni o‘qish tugagandan keyin s satr oxiriga, satr tugash alomati ‘\0’ belgisi qo‘shiladi. Agar satrni o‘qish muvaf-faqiyatli bo‘lsa, funksiya s argument ko‘rsatadigan satrni qaytaradi, aks holda NULL.

Fayl oqimiga satrni fputs() funksiyasi yordamida chiqarish mum-kin. Bu funksiya prototipi

int fputs (const char *s, FILE *stream);

ko‘rinishida aniqlangan. Satr oxiridagi yangi satrga o‘tish belgisi va terminatorlar oqimga chiqarilmaydi. Oqimga chiqarish muvaffa-qiyatli bo‘lsa, funksiya nomanfiy son qaytaradi, aks holda EOF.

foef() funksiyasi aslida makros bo‘lib, fayl ustida o‘qish-yozish amallari bajarilayotganda fayl oxiri belgisi uchragan yoki yo‘qligini bildiradi. Funksiya

int feof(FILE *stream);

prototipiga ega bo‘lib u fayl oxiri belgisi uchrasa, noldan farqli sonni qaytaradi, boshqa holatlarda 0 qiymatini qaytaradi.

Quyida keltirilgan misolda faylga yozish va o‘qishga amallari ko‘rsatilgan.

```
#include <iostream.h>
#include <stdio.h>
int main()
{
 char s;
 FILE *in,*out;
 if((in=fopen("D:\\USER\\TALABA.TXT","rt"))==NULL)
 {
  cout<<"Talaba.txt faylini ochilmadi!!\n";
  return 1;
 }

if((out=fopen("D:\\USER\\TALABA.DBL","wt+"))==NULL)
 {
  cout<<"Talaba.dbl faylini ochilmadi!!\n";
  return 1;
 }
 while (!feof(in))
 {
  char c=fgetc(in);
```

```
    cout<<c;
    fputc(c,out);
   }
   fclose(in);
   fclose(out);
   return 0;
}
```

Programmada «talaba.txt» fayli matn fayli sifatida o‘qish uchun ochilgan va u in o‘zgaruvchisi bilan bog‘langan. Xuddi shunday, «talaba.dbl» matn fayli yozish uchun ochilgan va out bilan bog‘langan. Agar fayllarni ochish muvaffaqiyatsiz bo‘lsa, mos xabar beriladi va programma o‘z ishini tugatadi. Keyinchalik, toki in fayli oxiriga etmaguncha, undan belgilar o‘qiladi va ekranga, hamda out fayliga chiqariladi. Programma oxirida ikkita fayl ham yopiladi.

Masala. G‘alvirli tartiblash usuli.

Berilgan x vektorini pufakcha usulida kamaymaydigan qilib tartiblash quyidagicha amalga oshiriladi: massivning qo‘shni elementlari x_k va x_{k+1} (k=1,..,n-1) solishtiriladi. Agar $x_k>x_{k+1}$ bo‘lsa, u holda bu elementlar o‘zaro o‘rin almashadi. SHu yo‘l bilan birinchi o‘tishda eng katta element vektorning oxiriga joylashadi. Keyingi qadamda vektor boshidan n-1 o‘rindagi elementgacha yuqorida qayd qilingan yo‘l bilan qolgan elementlarning eng kattasi n-1 o‘ringa joylashtiriladi va h.k.

G‘alvirli tartiblash usuli pufakchali tartiblash usuliga o‘xshash, lekin x_k va x_{k+1} (k=1,2,3,..,n-1) elementlar o‘rin almashgandan keyin «g‘alvirdan» o‘tkazish amali qo‘llaniladi: chap tomondagi kichik element imkon qadar chap tomonga tartiblash saqlangan holda ko‘chiriladi. Bu usul oddiy pufakchali tartiblash usuliga nisbatan tez ishlaydi.

Programma matni:

```
#include <stdio.h>
#include <alloc.h>
```

```
int * Pufakchali_Tartiblash(int*,int);
int main()
{
 char fnomi[80];
 printf("Fayl nomini kiriting:");
 scanf("%s", &fnomi);
 int Ulcham,i=0,* Massiv;
 FILE * f1, *f2;
 if((f1=fopen(fnomi,"rt"))==NULL)
 {
  printf("Xato:%s fayli ochilmadi!",fnomi);
  return 1;
 }
 fscanf(f1,"%d",&Ulcham);
 Massiv=(int *)malloc(Ulcham*sizeof(int));
 while(!feof(f1))
 fscanf(f1,"%d",&Massiv[i++]);
 fclose(f1);
 Massiv=Pufakchali_Tartiblash(Massiv,Ulcham);
 f2=fopen("natija.txt","wt");
 fprintf(f2,"%d%c",Ulcham,' ');
 for(i=0; i<Ulcham; i++)
 fprintf(f2,"%d%c",Massiv[i],' ');
 fclose(f2);
 return 0;
}
int * Pufakchali_Tartiblash(int M[],int n)
{
     int almashdi=1, vaqtincha;
     for(int i=0; i<n-1 && almashdi;i++)
```

```
{
 almashdi=0;
 for(int j=0; j<n-i-1;j++)
 if (M[j]>M[j+1])
 {
  almashdi=1;
  vaqtincha=M[j];
  M[j]=M[j+1];
  M[j+1]=vaqtincha;
  int k=j;
  if(k)
   while(k && M[k]>M[k-1])
   {
    vaqtincha=M[k-1];
    M[k-1]=M[k];
    M[k]=vaqtincha;
    k--;
    }
  }
 }
 return M;
}
```

Programmada berilganlarni oqimdan o‘qish yoki oqimga chiqarishda fayldan formatli o‘qish - fscanf() va yozish - fprintf() funksiyalaridan foydalanilgan. Bu funksiyalarning mos ravishda scanf() va printf() funksiyalaridan farqi - ular berilganlarni birinchi argument sifatida beriladigan matn fayldan o‘qiydi va yozadi.

Nomi foydalanuvchi tomonidan kiritiladigan f1 fayldan butun sonlar massivining uzunligi va qiymatlari o‘qiladi va tartiblangan massiv f2 faylga yoziladi.

Vektorni tartiblash Pufakchali_Tartiblash() funksiyasi tomoni-dan amalga oshiriladi. Unga vektor va uning uzunligi kiruvchi parametr bo‘ladi va tartiblangan vektor funksiya natijasi sifatida qaytariladi.

Navbatdagi ikkita funksiya fayl oqimidan formatlashmagan o‘qish-yozishni amalga oshirishga mo‘ljallangan.

fread() funksiyasi quyidagi prototipga ega:

size_t fread(void * ptr, size_t size, size_t n,

FILE *stream);

Bu funksiya oqimdan ptr ko‘rsatib turgan buferga, har biri size bayt bo‘lgan n ta berilganlar blokini o‘qiydi. O‘qish muvaffaqiyatli bo‘lsa, funksiya o‘qilgan bloklar sonini qaytaradi. Agar o‘qish jarayo-nida fayl oxiri uchrab qolsa yoki xatolik ro‘y bersa, funksiya to‘liq o‘qilgan bloklar sonini yoki 0 qaytaradi.

fwrite() funksiyasi prototipi

size_t fwrite(const void*ptr,size_t size,

size_t n,FILE *stream);

ko‘rinishi aniqlangan. Bu funksiya ptr ko‘rsatib turgan buferdan, har biri size bayt bo‘lgan n ta berilganlar blokini oqimga chiqaradi. YOzish muvaffaqiyatli bo‘lsa, funksiya yozilgan bloklar sonini qayta-radi. Agar yozish jarayonida xatolik ro‘y bersa, funksiya to‘liq yozilgan bloklar sonini yoki 0 qaytaradi.

Fayl ko‘rsatkichini boshqarish funksiyalari

Fayl ochilganda, u bilan «stdio.h» sarlavha faylida aniqlangan FILE strukturasi bog‘lanadi. Bu struktura har bir ochilgan fayl uchun joriy yozuv o‘rnini ko‘rsatuvchi hisoblagichni - fayl ko‘rsatkichini mos qo‘yadi. Odatda fayl ochilganda ko‘rsatkich qiymati 0 bo‘ladi. Fayl ustida bajarilgan har bir amaldan keyin ko‘rsatkich qiymati o‘qilgan yoki yozilgan baytlar soniga oshadi. Fayl ko‘rsatkichini boshqarish funksiyalari - fseek(), ftell() va rewind() funksiyalari fayl ko‘rsatkichini o‘zgartirish, qiymatini olish imkonini beradi.

ftell() funksiyasining prototipi

long int ftell(FILE *stream);

ko‘rinishida aniqlangan bo‘lib, argumentda ko‘rsatilgan fayl bilan bog‘langan fayl ko‘rsatkichi qiymatini qaytaradi. Agar xatolik ro‘y bersa funksiya -1L qiymatini qaytaradi.

int fseek(FILE *stream, long offset, int from);

prototipiga ega bo‘lgan fseek() funksiyasi stream fayli ko‘rsatkichini from joyiga nisbatan offset bayt masofaga surishni amalga oshiradi. Matn rejimidagi oqimlar uchun offset qiymati 0 yoki ftell() funksiyasi qaytargan qiymat bo‘lishi kerak. from parametri quyidagi qiymat-larni qabul qilishi mumkin:

SEEK_SET (=0) - fayl boshi;

SEEK_CUR (=1) - fayl ko‘rsatkichining ayni paytdagi qiymati;

SEEK_END (=2) - fayl oxiri.

Funksiya fayl ko‘rsatkichi qiymatini o‘zgartirish muvaffaqiyat-li bo‘lsa, 0 qiymatini, aks holda noldan farqli qiymat qaytaradi.

rewind() funksiyasi

void rewind(FILE *stream);

prototipi bilan aniqlangan bo‘lib, fayl ko‘rsatkichini fayl boshla-nishiga olib keladi.

Quyida keltirilgan programmada binar fayl bilan ishlash ko‘rsatilgan.

```
#include <iostream.h>
#include <stdio.h>
#include <string.h>
struct Shaxs
{
 char Familiya[20];
 char Ism[15];
 char Sharifi[20];
};
int main()
{
```

```
int n,k;
cout<<"Talabalar sonini kiriting: "; cin>>n;
FILE *oqim1,*oqim2;
Shaxs *shaxs1, *shaxs2, shaxsk;
shaxs1=new Shaxs[n];
shaxs2=new Shaxs[n];
if ((oqim1=fopen("Talaba.dat", "wb+"))==NULL)
{
 cout<<"Talaba.dat ochilmadi!!!";
 return 1;
}
for(int i=0; i<n; i++)
{
 cout<<i+1<<"- shaxs ma'lumotlarini kiriting:\n";
 cout<<"Familiysi: "; gets(shaxs1[i].Familiya);
 cout<<"Ismi: "; gets(shaxs1[i].Ism);
 cout<<"Sharifi: "; gets(shaxs1[i].Sharifi);
}
if (n==fwrite(shaxs1,sizeof(Shaxs),n,oqim1))
 cout<<"Berilganlarni yozish amalga oshirildi!\n";
else
{
 cout<<"Berilganlarni yozish amalga oshirilmadi!\n";
 return 3;
}
cout<<" Fayl uzunligi: "<<ftell(oqim1)<<'\n';
fclose(oqim1);
if((oqim2=fopen("Talaba.dat", "rb+"))==NULL)
{
```

```
 cout<<"Talaba.dat o'qishga ochilmadi!!!";
 return 2;
}
if (n==fread(shaxs2,sizeof(Shaxs),n,oqim2))
for(int i=0; i<n; i++)
{
 cout<<i+1<<"- shaxs ma'lumotlari:\n";
 cout<<"Familiysi: "<<shaxs2[i].Familiya<<'\n';
 cout<<"Ismi: "<<shaxs2[i].Ism<<'\n';
 cout<<"Sharifi: "<<shaxs2[i].Sharifi<<'\n';
 cout<<"*****************************\n"; }
else
{
 cout<<"Fayldan o'qish amalga oshirilmadi!\n" ;
 return 4;
}
do
{
 cout<<"Yo'zuv nomerini kiriting (1.."<<n<<"):";
 cin>>k;
 } while (k<0 && k>n);
k--;
cout<<"Oldingi Familiya: ";
cout<<shaxs2[k].Familiya <<'\n';
cout<<"Yangi Familiya: ";
gets(shaxs2[k].Familiya);
if (fseek(oqim2, k*sizeof(Shaxs),SEEK_SET))
{
 cout<<"Faylda"<<k+1;
```

```
 cout<<"-yo'zuvga o'tishda xatolik ro'y berdi???\n";
 return 5;
}
fwrite(shaxs2+k,sizeof(Shaxs),1,oqim2);
fseek(oqim2, k*sizeof(Shaxs),SEEK_SET);
fread(&shaxsk,sizeof(Shaxs),1,oqim2);
cout<<k+1<<"- shaxs ma'lumotlari:\n";
cout<<"Familiysi: "<<shaxsk.Familiya<<'\n';
cout<<"Ismi: "<<shaxsk.Ism<<'\n';
cout<<"Sharifi: "<<shaxsk.Sharifi<<'\n';
fclose(oqim2);
delete shaxs1;
delete shaxs2;
return 0;
}
```

YUqorida keltirilgan programmada, oldin «Talaba.dat» fayli binar fayl sifatida yozish uchun ochiladi va u oqim1 o‘zgaruvchisi bilan bog‘lanadi. SHaxs haqidagi ma’lumotni saqlovchi n o‘lchamli dinamik shaxs1 strukturalar massivi oqim1 fayliga yoziladi, fayl uzunligi chop qilinib fayl yopiladi. Keyin, xuddi shu fayl oqim2 nomi bilan o‘qish uchun ochiladi va undagi berilganlar shaxs2 struk-turalar massiviga o‘qiladi va ekranga chop qilinadi. Programmada fayldagi yozuvni o‘zgartirish (qayta yozish) amalga oshirilgan. O‘zgar-tirish qilinishi kerak bo‘lgan yozuv tartib nomeri foydalanuvchi tomonidan kiritiladi (k o‘zgaruvchisi) va shaxs2 strukturalar massividagi mos o‘rindagi strukturaning Familiya maydoni klavia-turadan kiritilgan yangi satr bilan o‘zgartiriladi. oqim2 fayl ko‘rsatkichi fayl boshidan k*sizeof(Shaxs) baytga suriladi va shaxs2 massivning k - strukturasi (shaxs2+k) shu o‘rindan boshlab faylga yoziladi. Keyin oqim2 fayli ko‘rsatkichi o‘zgartirish kiritilgan yozuv boshiga qaytariladi va bu yozuv shaxsk strukturasiga o‘qiladi hamda ekranga chop etiladi.

Masala. Haqiqiy sonlar yozilgan f fayli berilgan. f fayldagi elementlarning o‘rta arifmetigidan kichik bo‘lgan elementlar miqdorini aniqlansin.

Masalani echish uchun f faylini yaratish va qaytadan uni o‘qish uchun ochish zarur bo‘ladi. YAratilgan faylning barcha elementlarining yig‘indisi s o‘zgaruvchisida hosil qilinadi va u fayl elementlari soniga bo‘linadi. Keyin f fayl ko‘rsatkichi fayl boshiga olib kelinadi va elementlar qayta o‘qiladi va s qiymatidan kichik elementlar soni - k sanab boriladi.

Faylni yaratish va undagi o‘rta arifmetikdan kichik sonlar miqdorini aniqlashni alohida funksiya ko‘rinishida aniqlash mumkin.

Programma matni:

```
#include <iostream.h>
#include <stdio.h>
#include <string.h>
int Fayl_Yaratish()
{
 FILE * f;
 double x;
 // f fayli yangidan hosil qilish uchun ochiladi
 if ((f=fopen("Sonlar.dbl", "wb+"))==NULL)return 0;
 char *satr=new char[10];
 int n=1;
 do
 {cout<<"Sonni kiriting(bo'sh satr tugatish): ";
  gets(satr);
  if(strlen(satr))
  {x=atof(satr);
   fwrite(&x,sizeof(double),n,f);
  }
```

```
     } while(strlen(satr));// satr bo‘sh
bo‘lmasa,takrorlash
      fclose(f);
      return 1;
     }
     int OAdan_Kichiklar_Soni()
     {
      FILE * f;
      double x;
      f=fopen("Sonlar.dbl", "rb+");
      double s=0; // s - f fayl elementlari yig‘indisi
      while (!feof(f))
      {
       if (fread(&x,sizeof(double),1,f)) s+=x;
      }
      long sonlar_miqdori=ftell(f)/sizeof(double);
      s/=sonlar_miqdori; // s- o‘rta arifmetik
      cout<<"Fayldagi sonlar o'rta
arifmetiki="<<s<<endl;
      fseek(f,SEEK_SET,0); // fayl boshiga kelinsin
      int k=0;
      while (fread(&x,sizeof(x),1,f))
      {
       k+=(x<s); //o‘rta arifmetikdan kichik elementlar
soni
      }
      fclose(f);
      return k;
     }
     int main()
```

```
    {
     if(Fayl_Yaratish())
     {
      cout<<"Sonlar.dbl faylidagi\n";
      int OA_kichik=OAdan_Kichiklar_Soni();
      cout<<"O'rta arifmetikdan kichik sonlar
miqdori=";
      cout<<OA_kichik;
     }
      else  // f faylini yaratish muvaffaqiyatsiz
bo‘ldi.
      cout<<"Faylini ochish imkoni bo'lmadi!!!";
      return 0;
    }
```

Programmada bosh funksiyadan tashqari ikkita funksiya aniqlangan:

int Fayl_Yaratish() - diskda «Sonlar.dbl» nomli faylni yaratadi. Agar faylni yaratish muvaffaqiyatli bo‘lsa, funksiya 1 qiymatini, aks holda 0 qiymatini qaytaradi. Faylni yaratishda klaviaturadan sonlarning satr ko‘rinishi o‘qiladi va songa aylantirilib, faylga yoziladi. Agar bo‘sh satr kiritilsa, sonlarni kiritish jarayoni to‘xta-tiladi va fayl yopiladi;

int OAdan_Kichiklar_Soni() - diskdagi «Sonlar.dbl» nomli fayli o‘qish uchun ochiladi va fayl elementlarining s o‘rta arifmetigidan kichik elementlari soni k topiladi va funksiya natijasi sifatida qaytariladi.

Bosh funksiyada faylni yaratish muvaffaqiyatli kechganligi tekshiriladi va shunga mos xabar beriladi.

Nazorat savollari

1. Agar in_file.open("")ni belgilasangiz nima sodir bo;ladi?
2. Quyidagi kod bilan nima xatolik bor?

```
ifstream out_file;
out_file.open("output.txt");
out_file << "Hello, World!" << endl;
```

3. Quyidagi funksiya bilan nima xatolik bor? double sum(ifstream in)

```
{
 double total = 0;
 double input;
 while (in >> input) { total = total + input; }
 return total;
}
```

4. Qanday qilib babynames.cpp dasturnini o'zgartirasiz, shuningdek siz aholi orasida keng tarqalgan ismlarni bilib olasiz?

5. QAnday qilib babynames.cpp dasturini ozgartirasiz shuningdek dastur output 00 bo'lib saqlanadimi

7-Ma’ruza. Obektga yo`naltirilgan dasturlash tushunchasi

Ma’ruza rejasi:

7.1 Inkapsulyasiya, polimorfizm, vorislik;

7.2 Sinf ochiq intereysini aniqlash;

7.3 Obektga yo`naltirilgan dasturlash tushunchasi;

7.4 Konstruktorlar va destruktorlar.

Kalit so’zlar:, *ro’yxat, manzil, nolinchi ko’rchsatkich, tugun, adres olish &, bo’shatish, ko’rsatkich, virtual destruktor, xotira, xotira chiqishi, destruktor, toifani o’zlashtirish, Inkapsulyasiya, polimorfizm, vorislik, SINF a’zo destruktori.*

7.1 Inkapsulyasiya, polimorfizm, vorislik

Ob’yektga yo’naltirilgan dasturlash (OYD) – bu dasturlashga yangi bir yondashuvdir. Hisoblash texnikasining rivojlanishi va yechilayotgan masalalarni tobora murakkablashuvi dasturlashning turli modellarini (paradigmalarini) yuzaga kelishiga sabab bo’lmoqda. Birinchi kompilyatorlarda (masalan, FORTRAN tili) dasturlashning funksiyalardan foydalanishga asoslangan prosedura modelini qo’llab quvvatlagan. Bu model yordamida dastur tuzuvchi bir nechta ming satrli dasturlarni yozishi mumkin edi. Rivojlanishning keyingi bosqichida dasturlarning strukturali modeli paydo bo’ldi va ALGOL, Pascal va C tillari kompilyatorlarida o’z aksini topdi. Strukturali dasturlashning mohiyati – dasturni o’zaro bog’langan proseduralar (bloklar) va ular qayta ishlaydigan berilganlarning majmuasi deb qarashdan iborat. Ushbu model dastur bloklari keng qo’llashga, GOTO operatoridan imkon qadar kam foydalanishga tayangan va unda dastur tuzuvchi o’n ming satrdan ortiq dasturlarni yarata olgan. Yaratilgan dasturni prosedurali modelga nisbatan sozlash va nazorat qilish oson kechgan.

Murakkab masalalarni yechish uchun dasturlashning yangi uslubiga zarurat paydo bo’ldiki, u OYD modelida amalga oshirildi. OYD modeli bir nechta tayanch konsepsiyalarga asoslanadi.

Berilganlarni abstraksiyalash – berilganlarni yangi turini yaratish imkoniyati bo’lib, bu turlar bilan xuddi berilganlarning tayanch turlari bilan ishlagandek ishlash mumkin. Odatda yangi turlarni berilganlarning abstrakt turi deyiladi, garchi ularni soddaroq qilib “foydalanuvchi tomonidan aniqlangan tur” deb atash mumkin.

Inkapsulyasiya – bu berilganlar va ularni qayta ishlovchi kodni birlashtirish mexanizmidir. Inkapsulyasiya berilganlar va kodni tashqi ta’sirdan saqlash imkonini beradi.

Yuqoridagi ikkita konsepsiyani amalga oshirish uchun C++ tilida *sinflar* ishlatiladi. *Sinf* termini bilan Ob’yektlar turi aniqlanadi. Sinfning har bir vakili (nusxasi) *Ob’yekt* deb nomlanadi. Har bir Ob’yekt o’zining alohida holatiga ega bo’ladi. Ob’yekt holati uning *berilganlar-a’zolarning* ayni paytdagi qiymati bilan aniqlanadi. Sinf vazifasi uning *funksiya-a’zolarining* sinf ob’yektlari ustida bajaradigan amallar imkoniyati bilan aniqlanadi.

Berilgan sinf ob’yektini yaratish *konstruktor* deb nomlanuvchi maxsus funksiya-a’zo tomonidan, o’chirish esa destruktor deb nomlanuvchi maxsus funksiya–a’zo orqali amalga oshiriladi.

Sinf ichki berilganlarini murojaatni cheklab qo’yishi mumkin. Cheklov berilganlarni ochiq (public), yopiq (private) va himoyalangan (protected) deb aniqlash bilan tayinlanadi.

7.2 Sinf ochiq intereysini aniqlash

Sinf, shu turdagi ob’yektning tashqi dunyo bilan o’zaro bog’lanishi uchun qat’iy muloqot shartlarini aniqlaydi. Yopiq berilganlarga yoki kodga faqat shu ob’yekt ichida murojaat qilish mumkin. Boshqa tomondan, ochiq berilganlarga va kodlarga, garchi ular ob’yekt ichida aniqlangan bo’lsa ham, dasturning ixtiyoriy joyidan murojaat qilish mumkin va ular ob’yektni tashqi olam bilan muloqotni yaratishga xizmat qiladi. Yaratilgan ob’yektlarni, ularni funksiya–a’zolariga oddiygina murojaat orqali amalga oshiriluvchi *xabarlar* (yoki *so’rovlar*) yordamida

boshqarish mumkin. Keyinchalik Windows xabarlari bilan adashtirmaslik uchun so’rov termini ishlatiladi.

Vorislik – bu shunday jarayonki, unda bir ob’yekt boshqasining xossalarini o’zlashtirishi mumkin bo’ladi. Vorislik orqali mavjud sinflar asosida hosilaviy sinflarni qurish mumkin bo’ladi. Hosilaviy sinf (*sinf–avlod*) o’zining ona sinfidan (*sinf-ajdod*) berilganlar va funksiyalarni vorislik bo’yicha oladi, hamda ular satriga faqat o’ziga xos bo’lgan qirralarni amalga oshirishga imkon beruvchi berilgan va funksiyalarni qo’shadi. Ajdod sinfdagi himoyalangan berilgan-a’zolarga va funksiya-a’zolarga ajdod sinfda murojaat qilish mumkin bo’ladi. Bundan tashqari, hosilaviy sinfda ona sinf funksiyalari qayta aniqlanishi mumkin. Demak, vorislik asosida bir-biri bilan “ona-bola” munosabatidagi sinflar shajarasini yaratish mumkin. *Tayanch sinf* termini sinflar shajarasidagi ona sinf sinonimi sifatida ishlatiladi. Agar ob’yekt o’z atributlarini (berilganlar-a’zolar va funksiyalar–a’zolar) faqat bitta ona sinfdan vorislik bilan olsa, *yakka (*yoki *oddiy) vorislik* deyiladi. Agar ob’yekt o’z atributlarini bir nechta ona sinflardan olsa, *to’plamli vorislik* deyiladi.

Polimorfizm – bu kodning, bajarilish paytidan yuzaga keladigan holatga bog’liq ravishda o’zini turlicha amal qilish xususiyatidir. Polimorfizm – bu faqat ob’yektlar xususiyati bo’lmasdan, balki funksiyalar-a’zolar xususiyatidir va ular xususan, bitta nomdagi funksiya-a’zoni, har xil turdagi argumentlarga ega va bajaridagan amali unga uzatiladigan argumentlar turiga bog’liq bo’lgan funksiyalar uchun (o’rnida) foydalanish imkoniyatida namoyon bo’ladi. Bu holatga *funksiyalarni qayta yuklash* deyiladi. Polimorfizm amallarga ham qo’llanishi mumkin, ya’ni amal mazmuni (natijasi) operand (berilgan) turiga bog’liq bo’ladi. Polimorfizmning bunday turiga *amallarni qayta yuklash* deyiladi.

Polimorfizmning yana bir ta’rifi quyidagicha: polimorfizm – bu tayanch sinfga ko’rsatgichlarning (murojaatlarning), ularni virtual funksiyalarni chaqirishdagi turli shakl (qiymatlarni) qabul qilish imkoniyatidir. C++ tilining bunday imkoniyati *kechiktirilgan bog’lanish* natijasidir. Kechiktirilgan bog’lanishda chaqiriladigan funksiya-a’zolar adreslari dastur bajarilishi jarayonida

dinamik ravishda aniqlanadi. An’anaviy dasturlash tillarida esa bu adreslar statik bo’lib, ular kompilyasiya paytida aniqlanadi (*oldindan bog’lanish*). Kechiktirilgan bog’lanish faqat virtual funksiyalar uchun o’rinli.

C++ tili ob’yektga yo’naltirilgan dasturlash prinsiplarini qo’llab quvvatlaydi. Bu prinsiplar quyidagilardan iborat:

1. Inkapsulyasiya
2. Vorislik
3. Polimorfizm

Inkapsulyasiya. Agar muxandis ishlab chiqarishda diod, triod yoki rezistorni ishlatsa, u bu elementlarni yangitdan ihtiro qilmaydi, balki do’kondan sotib oladi [9]. Demak, muxandis ularning qanday tuzilganligiga e’tiborini qaratmaydi, bu elementlar yaxshi ishlasa yetarli. Aynan shu tashqi konstruksiyada ishlaydigan yashirinlik yoki avtonomlik xossasi inkapsulyasiya deyiladi.

Vorislik. Yangi ob’yekt yaratilayotgan bo’lsa, ikkita variantdan biri tanlanadi: mutlaqo yangisini yaratish yoki mavjud modelning konstruksiyasini takomillashtirishdir. Ko’pincha 2-variant tanlanadi, demak, ba’zi xususiyatlari o’zgartiriladi xolos. Bu narsa vorislik prinsipiga asos soladi. Yangi sinf oldin mavjud bo’lgan sinfni kengaytirishdan hosil bo’ladi. Bunda yangi sinf oldingi sinfning merosxo’ri deb ataladi.

Polimorfizm. Poli – ko’p, morfe – shakl degan ma’noni bildiradi. C++ tili bir xil nomdagi funksiya turli ob’yektlar tomonidan ishlatilganda turli amallarni bajarish imkoniyatini ta’minlaydi. Polimorfizm – shaklning ko’p xilligidir.

Dasturda ishlatiladigan har bir o’zgaruvchi o’z toifasiga ega va u quyidagilarni aniqlaydi:

1. Xotiradagi o’lchovini;
2. Unda saqlanayotgan ma’lumotlarni;
3. Uning yordamida bajarilishi mumkin bulgan amallarni.

C++ tilida dasturchi o’ziga kerakli ixtiyoriy toifani hosil qilishi mumkin. Bu yangi toifa ichki toifalarning xossalari va ularning funksional imkoniyatlarini o’zida

ifodalaydi. Yangi toifa sinfni e’lon qilish orqali tuziladi. Sinf bu – bir-biri bilan funksional bog’angan o’zgaruvchilar va usullar (funksiyalar) to’plamidir.

Masalan: Mushuk nomli sinf tuzmoqchimiz. Bu yerda uning yoshi, og’irligi kabi o’zgaruvchilar va miyovlash, sichqon tutish kabi funksiyalardan ishdatiladi. Yoki Mashina sinfi g’ildirak, eshik, o’rindiq, oyna kabi o’zgaruvchilar va xaydash, to’xtatish kabi funksiyalardan iborat.

Sinfdagi o’zgaruvchilar – sinf a’zolari yoki sinf xossalari deyiladi.

Sinfdagi funksiyalar odatda o’zgaruvchilar ustida biror bir amal bajaradi. Ularni sinf usullari (metodlari) deb ham ataladi.

Sinfni e’lon qilish uchun class so’zi , { } ichida esa shu sinfning a’zolari va usullari keltiriladi. Masalan:

```
class non
{ int ogirlik ;
  int baho ;
  void yasash ( );
  void yopish ( );
  void eyish ( );
}
```

Sinfni e’lon qilishda xotira ajratilmaydi. Sinf e’lon qilinganda kompilyator faqat shunday (non) sinf borligini, hamda unda qanday qiymatlar (ogirlik, baho) saqlanishi mumkinligini, ular yordamida qanday amallarni (yasash, yopish, yeyish) bajarish mumkinligi haqida xabar beradi. Bu sinf ob’yekti hammasi bo’lib 4 bayt joy egallaydi (2 ta int).

7.3 Obektga yo`naltirilgan dasturlash tushunchasi

Ob’yekt sinfning biror bir nusxasi hisoblanadi.

C++ tilida toifalarga qiymat o’zlashtirilmaydi, balki o’zgaruvchiga o’zlashtiriladi. Shuning uchun to’g’ridan-to’g’ri int = 55 deb yozib bo’lmaganidek non.baho=1200 deb ham bo’lmaydi. O’zlashtirishda xatolikka yo’l qo’ymaslik

uchun oldin non sinfiga tegishli patir ob’yektini hosil qilamiz keyin esa unga kerakli qiymatlarni beramiz.

Masalan:

```
int a;     // butun toifali a o’zgaruvchisi, ob’yekti
non patir; //
```

Endi non sinfining real ob’yekti aniqlanganidan so’ng uning a’zolariga murojaat

```
patir.baho = 1200;
patir.ogirlik = 500;
patir.yasash ( ) ;
```

Sinfni e’lon qilishda quyidagilardan foydalaniladi:

public - ochiq

private – yopiq

Konstruktorlar va destruktorlar

Sinfning barcha usul va a’zolari boshlang’ich holda avtomatik ravishda yopiq bo’ladi. Yopiq a’zolarga esa faqat shu sinfning usullari orqaligina murojaat qilish mumkin. Ob’yektning ochiq a’zolariga esa dasturdagi barcha funksiyalar murojaat qilishi mumkin. Lekin sinf a’zolariga murojaat qilish ancha mushkul ish hisoblanadi. Agar to’g’ridan to’g’ri:

```
non patir;
patir.baho = 1200;
patir.og`irlik = 500;  deb yozsak xato bo’ladi.
```

A’zolarga murojaat qilishdan oldin uni ochiq deb e’lon qilish kerak:

```
# include < iostream.h >
class non
{ public :
  int baho;
  int ogirlik;
```

```
  void yasash ( );  };
void main ( )
{ non patir;
patir.baho = 1200;  patir.ogirlik = 500;
cout <<“men olgan patir” <<patir.baho <<“so’m”<<endl;
cout <<“uning og’irligi =”<<patir.og’irlik <<endl ;   }
```

Nazorat savollari

1. Inkapsulyasiya nima?
2. Polimorfizm haqida tushuncha.
3. Vorislikning qo’llanishi.
4. Sinfdagi o’zgaruvchilar – sinf a’zolarining qo’llanishi.

8-Ma’ruza. Obektga yo`naltirilgan dasturlash tamoyillari

Ma’ruza rejasi:

8.1 Muammo echimi: ob’ektlar trassirovkasi;

8.2 Nusxalash konstruktori;

8.3 This ko’rsatgichi;

8.4 Joylashtiriladigan (inline) funksiyalar–a’zolar.

Kalit so’zlar:, *ro’yxat, manzil, nolinchi ko’rchsatkich, tugun, adres olish &, bo’shatish, ko’rsatkich, virtual destruktor, xotira, xotira chiqishi, destruktor, toifani o’zlashtirish, Inkapsulyasiya, inline, polimorfizm, vorislik, SINF a’zo destruktori.*

8.1 Muaamo echimi: Ob’ektlar trassirovkasi

Foydalanuvchi ma’lumotlarni kiritmaguncha menyu kutib turadi. Agarda foydalanuvchi to‘g‘ri qiymatni kiritmasa, menyu yangilanadi, foydalanuvchi ma’lumotlarni boshqatdan kiritishi mumkin.

1 Qadam O‘z ob’ektlaringizning majburiyatini belgilovchi ro‘yxat tuzing.

Faqat sizning topshirig‘ingizni echish uchun zarur bo‘lgan funksiyalarni amalga oshiring. Real narsalar, masalan kassa aparati yoki bank hisob-raqami funksiyasini amalga oshirish uchun o‘n ikkilik funksiyasidan foydalaniladi. Biroq, sizning vazifaningiz real dunyoning modelini yaratishdan iborat emas. Sizning topshirig‘ingizni echish uchun zarur bo‘ladigan vazifalarni aniqlashtirib olishingiz lozim.

Display the menu.(Menyu kiritish)

Get user input. (Foydalanuvchidan kirish ma’lumotlarini olish)

Tavsif muammosi qismiga kirmaydigan yashirin majburiyatlarga qarang.Ob’ekt qanday yaratiladi? Qanday oddiy faoliyatlar ro‘y berishi kerak, har bir savdoni boshlanishida kassa aparatini tozalashga o‘xshash? Menyuni tuzishni menyu yaratish misolida ko‘rib chiqing. Programmist menyuning bo‘sh ob’ektini

yaratadi va undan so‘ng “YAngi akkaunt ochish”, “YOrdam ” opsiyasini qo‘shadi. Bu erda yashirin majburiyat bor:

kabi:

```
Menu main_menu;
main_menu.add_option("Open new account");
// Add more options
int input = main_menu.get_input();
```

Endi biz o‘ziga xos metodlar ro‘yxatiga egamiz • void add_option(string option)

• int get_input() const

Menyu chiqarish masalasichi?Menyuni foydalanuvchidan ma’lumot kirtishni so‘ramasdan

ko‘rsatishning ma’nosi yo‘q.Agar foydalanuvchi xato ma’lumot kiritsa *get_input* menyuni bir martadan ortiq kiritadi. SHunday qilib *display* xususiy metod uchun yaxshi kandidatdir.

Ijtimoiy interfeysni yakunlash uchun siz konstruktorlarni aniqlashingiz kerak. Ozingizdan so‘rangob’ekt yaratish uchun sizga nima kerak. Bazansiz 2 ta konstruktorga extiyoj sezasiz: biri xamma elementlar uchun, ikkinchisi esa foydalanuvchi tomonidankiritilgan

qiymatlarni o‘rnatadi.

Menyu misolida biz bo‘sh menyu yaratuvchi yagona konstruktor bilan kifoyalanamiz.

Bu erda ijtimoiy interfeys

8.2. Nusxalash konstruktori

Ba’zida, bir xil uzunlikdagi vektorlarni ishlatyotganingizni anglaysiz, har bir saqlaydigan qismi ob’ekt hisoblanadi. Bu vaziyatda, dasturingizni qayta yaratish va elementlari ob’ekt sanaladigan yagona vektordan foydalanish yaxshi fikrdir.

Masalan, faraz qilaylik hisob raqami bir qator tavsif va narxlardan iboratdir.YAgona echim ikkita vektorni saqlab turishdir:

vector<string> descriptions;

vector<double> prices;

Vektorlarning har biri bir xil uzunlik va bo‘lakka ega bo‘lib, consisting of descrip-tions[i] tavsifi va and [i] narxlaridan iborat, birga ishlanadigan ma’lumotlar =am kiradi. Bu vektorlar paralel vektrlar deb aytiladi. (5- shaklga qarang).

Parallel vektorlar katta dasturlarda muammoni keltirib chiqaradi. Dasturchi ishonch hosil qilishi kerak, vektorlar doim o‘sha uzunlikka ega, qaysiki bir-biriga tegishli kesimlar qiymatlar bilan to‘ldirilgan bo‘ladi. Bundan tashqari kesimda ishlaydigan ixtiyoriy funksiya barcha vektorlarni argument sifatida qabul qilishi kerak, ya’ni dastur uchun zerikarli bo‘lganlarini.

Buning yo‘li oson. Kesimga qarang va vakil bo‘layotgan tushunchani toping, so‘ng bu tushunchani klassni ichiga kiriting. Bu kesimda har bir sinfning tavsifi va narx bandidan iborat; buni sinfga kiritingga kiriting.

8.3. This ko’rsatgichi

```
class Item
{
public:
  ...
private:
 string description;
 double price;
};
```

Fayldagi manba’

- Komponentlik funksiyalari tavsifi
- Komponentsiz funksiyalar tavsifi.

CashRegister cinfi uchun, siz bir juft faylni yaratasiz: cashregister.h va cashregister.cpp, unda interfeys va qo‘llanilish mavjud.

8.4. Joylashtiriladigan (inline) funksiyalar–a’zolar

Sarlaxa fayli sinf tavsiffini o‘z ichiga oladi:

ch09/cashregister.h

```
1 #ifndef CASHREGISTER_H
2 #define CASHREGISTER_H
3
4 /**
5  A simulated cash register that tracks the item count and
6 the total amount due.
7 */
8 class CashRegister
9 {
10 public:
11 /**
12 Constructs a cash register with cleared item count and total.
13 */
14 CashRegister();
15
16 /**
17 Clears the item count and the total.
18 */
19 void clear();
20
```

```
21 /**
22 Adds an item to this cash register.
23 @param price the price of this item
24   */
25 void add_item(double price);
26
27 /**
28 @return the total amount of the current sale
29 */
30 double get_total() const;
31
32 /**
33 @return the item count of the current sale
34 */
35 int get_count() const;
36
37 private:
38 int item_count;
39 double total_price;
40 };
41
42 #endif
```

Bu sarlavhani siz harqachon CashRegister sinf tavsifi talab etilganda qo‘shasiz. CHunki bu fayl standart sarlavxa fayli emas, siz uning nomi va koefitsentini qovus ichiga olishingiz kerak, <...> bunday emas, bu tarzda:

```
#include "cashregister.h"
```

Nazorat savollari

1. Tokrorlanuvchi operatorlar qanday ishlaydi?

2. Takrorlanuvchi operatorlariga kutubxona kerakmi?
3. String turi qaysi kutubxonada joylashgan?
4. String turi bilovchi qanday funksiyalarni bilasiz?
5. If operatorining nechta turi bor?
6. O’zgaruvchilar nima uchun qo’llaniladi?

9-Ma’ruza. Sinf. Sinf xususiyati va metodi

Ma’ruza rejasi:

9.1 Sinfning statik a’zolari.

9.2 Sinfning konstansta ob’yektlari va konstanta funksiya-a’zolari.

9.3 Sinf usullarining aniqlanishi.

Kalit so’zlar:*, ro’yxat, manzil, nolinchi ko’rchsatkich, tugun, adres olish &, bo’shatish, ko’rsatkich, virtual destruktor, xotira, xotira chiqishi, destruktor, toifani o’zlashtirish, Inkapsulyasiya, inline, polimorfizm, vorislik, SINF a’zo destruktori.*

9.1 Sinfning statik a’zolari

Sinf a’zolari static modifikatori bilan e’lon qilinishi mumkin. Sinf statik a’zosini sinf sohasi chegarasida murojaat qilish mumkin bo’lgan global o’zgaruvchi yoki funksiya deb qarash mumkin. Sinfning static deb e’lon qilingan berilganlar-a’zolari sinfning barcha ob’yektlari tomonidan birgalikda ishlatiladi, chunki bunday o’zgaruvchining yagona nusxasi bo’ladi. Amalda sinfning statik berilganlari uchun xotiradan joy ajratiladi, hattoki sinfning birorta ob’yekti bo’lmasa ham. Shu sababli sinf statik berilganini e’lon qilib qolmasdan, uni aniqlash shart. Masalan:

```
slass Sinf
{
public:
 Sinf();
static int Sanagich;//statik berilgan–a’zo e’loni
  }
        int Sinf::Sanagich=0; // statik berilgan–a’zo e’loni
```

Bu misolda, garchi Sanagich statik berilgan – a’zo public bo’limida e’lon qilingan sinf ob’yekti nomini ishlatish yordamida murojaat qilish mumkin.

```
Sinf sinf1;
Sinf1.Sanagich++;
```

Sinf sinf2;

Sinf2->Sanagich--;

Statik berilganlar – a’zolarga sinf nomi orqali murojaat qilgan ma’qul bo’ladi.

Sinf::Sanagich++;

Bu holat Sanagich statik berilgan–a’zo barcha sinf ob’yektlari uchun yagona ekanligini ta’kidlaydi.

Agarda statik berilganlar yopiq deb e’lon qilingan bo’lsa, ularga funksiyalar – a’zolar orqali murojaat qilish mumkin.

Umuman olganda statik berilganlar–a’zolarni ishlatishda quyidagi takliflarni berish mumkin:

statik berilganlar-a’zolarni bir nechta sinf ob’yektlari tomonidan birgalikda ishlatish uchun aniqlang;

statik berilganlar-a’zolarini private, protected modifikatorlar bilan e’lon qilish orqali ularga murojaatni cheklang.

Sinfning statik berilgan-a’zosini ishlatishga misol.

```
slass S;
      {
public:
S() {ob_soni++;}
  ~S() {ob_soni--;}
static int ob_soni ;
private:
int x;
 };
int S::ob_soni=0;
int main ()
 {
 S* p_ob=new S[5];
 cout<<”Sinfning ”<<S::ob_soni<<”Ob’yekti mavjud./n”;
```

```
delete [] p_ob;
return 0;
 }
```

Dastur ishlash natijasida ekranga

Sinfning 5 ob’yekti mavjud.

Sinfning statik funksiyalarni ishlatishning o’ziga xosligi shundaki, ular ham yagona nusxada aniqlanadi va birorta sinf ob’yektining “shaxsiy” funksiyasi bo’lmaydi. Shu sababli, bu funksiyalarga this ko’rsatgichi uzatilmaydi. Statik funksiyalarning bunday xususiyatidan Windows uchun dasturlashda keng foydalaniladi.

Yuqorida aytilgan fikrlardan bir nechta muhim xulosalar kelib chiqadi:

statik funksiya –a’zolarni sinfning birorta ham vakili (ob’yekti) mavjud bo’lmasa ham chaqirish mumkin;

sinfning statik funksiyasi faqat sinfning statik berilganlarini qayta ishlashi mumkin va faqat sinfning statik funksiya–a’zolarini chaqirishi mumkin;

statik funksiya-a’zo virtual modifikatori bilan e’lon qilinishi mumkin emas.

Quyida keltirilgan dastur funksiya–a’zoni ishlatishga misol bo’ladi:

```
#include <iostream.h>
class S
 {
public:
S() {sanagich++};
      ~S() {sanagich--};
      //..
    static int Sinf_Sanagichi(){return sanagich;}
private:
int x;
static int sanagich;
 };
```

```
int S::sanagich=0;

int main()
 {
 S * pOb= new S[10];
cout<<"S sinfning "<<S::Sinf_Sanagichi()
<<" Ob'yekti mavjud"<<endl;
delete [] pOb;
return 0;
 }
```

9.2 Sinfning konstansta ob'yektlari va konstanta funksiya-a'zolari

Sinfning funksiya-a'zolari parametrlar ro'yxatidan keyin keluvchi const modifikatori bilan e'lon qilinishi mumkin. Bunday funksiya sinf berilganlar-a'zolari qiymatlarini o'zgartira olmaydi va sinfning konstanta bo'lmagan funksiya–a'zolarini chaqirishi mumkin emas. Konstanta funksiya-a'zosi bo'lgan sinfga misol keltiramiz.

```
classNuqta
 {
int x,y;
    public:
Nuqta(int _x, int _y);
void Qiymat_Berish(int x0, int y0);
//konstanta funksiya-a'zo
void Qiymat_Olish(int xx, int yy) const;
};
Nuqta::Nuqta(int _x, int _y)
{ x=_x; y=_y;}
voidNuqta::Qiymat_Berish(int x0, int y0)
```

```
{ x=x0; y=y0;}
voidNuqta::Qiymat_Olish(int &xx, int &yy)const
{ xx=x; yy=y; }
```

Xuddi shunday, konstanta ob’yektlar yaratish mumkin. Buning uchun ob’yekt e’loni oldiga const modifikatori qo’yilishi kerak.

const Nuqta koord(3,6);

const kalit so’zi kompilyatorga ushbu ob’yektning holati o’zgarmasligi kerakligini bildiradi. Shu sababli ob’yekt berilgan-az’olari qiymatini o’zgartiradigan funksiya–a’zosini chaqirishi uchrasa, kompilyator xato haqida xabar beradi. Bu qoidaga konstanta funksiya-a’zolarni chaqirish rioya qilmaydi, chunki o’z mazmuniga ko’ra ular berilgan–a’zolarni qiymatini o’zgartira olmaydi. Yuqorida e’lon qilingan Nuqta sinfini konstanta ob’yekt ishlatishiga misol.

```
// Nuqta sinf e’loni va aniqlanishi
int main()
 {
Nuqtanuqta1(3,7);
constNuqta nuqta2(8,10);//Konstanta ob’yekt
int a,b;
 nuqta1.Qiymat_Olish (a,b);
 nuqta2. Qiymat_Berish(2,3);// Xato
nuqta2. Qiymat_Olish(a,b); // To’g’ri
 return 0;
 }
```

Konstanta funksiya–a’zolarning berilganlar–a’zolar bilan ishlashda bog’liq cheklovlarni “aylanib o’tish” uchun mutable kalit so’zi aniqlangan. Bu kalit so’z sinfning qaysi berilgan-a’zosi konstanta funksiya-a’zolar tomonidan o’zgartirilishi

mumkin ekanligini ko’rsatadi. Statik va konstanta berilganlar-a’zolariga mutable kalit so’zini ishlatish mumkin emas, u berilganlar turining modifikatori sifatida ishlatiladi.

Misol.

```
#include <iostream.h>
class Sinf
{
mutable int count;
mutable const int * intPtr;//O’rinli, garchi
// ko’rsatgich konstanta butun songa ko’rsatsa
// ham o’zi konstanta emas
public:
int Funktsiya(int i=0) const
 {
count=i++;
intPtr =&i;
cout<<*intPtr;
return count;
 }
};
int main()
{
 Sinf S;
S.Funktsiya();
return 0;
}
```

Dastur ishlashi natijasida ekranga

1

chop etiladi.

Shu vaqtgacha tuzgan dasturlarimiz berilgan ma’lumotlar ustida biror-bir amallarni bajaruvchi proseduralar ketma-ketligidan iborat edi. Prosedura va funksiyalar ham o’zida aniqlangan ketma-ket komandalar to’plamidan iboratdir.

Ob’yektga yo’naltirilgan dasturlashning asosiy maqsadi berilganlar va ular ustida amal bajaruvchi proseduralarni yagona ob’yekt deb qarashdan iboratdir. Masalan: non, avtomobil, odam … ob’yektlari.

9.3 Sinf usullarining aniqlanishi

Sinf usulini aniqlash uchun sinf nomidan so’ng ikkita ikki nuqta (::) belgisi, funksiya nomi va uning parametrlari ko’rsatiladi.

Masalan: non sinfining patir ob’yektidagi usullarni aniqlash dasturi:

```
# include < iostream.h >
class non
{ public :
  int baho;
  int og`irlik;
  void yasash ( );  };
void non :: yasash ( );
{ cout << “hamir qoriladi, zuvala tutiladi, doira holiga keltiriladi”<< endl; }
void main ( )
{ non patir;
patir.baho = 1200;  patir.og`irlik = 500;
cout <<”men olgan patir “<<patir.baho <<” so’m”<<endl;
cout <<” uning og’irligi =”<<patir.og`irlik <<endl ;
cout <<” u quyidagich yasaladi:”;
```

```
patir . yasash ( );  }
```

Misollar:

```
#include <vcl.h>
#pragma hdrstop
#include<iostream.h>
#include<conio.h>
#include<string.h>
#pragma argsused
int i;
struct waw
{
  char fam[15];
  char ism[15];
  int tel;
  int date[3];
};
class note
{
  waw A[3],t;
  public:
  void kiritish();
  void saralash();
  void taqqoslash();
  void chiqarish();
};
void note::kiritish()
{
  for( i=0; i<3; i++)
  {
```

```
        cout<<i+1<<«. «;
        cin>>A[i].fam;
        cin>>A[i].ism;
        cin>>A[i].tel;
        cin>>A[i].date[0]>>A[i].date[1]>>A[i].date[2];
      }
    }
    // saralash tel. raqamining 1-chi 3 ta raqami
bo’yicha
    void note::saralash()
    {
      int x,y;
      for( i = 0; i < 2; i++)
      {
        x = A[i].tel / 10000;
        for(int j = i; j < 3; j++)
        {
          y = A[j].tel / 10000;
          if(x > y)
          {
            t = A[i];
            A[i] =A[j];
            A[j] = t;
          }
        }
      }
    }
    void note::taqqoslash()
    {  char f[15];
      int w=0;
```

```
        cout<<"\n\nfamilyani kiriting = ";
        cin>>f;
        for(i=0; i<3; i++)
        if( strcmp(A[i].fam, f)==0)
          {
         cout<<«\n\nfamiliya ismi   ---
>"<<A[i].fam<<"\t"<<A[i].ism<<endl;
          cout<<«telifon nomeri   --->
"<<A[i].tel<<endl;
          cout<<«tug'ilgan sanasi  --->
"<<A[i].date[0]<<" "<<A[i].date[1]<<"
"<<A[i].date[2]<<endl;
          w++;  break;
          }
        if(w==0)
        cout<<"bunday familiya haqida ma'lumot yo'q";
      }
      void note::chiqarish()
      {
        for(i = 0; i < 3; i++)
        {
          cout<<"\n\nfamiliya ismi ---
>"<<A[i].fam<<"\t"<<A[i].ism<<endl;
          cout<<"telifon nomeri   --->
"<<A[i].tel<<endl;
          cout<<"tug'ilgan sanasi  --->
"<<A[i].date[0]<<" "<<A[i].date[1]<<"
"<<A[i].date[2]<<endl;
        }
```

```
}
void main()
{
  note B;          //ob'yekt tavsifi
  cout<<"ma'lumotlarni kiriting: (familiya, ism,
tel.nomer, tug'ilgan sana):\n";
  B.kiritish();
  cout<<"\n tartiblangan holatda : \n\n";
  B.saralash();
  B.chiqarish();
  cout<<"\n\n kiritilgan familiya haqida malumot
chiqarish:\n\n";
  B.taqqoslash();
getch();
}
```

y = Ax+B chiziqli tenglama. A koeffisient sifatida first maydoni – kasr son; B koeffisient sifatida kasr son bo'lgan second maydoni olinmoqda. Chiziqli tenglama ildizini aniqlovchi root () usuli tatbiq qilinsin. Usul B koeffisientning nolga teng emasligini tekshirmog'i lozim.

```
#include <vcl.h>
#pragma hdrstop
#include <iostream.h>
#include <conio.h>
#include <stdio.h>
#include <string.h>
#pragma argsused
struct kasr
{
 float surat;
```

```
 unsigned int maxraj;
};
class tenglama
{
 public:
    kasr A, B;
    float y, x;
    void vvod (void);
    void root(void);
};
 void tenglama::vvod(void)
 {
  cout << «\n x = « ;
  cin >> x;
  cout << «\nA kasr surati = « ;
  cin >> A.surat;
  cout << «\nA kasr maxraji = « ;
  cin >> A.maxraj;
  cout << «\nB kasr surati = « ;
  cin >> B.surat;
  cout << «\nB kasr maxraji = « ;
  cin >> B.maxraj;
 }
 void tenglama::root(void)
 {
  y = x * (float)(A.surat / A.maxraj) + (float)(B.surat / B.maxraj);
 }
int main(int argc, char* argv[])
{
```

```
 tenglama C;
 C.vvod();
 C.root();
 cout << “\n\nnatija :\n y =”<< C.y ;
 getch();
 return 0;
}
```

Qiymatlari gradus va minutlarda berilgan tekislikdagi burchaklar bilan ishlovchi Angle sinfi tuzilsin.

Bunda radianlarga o’tkazish, 0-360 diapazonga keltirish, burchakni berilgan qiymatga ko’paytirish yoki kamaytirish, sinusni tashkil etish, burchaklarni taqqoslashni tatbiq etilsin.

```
#include <vcl.h>
#pragma hdrstop
#include <iostream.h>
#include <conio.h>
#include <stdio.h>
#include <string.h>
#include <math.h>
#pragma argsused
class angle
{
    float gradus;
    float minut;
    float radian;
    float sinus;
  public :
```

```
        void read(void);//float *gradus, float *minut,
float *sinus);
        void converter(void);
        void write(void);
    };
    void angle :: converter(void)
    {
      radian = (gradus + minut / 60) / 180 * 3.14;
    }
    void angle :: read(void)//float *gradus_value,
float *minut_value, float *sinus_value)
    {
      cout << «\ngradus = « ;
      cin >> gradus ;
      if(gradus > 360)
      {
        int qoldiq = gradus;
        qoldiq = qoldiq % 360;
        gradus = qoldiq;
      }
      cout <<”minut =”;
      cin >> minut ;
      radian = (gradus + minut / 60) / 180 * 3.14;
      sinus = sin(radian * 180 / 3.14);
    }
    void angle :: write(void)
    {
      cout <<”\ngradus = ” << gradus;
      cout <<”\nminut = ” << minut;
      cout << ”\nradian = ” << radian;
```

```
  cout <<"\nburchak sinusi = " << sinus;
}
int main(int argc, char* argv[])
{
  angle info[10];
  int n, i, min, max;
  cout << "elementlar soni =" ;
  cin >> n ;
  for(i = 0; i < n; i++)
  {
    cout << i + 1 <<"- inchi ma'lumotni kiriting" ;
    info[i].read();
  }
  cout << "\nnatija : \n" ;
  for(i = 0; i < n; i++)
  {
    cout <<"\n\n"<< i <<" - inchi ma'lumotni : " ;
    info[i].write();
  }
  getch();
  return 0;
}
```

Student sinfi tashkil etilib, guruh talabalari orasidan 4 va 5 baho olgan talabalarni familiyasi, guruhi chiqadigan dastur tuzilsin.

```
#include <conio.h>
#include <stdio.h>
#include<math.h>
#include<iostream.h>
#include<string.h>
#include <stdlib.h>
```

```
class student
{ public :
     char fam[20];
  int baho;
  int guruh;
  int bilimdon (int );  };
int student :: bilimdon (int x)
{ if(x>=4) return x;
else return 0; }
int main ( )
{student alo[10]; int i,a=0;
for (i=0;i<5;i++){
  cout<<i+1<<”-talaba:”<<endl;
  cout<<”familiya:”;
  cin>>alo[i]. fam;
  cout<< ”guruhi: ”;
  cin>>alo[i].guruh;
  cout<< ”bahosi: ”;
  cin>>alo[i].baho;} cout<<”4 va 5 olgan talabalar:
”<<endl;
  for(i=0;i<5;i++){
int y=alo[i] . bilimdon (alo[i].baho );
if(y!=0){a++;
cout<< familiya: ”<<alo[i].fam<<'\t'<<”guruhi: ”<<alo[i
].guruh<<endl;}}
if(a==0) cout<<”talabalarning hammasi ikkichi”;
 getch();
return 0;
}
```

Uchburchakni aniqlashtiruvchi Triangle sinfi tashkil etilsin. Ma’lumotlar maydoni tarkibiga burchaklar va tomomnlari kiritilsin. Berilganlar maydonlarini olish va o’zgartirish, yuzasini hisoblash, perimetrni hisoblash, balandliklarni hisoblash hamda uchburchak turini aniqlash amallarini joriy etish talab etiladi.

```
#include <vcl.h>
#pragma hdrstop
#include <iostream.h>
#include <conio.h>
#include <math.h>
#pragma argsused
  class triangle
  {
    float x[3];
    float y[3];
    float tomon[3];
    float h[3];
    float s;
    float p;
    char stil_treugolnik[101];
  public :
    void vvod(void);
    void schitat(void);
  };
  void triangle :: vvod(void)
  {
    cout << "\nA uchni \nX =" ;
    cin >> x[0];
    cout << "Y =" ;
    cin >> y[0];
```

```
        cout << "\nB uchni \nX = " ;
        cin >> x[1];
        cout << "Y = " ;
        cin >> y[1];
        cout << "\nC uchni \nX = " ;
        cin >> x[2];
        cout << "Y = " ;
        cin >> y[2];
      }
      void triangle :: schitat(void)
      {
       tomon[0] = pow( (pow((x[0]-x[1]), 2) +
pow((y[0]-y[1]), 2) ), 1/2);
       tomon[1] = pow( (pow((x[1]-x[2]), 2) +
pow((y[1]-y[2]), 2) ), 1/2);
       tomon[2] = pow( (pow((x[2]-x[0]), 2) +
pow((y[2]-y[0]), 2) ), 1/2);
        cout <<"\n ucburchak tomonlari :
"<<tomon[0]<<", "<<tomon[1]<<", "<<tomon[2];
        p = tomon[0] + tomon[1] + tomon[2];
        cout <<"\nperimetr ="<<p;
        float p2 = p;
        s = sqrt(p2 * (p2-tomon[0])*(p2-tomon[1])*(p2-
tomon[2]) );
        cout << "\nyuza ="<< s;
        h[0] = s / tomon[0];
        h[1] = s / tomon[1];
        h[2] = s / tomon[2];
        cout <<"\n balandliklar :"<<h[0]
<<","<<h[1]<<","<<h[2];
```

```
        if(tomon[0] == tomon[1] && tomon[1] ==
tomon[2])
        strcpy(stil_treugolnik, «teng tomonli»);
        else if(tomon[0] == tomon[1] || tomon[1] ==
tomon[2] || tomon[0] == tomon[2])
        strcpy(stil_treugolnik, “teng yonli”);
        else strcpy(stil_treugolnik, “turli tomonli”);
        cout <<”\n uchburchak turi :”<<stil_treugolnik;
      }
    int main(int argc, char* argv[])
    {
      triangle info;
      info.vvod();
      info.schitat();
      getch();
      return 0;
    }
```

Nazorat savollari

Grade sinfi element lari. "B+" kabi qatordan gradus yasaydigan konstruktorni ta’minlang. Graduslarni ularning raqamli qiymatlariga o‘giradigan funksiya ham sizga kerak bo‘ladi. (masalan, "B+" dan 3.3 kelib chiqadi).

P9.13 Davlat nomi, uning aholisi va maydonini saqlaydigan Davlat sinfini aniqlang. Ushbu sinfdan foydalanib, davlatlar guruhini o‘qiydigan va chop etadigan dastur yozing.

• Eng katta maydonli davlat.

• Aholisi eng ko‘p davlat.

• Aholi zichligi eng katta davlat (har bir kilometr yoki mil maydonga aholi zichligi).

P9.14 e-mail xabarlarini tuzadigan Xabar sinfini yarating. Xabarning qabul qiluvchisi, yuboruvchisi va matni bo‘ladi. Quyidagi kompleks funksiyalarni qo‘llab quvvatlang:

• yuboruvchi va qabul qiluvchini oladigan, vaqt shtampini joriy vaqtga o‘rnatadigan konstruktor

• Matn qatorini xabarga qo‘shadigan yoz uvlar qo‘shish kompleks fu nksiyas i

• Xabarni bitta uzun qatorga tuzuvchi to_string kompleks funksiyasi, quyidagi kabi: "From: Harry Hacker\nTo: Rudolf Reindeer\n ..."

• Xabar matni chop etadigan print kompleks fu nksiyas i. Maslahat: to_string dan foydalanib ushbu sinfdan foydalanib xabar t uz adig an va chop et adig an dast ur yozing .

10-Ma’ruza. Modifikatorlar

Ma’ruza rejasi:

10.1 Vorislikda murojaat huquqlari;

10.2 Vorislikda destruktorlar hossalari.

Kalit so’zlar:*, ro’yxat, manzil, nolinchi ko’rchsatkich, tugun, adres olish &, bo’shatish, ko’rsatkich, virtual destruktor, xotira, xotira chiqishi, destruktor, toifani o’zlashtirish, Inkapsulyasiya, inline, polimorfizm, vorislik, SINF a’zo destruktori.*

Vorislikda asosiy sinfning ba'zi komponenta ma'lumotlari yoki komponenta funktsiyalari hosilaviy sinfda yangidan ta'riflanishi mumkin. Bu holda asosiy sinfning komponentalariga hosilaviy sinfdan to’g’ridan-to’g’ri murojaat qilib bo’lmaydi. Bu holda ko’rinish doirasini aniqlovchi ^:: amalidan foydalanish lozimdir. Har qanday komplament sinf o’z urnida boshqa sinflar uchun hosilaviy bo’ladi. Sinflar va ob'ektlar ierarhiyasida hosilaviy ob'ekt hamma asosiy sinflarning ruhsat berilgan komponentalarini hisobga oladi. Sinflar volisligida komponentalarning murojaat huquqlari katta rol' o’ynaydi. Haqiqiy sinf uchun komponentalari ta'sir doirasida yotadi. Shuning uchun sinfga tegishli har qanday funktsiya ihtiyoriy komponenta ma'lumotlariga murojaat qilishi va sinfga tegishli ihtiyoriy funktsiyani chaqirishi mumkin. Sinf tashqarisida faqat public huquqiga ega komponentalarga murojaat qilish mumkin. Sinflar perergiyasida sinf komponentalariga murojaat huquqlari qo’yidagilar:

Hususiy (private) metodlar va ma'lumotlarga faqat sinf ichida murojaat qilish mumkin.

Himoyalangan (protected) komponentalari o’z sinflari va shu sinfga me'roshur

bo’lgan hamma hosilaviy sinflarga murojaat qilish mumkin.

Umumiy (public) komponentlar global ya'ni dastur ihtiyoriy nuqtasidan murojaat qilish mumkin.

Himoyalanganlari murojaat sinf clarl, istruct yoki union so’zlarining qaysi biri bilan ta'riflanganligiga ham bog’likdir.

A " ekrandagi nuqta" asosiy sinf hisoblansa uning asosida " ekrandagi darcha sinfni ugirish mumkin. Bu sinf berilganlarni ikki nuqta :

& chap yuqori burchakli aniqlovchi nuqta

& darcha o’lchovlarini ya'ni chap yuqori burchakka nisbatan koordinatalar o’qi

buyicha siljish.

Ekrandagi darcha sinfi me'todlari:

& darchani H o’qi bo’yicha DX ga surish

& darchani U o’qi bo’yicha DY ga surish

& chap yuqori burchak H koordinatasini aniqlash

& chap yuqori burchak U koordinatasini aniqlash;

& H o’qi buyicha darcha uzunligini aniqlash

& Y uki buyicha darcha uzunligini aniklash

Darchaning ekrandagi konstrutuktori:

& ekranda chap yuqori burchagi va ulchamlari asosida berilgan nomli darcha yaratish;

Ekrandagi destruktori

& berilgan nomli darchani yo’q qilish:

Nasldan o’tuvchi komponentalarga qo’shimcha jpot sinfiga qo’yidagi komponentalarni kiritamiz: tasvir radiusi (rad); ekralus sos etilishi (vir=0 ekranda tasvir yuk; vi1==1 ekranda tasvir bor); tasvirni bitli matnda saylash chun ajratilgan hotira qismiga ko’rsatgich pspot.

```
Spot.cpp
# Ifudef. Spot
# Spot1
```

```
"Include " " point. Epp"
Clall spot;
{Int rad;
Int vil;
Int tag;
Void * pspot;
Public;
Spot (int xi, int yi, int ri);
Point (xi, yi)}
{int size ;
vir =0;tag=0;rad=ri;
Size=image size (xi-ri; yi-ri; xis ri; yiri);
Pspot=neo char [Size];
}
~ Spot ()
{h del();
tag =0;
Delete pspot;
}
Voit show ()
{If (tag==0)
{Cirele (x, y, rad);
Flood siell (x, y, getcolor ());
Get image (x-rad, y-rad, y+rad, pspot);
Tag=1};
Else
Putimage (x-rad, y-rad, pspot, XOR-PUT);
Vi1=1;
}
Void hide ()
```

```
{If (vi1==0) return;
Putimage (x-rad,y-rad, pspot, XOR-PUT);
Vi1=0;
}
Void move (int xn, int yn)
{Hide ();
x- xn, y-yn;
Shov ();
}
Viod vary (float dr)
{Floata;
Int size;
Hide ();
Tag=0;
Delete pspot;
A=dr*rad;
If (a<=0) rad=0;
Else rad= (int) a;
Size=imagerize (x-rad; y-rad, x+rad, y+rad);
New char [size];
Show ();
}
int& giver (void);
{Return rad;}
};
# Endif
```

Spot sinfida construktor destruktor ~ spot () va beshta metod ko'rsatilgan:

Show ()-- ekranga doirani chizib, bitli tasvirni hotiraga olish;

Hide ()-- ekrandan doira tasvirini uchirish;

Move ()--tasvirni ekranning bitta joyiga ko’chirish;

Vary ()--ekrandagi tasvirni o’zgartirish (kichkinalashtirish yoki kattalashtirish);

Giver () --doira radiusiga murojatni ta'minlash;

Point sinfidan Jpot sinfi naslga nuqta markazi (h,u) koordinatalarini va givek, givey metodlarni oladi, Point : : show () va point : : move () metodini huddi shu nomli yangi funktsiyalar bilan almashtirilgan.point :: hide funktsiyasi nomi o’tmaydi chunki point sinfida u hususiy (private) statiyasiga ega. Stop() konstruktor uch parametrga ega -mernez koordinatalari (xi,yi) va doira radiusi (ri).

Avval point sinfi konstruktori chaqiriladi bu konstruktor xi,yi ga mos keluvchi haqiqiy parametr asosida doira markazini aniqlaydi. Asosiy sinf konstruktori har doim hosilaviy sinf konstruktoridan oldin chaqiriladi. So’ngra spot() sinfi konstruktolari boshlanadi. Bu konstruktor vi1, tag parametrlarining boshlang’ich qiymatini aniqlaydi va ri gamos keluvchi haqiqiy parametr qiymati asosida doira radiusi red aniqlanadi. Standart funktsiya imagelizi yordamida doira joylashuvchi kvadratik operativ hotirada aniqlash uchun zarur bo’lgan hotira hajmi hisoblanadi. Kerakli hotira new standart operatsiya yordamida ajratib size elimentidan iborat chur massivlar yoziladi. Agar aytilgan hotira spot sinfida protected statutisiga ega bo’lgan spot ko’rsatkichiga ulanadi.

Sinfning har bir ob'ekti yaratilganda sinf konstruktori chaqirilib, ob'ekt uchun kerakli hotira yaratish va liniyalizatsiya qilish vazifalarini bajaradi. Ob'ekt yuqotilganda yoki sinf ta'sir doirasidan tashqariga chiqilganda teskari inertsiyalarni boshqarish kerak bo’lib,bu lediatsiyalar ichida eng kirishli hotirani ozod qilishdir. Bu vazifalarni boshqarish uchun sinfga mahsus funktsiya dastruktor kiritiladi. Dastruktor quyidagi shaklga ega bo’lgan aniq nomga ega ~ sinf-nomi.

Dastruktor hatto void tipidagi parametrlarga ega bo’lmaydi va hatto void tipidagi qiymat qaytarmaydi. Destruktor statusi ikki e'lon qilinmagan bo’lsa umumiydir. Sodda sinflarda dastruktor avtomatik aniqlanadi, misol uchun paint

sinfida destruktor e'lon qilinmagan va pomilyator qo’yidagi dasturlarni avtomatik chaqiradi

Point () {};

Spot sinfida destruktor anik kurinishga ega;

Spot () {hide (); tag=0;delete [] p1pot;}

Bu dastruktor vazifalari doira tasvirini spot::hide() funktsiyasi orqali o’chirish; tag belgisiga 0 qiymatini berish; ob'ekt bitli tasvirni saqlash uchun ajratilgan hotirani tozalash.

Destruktorlar naslga o’tmaydi, shuning uchun hosilaviy sinfda destruktor mavjud bo’lmasa asosiy sinfdagi destruktor chaqirilmaydi. Balki kominator tomonidan yaratiladi. Ko’rilayotgan misolda qo’yidagicha;

Public: ~spot () {~point ();}

Asosiy sinflar destruktorlar ruyhatda ko’rsatilganidek teskari tartibda boshariladi. Shunday qilib ob'ektlarni o’chirish tartibi yaratilish tartibiga teskaridir.Sinf ob'ektini va asosiy sinflar uchun destruktorlar avtomatik chaqiriladi. Agar ob'ekt yaratilganda dasturda hotira ajratilgan bo’lsa destruktor dasturda chaqirilishi lozim. Spot sinfi ob'ektlari bilan ishlovchi dasturni keltiramiz:

```
# Include<graphics.h>
# Include<conio. H >
# Include "spot. Cpp"
 Void main ()
{Int dr=DETECT, mod;
Initgraph (8dr, 8mod);
{Spot A (200,50,20);
Spot D (500,200,30);
A. show;
Det ch ();
O. Show ();
Det ch; A. Move (50,60);
```

```
Det ch (); {closegrap ();}
```

ELIPS, SPOT va SPOTELLI sinflarida POINT sinfining x,y komponentalari nuqtasining ekrandagi koordinatalari naslga o’tadi.POIN sinfida ular himoyalangan

(protected) sifatida aniqlangan va bu statusti hosilaviy sinflarda ham saqlab qoladi. SPOTELLI sinfi konstruktori hech qanday vazifa bajarmaydi ketma-ket

ELLIPS sinfi va SPOT sinfi konstruktorlari chaqiriladi.U holda yaratilayotgan shakillar markazlari ustma-ust tushadi, doira radiusi sifatida ellipsning radiusi olinadi. Bu holda ishlatiladigan min() funktsiyasi hususiy (private) joylashtirilgan (inline) funktsiya sifatida aniqlangan. Bevosita bo’lmagan sinflar ob'ektlarning bir necha marta yaratilishi oldini

olish uchun, bu asosiy sinf virtual sinf deb e'lon qilinadi. Masalan qo’yidagi ta'rifda H

sinfi virtual bo’ladi:

```
Class base
 {int j=0,char c='*' )
 {jj=j;
 cc=c;
 }
 };
 double dd;
 public:
 dbase (double d=0.0):base()
 {dd=d;}
 };
 class j base ; public virtual base
 {float ff ;
 public ;
 jbase (float j=0.0):base ()
```

```
{ff=f;}
};
class top:public dbase,public jbase
{long tt;
public;
top(long t=0):dbase (),jbase ()
{tt=t;}
};
void main()
{cont<<"\n asosiy sinf:size of (base) ="<c h zeaf
(base )
cont << "\n tugri asos :size of (d base)="<<size
of(dbase)
cont << :\ n tugri asos :size of (fbase)="<<size
of (fbase)
cont <<\n hosilaviy sinf :sizeof (top)="<<sizeof
(top);
```

ko’plab vorislikda bitta asosiy sinf hosilaviy virtual yoki no virtual bir necha marta kirishi mumkin. Misol keltiramiz:

```
clarr x{…..};
clarr'/:virtual public x{….}:
clarr z:virtual public x{…};
clarr b :virtual public x{…};
clarr c:virtual public x{….};
clarr e:public x{….};
clarr d:public x{…..};
clarr a :public p;public b;
public y ,public z ;
public c ,public e{…..};
```

Bu misolda A sinf ob'ekti o’z ichiga H sinfining 3 ob'ektini bitta virtual B ,Y,C,Z sinflari bilan birgalikda ishlatuvchi ; va ikki virtual mos ravishda D va E sinflariga tegishli bulgan.Shunday kilib virtuallik bu sinfning hususiyati emas, balki nasldan-nalga

utishning hususiyatidir.

Nasldorlik va ko’plik nasldorlikdan foydalanilganda har hil sinflarning bir hil nomli kompanentalariga murojaat qilganda har hil talqin yuzaga keladi. Bu har hil talqin oldini olishning eng sodda va ishonchli usuli --kompanentalarining kvalifikatsiyalangan nomlaridan foydalanishdir. Kompanenta nomini kvalifikatsiya qilish uchun sinf nomi ishlatiladi. Qo’yidagi misollar kvalifikatsiyalangan nomlardan foydalanish ko’rsatilgan.

```
 Clarr x {public:int d;…};
Clarr y {public:int d;…};
 Clarr z :public y,public y,
 {public ;
 int d;
 --------
 d=y::d+y::d;
 --------
 };
```

Bu dasturning boshqa ko’rilgan dasturlardan printsipial farqi spot sinfida HIDE() kompanenti funktsiyasiga murojat qiluvchi dastruktor ishlashi bilan bog’lik, ishlab chiqarishni blokning mavjudligidir. Agar dasturli point sinfli dasturlar kabi ichki bloksiz yaratilsa dastruktor ochik murojat qilinmaganda faqat dastur tugaganda ya'ni grafik rejim berkitilganda chaqiriladi. Bu hatoni ikki yul bilan oldini olish mumkin. Yoki A va D ob'ektlani yuqotish uchun destruktorni dasturda chaqirish va shundan so’ng grafik rejimni berkitish yoki grafik rejimni initsializatsiya qilgandan so’ng A va D ob'ektlar aniqlangan ichki blok kiriting.

Bu ichki blokdan chiqilayotganda A va D ob'ektlar avtomatik uchiriladilar.Buning uchun dastruktor ikki marta avtomatik chaqiriladi.Grafik rejim A,D ob'ektlar yuqotilgandan sung tashqi blokda berkitiladi.

Qo’yidagi misolda ikkinchi usul qo’llanilishi ko’rsatilgan:

```
-----
getch ();d. Vary (3);
getch();
A.spot ::<spot();
Getch();
D.spot::<spot ();
Closegraph();
}
```

Sodda misol

Misol uchun computer_screen sinfi mavjud bo’lsin:

```
class computer_screen
{
public:
 computer_screen(char *, long, int, int);
 void show_screen(void);
private:
 char type[32] ;
 long colors;
 int x_resolution;
 int y_resolution;
};
```

Bundan tashqari mother_board sinfi ham mavjud bo’lsin:

```
class mother_board
{
public:
```

```
 mother_board(int, int, int);
 void show_mother_board(void);
private:
 int processor;
 int speed;
int RAM;
};
```

Bu sinflardan foydalanilgan holda yangi computer sinfini yaratish mumkin:

```
class computer : public computer_screen, public
mother_board
{
public:
 computer(char *, int, float, char *, long, int,
int, int, int, int);
 void show_computer(void);
private:
 char name[64];
 int hard_disk;
 float floppy;
};
```

Bu sinf uzining avlod sinflarini sinf nomidan so'ng ko'rsatadir.

class computer : public computer_screen, public mother_board //————————> Qo'yidagi COMPUTER. CPP dasturida computer_screen va mother_board sinflari asosida computer sinfi yaratiladi:

```
#include <iostream.h>
#include <string.h>
class computer_screen
{
public:
```

```
 computer_screen(char *, long, int, int);
 void show_screen(void);
private:
 char type[32];
 long colors;
 int x_resolution;
 int y_resolution;
};
computer_screen::computer_screen(char *type, long
colors, int x_res, int y_ree)
{
 strcpy(computer_screen::type, type);
 computer_screen::colors = colors;
 computer_screen::x_resolution = x_res;
 computer_screen::y_resolution = y_res;
}
void computer_screen::show_screen(void)
{
 cout << "Tip ekrana: " << type << endl;
 cout << "Tsvetov: " << colors << endl;
 cout << "Razreshenie: " << x_resolution << " na "
<< y_resolution << endl;
}
class mother_board
{
public:
 mother_board(int, int, int);
 void show_mother_board(void);
private:
 int processor;
```

```
    int speed;
     int RAM;
    };
    mother_board::mother_board(int processor, int
speed, int RAM)
    {
     mother_board::processor = processor;
     mother_board::speed = speed;
     mother_board::RAM = ram;
    }
    void mother_board::show_mother_board(void)
    {
     cout << "Protsessor: " << processor << endl;
     cout << "Chastota: " << speed << "MGts" << endl;
     cout << "OZU: " << RAM << " MVayt" << endl;
    }
    class computer : public computer_screen, public
mother_board
    {
    public:
     computer(char *, int, float, char *, long, int,
int, int, int, int);
     void show_computerf void);
    private:
     char name [64];
     int hard_disk;
     float floppy;
    };
    computer::computer(char *name, int hard_disk, float
floppy, char *screen, long colors, int x_res, int
```

```
y_res, int processor, int speed, int RAM) :
computer_screen(screen, colors, x_res, y_res),
mother_board(processor, speed, ram)
    {
     strcpy(computer::name, name);
     computer::hard_disk = hard_disk;
     computer::floppy = floppy;
    }
    void computer::show_computer(void)
    {
     cout << "Tip: " << name << endl;
     cout << "Jestkiy disk: " << hard_disk << "MVayt"
<< endl;
     cout << "Gibkiy disk: " << floppy << "MVayt" <<
endl;
     show_mother_board();
     show_screen();
    }
    void main(void)
    {
     computer my_pc("Compaq", 212, 1.44, "SVGA",
16000000, 640, 480, 486, 66, 8);
     my_pc.show_computer();
    }
```

Bu misolda computer sinfi konstruktori, mother_board va computer_screen konstruktorlarini chaqiradi: computer::computer(char *name, int hard_disk, float floppy, char *screen, long colors, int x_res, int y_res, int processor, int speed, int RAM) : computer_screen(screen, colors, x_res, y_res), mother_board(processor, speed, RAM)

Nazorat savollari

Grade sinfi element lari. "B+" kabi qatordan gradus yasaydigan konstruktorni ta’minlang. Graduslarni ularning raqamli qiymatlariga o‘giradigan funksiya ham sizga kerak bo‘ladi. (masalan, "B+" dan 3.3 kelib chiqadi).

P9.13 Davlat nomi, uning aholisi va maydonini saqlaydigan Davlat sinfini aniqlang. Ushbu sinfdan foydalanib, davlatlar guruhini o‘qiydigan va chop etadigan dastur yozing.

• Eng katta maydonli davlat.

• Aholisi eng ko‘p davlat.

• Aholi zichligi eng katta davlat (har bir kilometr yoki mil maydonga aholi zichligi).

11-ma’ruza. Merosxo’rlik

Ma’ruza rejasi:

11.1 Meros ierarxiyasi.

11.2 Mavjud sinflarni joriy etish.

Kalit so’zlar:*, ro’yxat, manzil, nolinchi ko’rchsatkich, tugun, adres olish &, bo’shatish, ko’rsatkich, virtual destruktor, xotira, xotira chiqishi, destruktor, toifani o’zlashtirish, Inkapsulyasiya, inline, polimorfizm, vorislik, meros sinflar, MSINF a’zo destruktori.*

Bir biriga bog‘langan sinflar odatda umumiy xarakatga egadir. Masalan, kurak, xaskash va qaychilarning barchasi bog‘dagi ishlarni qilishga mo‘ljallangan. Ushbu bobda, siz merosxo‘rlik tushunchasining maxsus va umumiy sinflar o‘rtasidagi munosabatlarni qanday ifodalashini o‘rganasiz. Merosxo‘rlikni ishlatish orqali, sinflar o‘rtasidagi kodni bo‘lishishingiz, hamda bir necha sinflar uchun xizmat ko‘rsatishiniz mumkin.

11.1 Merosxo‘rlik Ierarxiyasi

Ob’ektga yo‘naltirilgan sxemada, merosxo‘rlik bu umumiy sinf (asosiy sinf) va maxsus sinf(quyi sinf**)** o‘rtasidagi munosabatdir. YOrdamchi sinf ma’lumot va aniqlanmagan xarakatlarni asosiy sinfdan meros qilib oladi. Masalan, 1-shaklda tasvirlangan turli xil transport vositalarining munosabalariga ahamiyat bering.

Avtomobillar barcha transport vositalari uchun umumiy bo‘lgan xususiyatga, ya’ni yo‘lvchilarni bir joydan ikkinchi joyga etkazish vazifasini

bajarmoqdalar. Bu biz Avtomobilь sinfi Transport sinfidan meros qilib olyapti deb aytamiz. Bu o‘rinda, Transport vositasi asosiy sinf va Avtomobilь sinfi quyi sinf hisoblanadi.

Norasmiy ma’noda, merosxo‘rlik munosabati relationship is called the *is-a* rela-tionship. Bu biz 9.7 bo‘limda da muxokama qilgan *ega bo‘lish* munosabatidan

farqlanadi. Har bir avtomobiль transport vositasi*dir*. Har bir transport vositasi motorga *ega*.

Merosxo‘rlik munosabati juda kuchlidir, chunki u algoritmlar bilan ob’ektlarni turli xil sinflarda qayti ishlatish imkonini beradi. Balki bizda Transport ob’ktini boshqaruvchi algoritm mavdud bo‘lishi mumkin. CHunki avtomobiль transport vositasining maxsus turlaridan biri bo‘lib, avtomobiль ob’ektini shunday algoritm bilan ta’minlashimiz mumkinki,u to‘g‘ri ishlay oladi. Bu o‘rinni bosish prinsipiga doir misol bo‘lib,doimo asosiy sinf ob’ekti kerak bo‘lgan vaqtda yordamchi sinf obьektidan foydalanishingiz mumkinligidan dalolat beradi.

Merosxo‘rlik munosabati 1 shaklda ko‘psatilgani kabi, sinflarning ko‘proq ixtisolashtirilgan ierarxiyalarni oshirira oladi. 2 shaklda ko‘rsatilgan C++ sinflarining ishlash yo‘li bunday ierarxiyaning yana bir misolidir. 2 shaklda merosxo‘rlikdagi asosiy va quyi sinflarni asosiy sinfga yo‘naltiruvchi o‘q bilan birlashishida UML belgilaridan foydalanilgan.

Ko‘rib turganingizdek, ifstream (kiruchi oqim bo‘lib, ma’lumotni fayldan o‘qiy oladi) "istream" ning maxsus xolatidir (kiruvchi oqim bo‘lib, ma’lumoni istalgan manba’dan o‘qiy oladi). Agar "ifstream" ga ega bo‘lsangiz, bu "istream"da kutilayotgan funksiya uchun argument bo‘lib xizmat qiladi.

void process_input(istream& in) // Can call with an ifstream object

Nima uchun ifstream ob’ektlari o‘rniga uning vazifasini bajaruvchi istream ob’ektlarini beramiz?

Bu funksiyaning foydasi ko‘proq, chunki u har qanday kiruvchi oqim bilan ishlay oladi (masalan istringstream kabi, imtixon qilishga qaysi bii qulay bo‘lsa).

Bu yana ishdagi o‘rin bosish tamoyilidir.

Bu bo‘limda, biz sinflarning sodda

ierarxiyasini ko‘rib chiqamiz. Ehtimol, kompьyuterga oid testlarni ishlab borasiz. Test bir necha savollar

tartibidan tashkil topgan:

• Bo‘sh qatorlarni to‘ldiring

• Tanlash (bir yoki bir necha)

• Sonli (bu erdagi taxminiy javob

ok; masalan, 1.33 agar xaqiqiy javob 4/3

bo‘lsa)

• Erkin javob

3- shaklda ana shu savol turlari uchun merosxo‘rlik ierarxiyasi ko‘rsatilgan.

Bu ierarxiyaning o‘zagida Savol turi turibdi. Savol matnni

monitorda ko‘sata oladi va berilgan javobning qanchalik to‘g‘riligini tekshiradi:

```
class Question
{
public:
 Question();
 void set_text(string question_text);
 void set_answer(string correct_response);
 bool check_answer(string response) const;
 void display() const;
private:
 string text;
 string answer;
};
```

Matnning qanday aks etishi savol turiga bog‘liq. Keyinroq bu bobda, ba’zi xilma xilliklarni ko‘rib chiqasiz, lekin asosiy sinf savol matnini shundayligicha "cout" ga uzatadi. Javobning tekshirilishi ham savol turiga bog‘liq. Aytib o‘tilganidek, raqamli savol taxminiy javoblarni qabul qilishi mumkin. (Mashq P10.1 ga qarang). P10.3 mashqda, siz javobni tekshirilishining boshqa yo‘lini ko‘rasiz. Lekin asosiy sinfda, biz javobni to‘g‘ri javobga aynan mos tushishini talab qilamiz.

Quyidagi bo‘limlarda, siz asosiy sinfdan komponentlik funksiyasi va ma’lumotlar elementini meros qilib oluvchi yordamchi sinflarni qanday qilib

xosil qilishni ko‘rib chiqasiz.

Bu erda Savol sinfining joriy qilinishi va sodda test dasturi berilgan.

SHuni nazarda tutingki, Savol sinfi konstruktori hech qanday ishni amalga oshirishi kerak emas, chunki matn va javob maydonlari avtomatik ravishda bo‘sh qatorga joylashtirilgan. "boolalpha" oqimi boshqaruvchisi asosiy vazifasi orqali Bulev (mantiqiy) qiymatini keltirib chiqarib, 1 yoki 0 ni o‘rniga *to‘g‘ri* yoki *noto‘g‘ri* bo‘lib ko‘rsatilmoqda.

ch10/quiz1/test.cpp

```
1 #include <iostream>
2 #include <sstream>
3 #include <string>
4
5 using namespace std;
6
7 class Question
8 {
9 public:
10 /**
11 Constructs a question with empty text and
answer.
12 */
13 Question();
14
15 /**
16 @param question_text the text of this question
17 */
18 void set_text(string question_text);
```

```
19
20 /**
21 @param correct_response the answer for this
question
22 */
23 void set_answer(string correct_response);
24
25 /**
26 @param response the response to check
27 @return true if the response was correct, false
otherwise
28 */
29 bool check_answer(string response) const;
30
31 /**
32 Displays this question.
33 */
34 void display() const;
35
36 private:
37 string text;
38 string answer;
39 };
40
41 Question::Question()
42 {
43 }
44
45 void Question::set_text(string question_text)
```

```
46 {
47 text = question_text;
48 }
49
50 void Question::set_answer(string correct_response)
51 {
52 answer = correct_response;
53 }
54
55 bool Question::check_answer(string response) const
56 {
57 return response == answer;
58 }
59
60 void Question::display() const
61 {
62 cout << text << endl;
63 }
64
65 int main()
66 {
67 string response;
68 cout << boolalpha; // Show Boolean values as true,
false
69
70 Question q1;
71 q1.set_text("Who was the inventor of C++?");
72 q1.set_answer("Bjarne Stroustrup");
73
74 q1.display();
```

```
75 cout << "Your answer: ";
76 getline(cin, response);
77 cout << q1.check_answer(response) << endl;
78
79 return 0;
80 }
```

Dasturning bajarilishi

Kim C++ni yaratgan?

Sizning javobingiz : Bjarne Stroustrup

to‘g‘ri

1. *Boshqaruvchi* va *Ishchi* sinflarini ko‘rib chiqing. Qaysi biri asosiy sinf va qaysinisi yordamchi sinfga kiradi?

2. Quyidagi sinflar orasidagi merosxo‘rlik munosabatlari qanday: Bankhisobraqami, Joriyhisobraqam, and Jamg‘arma?

3. do_something funksiyasini ko‘rib chiqing(istream& stream). 2-shaklda berilgan ob’ektlardan berilgan funksiyaga o‘ta oladigan barcha oqim sinflarini ro‘yxatini tuzing.

4. do_something(Car& c) funksiyasini ko‘rib chiqing. 1-shaklda berilgan transport vositalaridan berilgan funksiyaga o‘ta olmaydigan sinflarni ro‘yxatini tuzing.

5. Test sinfi Savol sinfidan meros olishi kerakmi? Nima uchun?

Amaliyot Ushbu bob oxirida berilgan mashqlarni ishlab ko‘rishingiz mumkin: R10.1, R10.2,

R10.5

11.2. Mavjud sinflarni joriy etish

C++ da, siz quyi sinfni asosiy sinfdan uni qanday xususiyat o‘zgartirib ko‘rsatishiga qarab xosil qilasiz. Quyi sinf uchun yangi bo‘lgan komponentlik funksiyalarini aniqlaysiz. Quyi sinf komponentlik funksiyasining barchasini asosiy

sinfdan meros qilib oladi, lekin agar meros qilib olingan xarakat noto‘g‘ri bo‘lsa, uning joriy qilinishini o‘zgartirishingiz mumkin.

Quyi sinf ma’lumotlar elementining barchasini asosiy sinfdan avtomatik ravishda meros qilib oladi. Siz faqatgina ma’lumotlar elementini aniqlaysiz.

Quyida quyi sinf tafsifi uchun sintaksis berilgan:

```
class ChoiceQuestion : public Question
{
public:
New and changed member functions
private:
Additional data members
};
```

Ushbu belgi : merosxo‘rlikni anglatadi. Zaxiradagi public so‘zi texnik sabablar uchun talab etiladi (449 betdagi 10.1 dagi Odatiy xatoga qarang).

SavollarVarianti ob’ekti Savol ob’ektidan uchta jihatdan farq qiladi:

- Undagi ob’ektlar javob uchun turli variantlarni saqlaydi.
- Bu erda boshqa variantni qo‘shish uchun komponentlik funksiyasi mavjud.
- Javob beruvchi tanlab olishi uchun, SavollarVarianti sinfi Monitor funksiyasi ushbu variantlarni ko‘rsatadi.

Qachonki, SavollarVarianti Savol sinfidan meros olganda, ushbu uch tafovutni xosil qilishi talab etiladi.

```
class ChoiceQuestion : public Question
{
public:
 ChoiceQuestion();
 void add_choice(string choice, bool correct);
 void display() const;
private:
 vector<string> choices;
```

};

4 -shaklda SavollarVarianti ob’ektining joylashuvi tasvirlangan. U *Savol*ning asosiy sinfidan ma’lumotlarning *matn* va *javob* elementlarini meros qilib olmoqda, shuningdek qo‘shimcha ma’lumotlar elementini: *variantlar vektorini* qo‘shmoqda.

add_choice funksiyasi SavollaVarianti sinfiga xos.

Siz buni faqatgina SavollarVarianti ob’ektlari uchun qo‘llay olasiz, umumiy Savol ob’ektlari uchun emas. SHunga qaramay, monitor funksiyasi asosiy sinfda mavjud bo‘lgan funksiyaning boshqa ko‘rinishi bo‘lib, yordamchi sinfga kerak bo‘lgan maxsus zaruriyatlarni qamrab olgan. Bu xolatda biz yordamchi sinf bu funksiyani o‘rnini bosyapti deb aytamiz. Buning qanday bo‘lishini 10.3 -bo‘limda ko‘rasiz.

SavollarVarianti sinfi tavsifida siz faqat yangi komponentlik funksiyalarini va ma’lumotlar elementini aniqlaysiz. Savollar sinfidagi qolgan barcha komponentlik funksiyalari va ma’lumotlar elementlari Savollar sinfi orqali avtomatik ravishda meros qilib olinadi. Masalan, har bir SavollarVarianti ob’ekti hamon matn va javob ma’lumotlar elemetiga egadir, shuningdek set_text, set_answer, va check_answer komponentlik funksiyalari ham mavjuddir.

Nazorat savollari:

1. Meros tushunchasi nima?
2. Merosxo’rlik usullari va hossalarini ayting?
3. Merosni tashkil qilish qanday amalga oshiriladi?

12-Ma’ruza. Merosxo’rlik

Ma’ruza rejasi:

12.1 Komponentlik funksiyasini.

12.2 Virtual funksiyalar va polimorfizm.

Kalit so’zlar:*, ro’yxat, manzil, nolinchi ko’rchsatkich, tugun, adres olish &, bo’shatish, ko’rsatkich, virtual destruktor, xotira, xotira chiqishi, destruktor, toifani o’zlashtirish, Inkapsulyasiya, inline, polimorfizm, vorislik, meros sinflar, virtual, MSINF a’zo destruktori.*

12.1 Komponentlik funksiyasini

YOrdamchi sinf nomidan keyin turadigan ikki nuqtadan so‘ng qo‘yiladigan zaxiradagi public so‘zining esdan chiqishi odatiy xatodir.

```
class ChoiceQuestion : Question // Error
{
 ...
};
```

Sinf tavsifi uzatiladi. ChoiceQuestion hamon Question dan meros oladi, lekin norasmiy tarzda meros oladi. YA’ni, ChoiceQuestion dagi komponentlik funksiyasining o‘zi Question dagi komponentlik funksiyasini chaqira oladi. Qachonki boshqa funksiya Question komponentlik funksiyasi ob’ektini faollashtirsa, kompiller xato bo‘lganligi haqida belgi beradi:

```
int main()
{
ChoiceQuestion q;
 ...
 cout << q.check_answer(response); // Error
}
```

Bunday shaxsiy merosxo‘rlikning foydasi kamdan kam tegadi. Aslida, birinchi galda bu merosxo‘rlikning ishlatilish ruxini buzadi,—ya’ni, asosiy sinf ob’ektlari kabi qulay ob’ektlarni yaratadi. Siz doimo ochiq merosxo‘rlikdan foydalanib, yordamchi sinf tavsifini zaxiradagi public so‘zi bilan ta’minlashni unutmasligingiz kerak.

Asosiy Sinf Komponentlaridan Nusxa Olish

YOrdamchi sinf asosiy sinf ma’lumotlar elementidan foydalana ololmaydi.

```
ChoiceQuestion::ChoiceQuestion(string question_text)
{
 text = question_text; // Error—tries to access private base-class member
}
```

Kompilyator xato qilganda, bu muammoni echish uchun odatda boshlovchilar ayni nomli boshqa ma’lumot elementini yordamchi sinfga biriktiradilar:

```
class ChoiceQuestion : public Question
{
 ...
private:
 vector<string> choices;
 string text; // Don’t!
}
```

Albatta, konstruktor dasturni tuzadi, lekin u to‘g‘ri matnni o‘rnatmaydi! Bu kabi ChoiceQuestion ob’ekti ikki xil ma’lumot elementiga ega, ikkisi ham matn deb nomlanadi. Konstruktor birini o‘rnatadi va namoyish komponentlik funksiyasi ikkinchisini namoyish qiladi.

Asosiy sinf ma’lumot elementidan bexuda nusxa olgandan ko‘ra, asosiy sinf komponentini yangilovchi komponentlik funksiyasiga murojat qilishingiz darkor,

misolda keltirilgan set_text funksiyasi kabi.

Qiymatlar Uchun YAgona Sinf O‘zgarishidan foydalanish,

Harakatdagi O‘zgarish Merosxo‘rligi

Merosxo‘rlikning maqsadi turli xarakatdagi ob’ektlarni modellashtirishdir. Talabalar merosxo‘rlikni ilk bor o‘rganganlarida, uni xaddan tashqari ishlatishga moyildirlar. O‘zgarishlar oddiy ma’lumotlar elementi yordamida na’moyon bo‘lishiga qaramay, bir necha sinflarni yaratadilar. Avtoparkda yoqilg‘i ishlatilish samaradorligini kuzatuvchi bir dasturni ko‘rib chiqsak:bosib o‘tilgan masofa va quyilgan yoqilg‘i miqdorini qayd qilamiz. Parkdagi ba’zi avtomobillar gibridlardir. Siz HybridCar xosila sinfini yaratmoqchimisiz? Bu dasturiy vositada emas. Gibridlar xaydashda va yoqilg‘i quyishga kelganda boshqa avtomobillardan farq qilmaydilar.

Asosiy sinf konstruktoriga murojaat qilish

YOrdamchi sinf ob’ektini yaratish jarayonini tasavur qiling. YOrdamchi sinf kostruktori faqatgina ushbu sinf ma’lumotlar elementini avvalgi xolatiga qaytarishi mumkin. Lekin asosiy sinf ma’lumotlar elemeti ham avalgi xolatga qaytarilishni talab etadi. Biroq siz aksini qilsangiz, asosiy sinf ma’lumotlar elementi asosiy sinfning mavjud konstruktori orqali avvalgi xolatga qaytariladilar.

Boshqa konstruktorni aniqlash uchun, 9.1- maxsus mavzuda tasvirlangani kabi, initsializator ro‘yxatini ishlating.

Asosiy sinf nomini va initsializator ro‘yxatidan konstruksiya parametrlarini aniqlang. Masalan, Question asosiy sinfi savol matnini o‘rnatuvchi konstruktorga ega bo‘lgan deb hisoblang. Bu erda yordamchi sinf konstruktori asosiy sinf konstruktoriga qanday murojaat qilishi ko‘rsatilgan.

```
ChoiceQuestion::ChoiceQuestion(string question_text)
 : Question(question_text)
{
}
```

YOrdamchi sinf konstrutori { } ichidagi kodni bajarishdan oldin murojaat qiladi. Na’muna qilib berilgan dasturimizda, biz asosiy sinfning defolt konstruktordan foydalandik. SHunga qaramay, agar asosiy

sinfda defolt konstruktor bo‘lmasa, initsiyalizator ro‘yxati sintaksisidan foydalanishiingiz zarur.

YOrdamchi sinf komponentlik funksiyalarini asosiy sinfdan meros qilib oladi. Agar siz meros qilib olingan komponentlik funksiyasi harakatidan qoniqmasangiz, yordamchi sinfga yangilik kiritish orqali, uni bekor qilishingiz mumkin.

ChoiceQuestion sinfidagi display funksiyasini ko‘rib chiqing. Asosiy

sinfning display funksiyasi, javob variantlarini taqdim etish maqsadida bekor qilinishi lozim. Asosan, asosiy sinf funksiyasi:

• Savol matnini ko‘rsatishi.

• Javob variantlarini ko‘rsatishi

kerak bo‘ladi.

Ikkinchi qismi oson xisoblanadi, chunki javob variantlari yordamchi sinfning ma’lumot elementidir.

```
void ChoiceQuestion::display() const
{
 // Display the question text
 ...
 // Display the answer choices
 for (int i = 0; i < choices.size(); i++)
 {
 cout << i + 1 << ": " << choices[i] << endl;
 }
}
```

Lekin siz qanday qilib savol matnini olasiz? Siz asosiy sinfning matn elementidan to‘g‘ridan to‘g‘ri foydalana olmaysiz, chunki u yopiqdir. Buning o‘rniga, asosiy sinfning display funksiyasini chaqirasiz.

```
void ChoiceQuestion::display() const
```

```
{
 // Display the question text
 display(); // Invokes implicit parameter.display()
 // Display the answer choices
 ...
}
```

SHunga qaramay, bu yaxshi ish bermaydi. CHunki ChoiceQuestion::display dagi yashirin parametr ChoiceQuestion turiga mansub bo‘lib, ChoiceQuestion sinfida dis-play deb nomlanuvchi funksiyaga ega. Bu funksiya chaqiriladi—lekin bu siz ayni vaqtda yozgan funksiyangizning o‘zidir! Bu funksiya o‘zini o‘zi qayta qayta chaqiraveradi.

Savol matnini ko‘rsatish uchun, chaqirmoqchi bo‘lgan display deb nomlanuvchi funksiyaning qaysi birini tanlashni aniqlashingiz shart . Sizga Question::display:

kerak:

```
void ChoiceQuestion::display() const
{
 // Display the question text
 Question::display(); // OK
 // Display the answer choices
 ...
}
```

Siz funksiyani bekor qilganingizda, odatda asosiy sinf versiyasining funksionalligini oshirishga urinasiz. SHuning uchun, asosiy sinf versiyasini kengaytirishdan oldin, uni faollashtrishingiz darkor. Uni faollashtirish uchun, BaseClass::function notatsiyasidan foydalanishingiz lozim. Ammo, asosiy sinf funksiyasini chaqirishingiz majburiy emas. Ba’zida, yordamchi sinf asosiy sinf funksiyasini bekor qilib,butunlay boshqa funksionallikni aniqlaydi

Quyida Question ob’ekti va Choice- Question ob’ekti aqiq ko‘rsatilgan to‘liq dastur berilgan. (Siz yuqorida ko‘rgan Question sinfining tavsifi question.h ga joylashtirilgan va question.cpp. ga joriy qilingan) Bu misol asosiy sinfdan maxsus sinfni shakllantirishda merosxo‘rlikning ishlatilishi ko‘rsatilgan.

ch10/quiz2/test.cpp

```
 1 #include <iostream> 2 #include <sstream>
3 #include <vector>
4 #include "question.h"
5
6 class ChoiceQuestion : public Question
7 {
8 public:
9 /**
10 Constructs a choice question with no choices.
11 */
12 ChoiceQuestion();
13
14 /**
15 Adds an answer choice to this question.
16 @param choice the choice to add
17 @param correct true if this is the correct
choice, false otherwise
18 */
19 void add_choice(string choice, bool correct);
20
21 void display() const;
22 private:
23 vector<string> choices;
24 };
```

```
25
26 ChoiceQuestion::ChoiceQuestion()
27 {
28 }
29
30 void ChoiceQuestion::add_choice(string choice,
bool correct)
31 {
32 choices.push_back(choice);
33 if (correct)
34 {
35 // Convert choices.size() to string
36 ostringstream stream;
37 stream << choices.size();
38 string num_str = stream.str();
39 set_answer(num_str);
40 }
41 }
42
43 void ChoiceQuestion::display() const
44 {
45 // Display the question text
46 Question::display();
47 // Display the answer choices
48 for (int i = 0; i < choices.size(); i++)
49 {
50 cout << i + 1 << ": " << choices[i] << endl;
51 }
52 }
53
```

```
int main()

string response;
cout << boolalpha;

// Ask a basic question

Question q1;
q1.set_text("Who was the inventor of C++?");
q1.set_answer("Bjarne Stroustrup");

q1.display();
cout << "Your answer: ";
getline(cin, response);
cout << q1.check_answer(response) << endl;

// Ask a choice question

ChoiceQuestion q2;
q2.set_text("In which country was the inventor
of C++ born?");
q2.add_choice("Australia", false);
q2.add_choice("Denmark", true);
q2.add_choice("Korea", false);
q2.add_choice("United States", false);

q2.display();
cout << "Your answer: ";
getline(cin, response);
cout << q2.check_answer(response) << endl;
```

```
83
84 return 0;
85 }
```

program run

Who was the inventor of C++?

Your answer: Bjarne Stroustrup

true

In which country was the inventor of C++ born?

1: Australia

2: Denmark

3: Korea

4: United States

Your answer: 2

true 11. Quyidagi display funksiyasini joriy qilishda qanday xatolikka yo‘l qo‘yilgan?

```
void ChoiceQuestion::display() const
{
 cout << text << endl;
 for (int i = 0; i < choices.size(); i++)
 {
 cout << i + 1 << ": " << choices[i] << endl;
 }
}
```

12. Quyidagi display funksiyasini joriy qilishda qanday xatolikka yo‘l qo‘yilgan?

```
void ChoiceQuestion::display() const
{
 this->display();
 for (int i = 0; i < choices.size(); i++)
```

```
{
10.4 Virtual Funksiyalar va Ko‘p shakllilik
cout << i + 1 << ": " << choices[i] << endl;
}
}
```

13. Asosiy sinfning set_answer funksiyasini chaqiruvchi add_choice funksiyasining joriy qilinishiga yana bir bor qarang. Nima uchun siz Question:: set_answer ni chaqirishingiz kerak emas?

14. 9- mustaqil nazoratning Manager sinfida, get_name funksiyasini shunday bekor qilingki, menedjerlar o‘z nomidan oldin * ega bo‘lsinlar (*Lin, Sally kabi).

15. 9- mustaqil nazoratning Manager sinfida, e get_salary funksiyasini shunday bekor qilingki, u maoshning summasini va bonusini qaytarib bersin.

Amaliyot Bu mashqlarni bobning oxirida ishlab ko‘rishingiz mumkin: R10.12, P10.1.

Asosiy sinf nomini unutish

Asosiy sinf funksiyasi funksionalligini oshirishdagi odatiy xato bu asosiy sinf nomini unutishdir. Masalan, menedjerning maoshini hisoblaganda, maoshni Employee ob’ektining asosidan oling va bonus qo‘shing.

```
double Manager::get_salary() const
{
double base_salary = get_salary();
// Error—should be Employee::get_salary()
return base_salary + bonus;
}
```

Bu erda get_salary() get_salary funksiyasiga tegishli bo‘lib, komponentlik funksiyasining yashirin parametri uchun qo‘llanilgan.YAshirin parametr Manager turiga xos bo‘lib, bu erda Manager::get_salary funksiyasi mavjud vabu funksiya chaqiriladi. Albatta, biz yozgan funksiyaning rekursiv chaqirig‘idir. Buning o‘rniga,

siz qaysi get_salary funksiyasini chaqirmoqchiligingizni aniqlashtiring. Bunday g‘olatda, siz ochiq ravishda Employee::get_salary funksiyasini chaqirishingiz kerak.

Harqachon, yordamchi sinf funksiyasidan asosiy sinf funksiyasini bir xil nom bilan chaqirganingizda, funksiyaga asosiy sinf nomini hisobga olgan holda to‘liq nom berganingizga amin bo‘ling.

Virtual Funksiyalar va Polimorfizm

Oldingi bo‘limlarda merosxo‘rlikdan foydalanishning muxim jixatlarini ko‘rdingiz: asosiy sinfdan maxsus sinfni xosil qilish. Quyidagi bo‘limlarda merosxo‘rlikning yanada kuchliroq dasturiy vositalarini ko‘rib chiqasiz: dastur ishlayotgan vaqtda harakati va turi o‘zgaruvchi ob’ektlar bilan ishlaysiz. Bunday harakat xilma xilligiga virtual funksiyalar tufayli erishiladi. Ob’ektda virtual funksiyani faollashtirganingizda, C++ dastur tizimi aynan qaysi komponentlik funksiyasini chaqirishni aniqlaydi, bu ob’ektning qanday sinfga tegishli ekanligiga bog‘liqdir.

Quyidagi bo‘limlarda, nima uchun sinfi dastur ishlayotgan vaqtda o‘zgaruvchi ob’ektlardan foydalanishda ko‘rsatgichlarni ishlatish kerakligini bilasiz, va qanday qilib virtual funksiyalar keltirilgan ob’ektga mos keluvchi komponentlik funksiyasini tanlashini ko‘rib chiqasiz.

Muammoni Kesib Tashlash

Bu bo‘limda, sinf ierarxiyasida turli xil sinflarga tegishli ob’ektlar yig‘indisi bilan ishlaganda yuzaga keladigan odatiy muammoni muxokama qilamiz.

Agar siz quiz2/test.cpp ning asosiy funksiyasiga nazar tashlasangiz, u erda savolni ko‘rsatuvchi va javoblarni tekshiruvchi bir necha marotaba qaytarilayotgan kodlarni uchratasiz. Agar barcha savollar bir massivda joylashtrilgan bo‘lib, ularni foydalanuvchiga taqdim etishdassikldan foydalanilsa maqsadga muvofiq bo‘lar edi:

```
const int QUIZZES = 2;
Question quiz[QUIZZES];
quiz[0].set_text("Who was the inventor of C++?");
```

```
quiz[0].set_answer("Bjarne Stroustrup");
ChoiceQuestion cq;
cq.set_text("In which country was the inventor of C++ born?");
cq.add_choice("Australia", false);
...
quiz[1] = cq;
for (int i = 0; i < QUIZZES; i++)
{
 quiz[i].display();
 cout << "Your answer: ";
 getline(cin, response);
 cout << quiz[i].check_answer(response) << endl;
}
```

Massivdagi quiz Question turidagi ob’ektga ega. Kompilyator Choice-Question Questionningmaxsus xolati ekanligini anlaydi. SHuning uchun, u variantlar savolini savoldan o‘zlashtirib olishga ruzsat beradi:

```
quiz[1] = cq;
```

Biroq, ChoiceQuestion ob’ekti usta ma’lumotlar elementiga ega, Question ob’ekti

esa faqatgina ikkita ega. YOrdamchi sinf ma’lumotini saqlash uchun xona mavjud emas. Bu ma’lumot, asosiy sinf o‘zgaruvchisidan yordamchi sinf ob’ektini o‘zlashtirganingizda shunchaki kesib tashlanadie (5 -shaklga qarang).

Agar natijani dasturda qo‘llasangiz, variantlar ko‘rsatib berilmaydi :

```
Who was the inventor of C++?
Your answer: Bjarne Stroustrup
true
In which country was the inventor of C++ born?
Your answer:
```

Bu xolat, merosxo‘rlik ierarxiyasida sinflar aralashmasidagi ob’ektni boshqarmoqchi bo‘lgan kodga o‘ta xos muammodir. Quyi sinf ob’ektlari odatda asosiy sinf ob’ektlaridan kattaroq bo‘lib, turli yordamchi sinf ob’ektlari turli o‘lchamga egadir. Ob’ektlar massivi bu o‘lchamlar xilma xilligi bilan ish ko‘ra olmaydi.

Asosiy va YOrdamchi Sinf ko‘rsatkichlari

Sinf ierarxiyasida, turli sinflardagi ob’ektlardan foydalanish uchun, ko‘rsatgichlarni ishlating. Turli ob’ektlar ko‘rsatgichlarining barchasi yagona o‘lchamga ega —ya’ni, xotira adresi o‘lchami— shunga qaramay, ob’ektlarning o‘zi turli o‘lchamlarga ega bo‘lishlari mumkin.

Bu erda ko‘rsatkichlar massisini o‘rnatish uchun kod berilgan (6-shaklga qarang):

```
Question* quiz[2];
quiz[0] = new Question;
quiz[0]->set_text("Who was the inventor of C++?");
quiz[0]->set_answer("Bjarne Stroustrup");
ChoiceQuestion* cq_pointer = new ChoiceQuestion;
cq_pointer->set_text("In which country was the inventor of C++ born?");
cq_pointer->add_choice("Australia", false);
...
quiz[1] = cq_pointer;
```

Ajratib qo‘yilgan kod ko‘rsatib turganidek, siz massivni ko‘rsatgichlarni saqlash uchungina aniqlaysiz, barcha ob’ektlarni yangisini new chaqirish orqali o‘rnating, va nuqta operatori o‘rniga -> operatorini ishlating.

E’tiborlisi shundaki, oxirgi o‘zlashtirma ChoiceQuestion* tipidagi yordamchi sinf ko‘rsatkichini, asosiy sinf Question* tipidagi ko‘rsatkichga biriktiradi. Bunin uddasidan chiqish mutlaqo oson. Ko‘rsatkich ob’ektning boshlang‘ich adresidir. CHunki har bir ChoiceQuestion "Question" ning maxsus xolati bo‘lib,

ChoiceQuestion ob’ektinig boshlang‘ich adresi, asosan, Question ob’ektining boshlang‘ich

adresidir. Qarama qarshisi—ya’ni asosiy sinf ko‘rsatkichidan yordamchi sinf ko‘rsatkichini o‘zlashtirish —xatodir.

Barcha savollarni taqdim etuvchi kod bu:

```
for (int i = 0; i < QUIZZES; i++)
{
 quiz[i]->display();
 cout << "Your answer: ";
 getline(cin, response);
 cout << quiz[i]->check_answer(response) << endl;
}
```

SHuni ta’kidash kerakki, -> operatorini ishlating, chunkio quiz[i] ko‘rsatkichdir.

Virtual Funksiyalar

Sinf ierarxiyasidgi turli siflarning ob’ektlarini to‘plaganingizda, va komponentlik funksiyasini faollashtirganingizda, to‘g‘ri bo‘lgan komponentlik funksiyasini qo‘llanillishini istaysiz. Masalan, display komponentlik funksiyasini Question* ko‘rsatgichi uchun chaqirganingizda, bu ChoiceQuestionga ishora qilinganda sodir bo‘lib, variantlar namoyish etilishini xoxlaysiz.

Samaradorlik nuqtai nazaridan, bu defolt C++ emas. Defolt :

quiz[i]->display();

chaqirig‘i doimo Question::displayni chaqiradi,chunki quiz[i] turi Question*ga oid.

SHunga qaramay, bu xolatda siz haqiqatdan ham quiz[i]ga ishora qiluvchi ob’ektning muhim turini aniqlashni istaysiz, u yo Question yoki ChoiceQuestion ob’ekti bo‘ladi, keyin esa to‘g‘ri keladigan funksiyani chaqiraerasiz. C++da, siz kompilyatorni, funksiyaning chaqirilishi mos keluvchi funksiya tanlovidan oldin

bo‘lishi kerakligi haqtda ogohlantirishingiz shart, bu esassikldagi har bir iteratsiya uchun turlichadir. Bu maqsadda, virtual yashirin so‘zni ishlating:

```
class Question
{
public:
 Question();
 void set_text(string question_text);
 void set_answer(string correct_response);
 virtual bool check_answer(string response) const;
 virtual void display() const;
private: ...};
```

virtual yashirin so‘z asosiy sinfda ishlatilishi kerak. YOrdamchi sinfda bir nom va turli parametr turlarigi ega bo‘lgan funksiyalarning barchasi keyinchalik avtomatik ravishda virtul xolatga o‘tadi. Biroq, virtual yashirin so‘zni yordamchi sinf funksiyalarida ham qo‘llash yaxshi hisoblanadi.

```
class ChoiceQuestion : public Question
{
public:
 ChoiceQuestion();
 void add_choice(string choice, bool correct);
 virtual void display() const;
private: ...};
```

virtual yashirin so‘zni funksiya tavifida ishlatmang:

```
void Question::display() const // No virtual reserved word
{ cout << text << endl;}
```

Virtual funksiya chaqirilgan har qanday vaqtda, kompilyator dastur bajarilayotgan vaqtda o‘ziga xos chaqiriqning yashirin parametr turini aniqlaydit. So‘ngra ob’ektga mos keluvchi funksiya chaqiriladi. Masalan, display funksiyasi virtual deb e’lon qilinganda, chaqiriq

quiz[i]->display();

doimo ob’ektning aktual turiga tegishli bo‘lgan funksiyani chaqiradi, bunda dis-play[i]—yoki Question::displayga yo bo‘lmasa, ChoiceQuestion::displayga ishora qiladi.

Nazorat savollari

1. C++da funksiya qanday ishlaydi?
2. funksiyaga kutubxona kerakmi?
3. For operatori funksiyada qanday ishlatiladi?
4. Matematik funksiyalar qanday ishlaydi?
5. Funksiya parametrlar nima?
6. Funksiya qanday chaqiriladi?
7. Funksiya parametrlari orqali nima uzatiladi?
8. If operatorining nechta turi bor?
9. O’zgaruvchilar nima uchun qo’llaniladi?

13-Ma’ruza. Polimorfizm

Ma’ruza rejasi:

13.1 Ko‘p shakllilik(polimorfizm);

13.2 Polimorf ob'ekt-telefon yaratish

Kalit so’zlar:*, ro’yxat, manzil, nolinchi ko’rchsatkich, tugun, adres olish &, bo’shatish, ko’rsatkich, virtual destruktor, xotira, xotira chiqishi, destruktor, toifani o’zlashtirish, Inkapsulyasiya, inline, polimorfizm, vorislik, meros sinflar, virtual, MSINF a’zo destruktori.*

13.1 Ko‘p shakllilik(polimorfizm)

Polimorf ob'ekt bu dastur bajarilishi davomida shaklini o’zgartirishi mumkin bo’lgan ob'ektdir. Misol uchun telenfon sinfi kiritilgan bo’lsin:

```
class phone
{
public:
 void dial(char "number) { cout << "Nabor nomera " << number << endl; }
 void answer(void) { cout << "Ojidanie otveta" << endl; }
 void hangup(void) { cout << "Zvonok vihpolnen - povesit' trubku" << endl;
}
 void ring(void) { cout << "Zvonok, zvonok, zvonok" << endl;)
 phone(char *number) { strcpy(phone::number, number); };
private:
 char number[13];
);
```

Kuyidagi PHONEONE.CPP dasturi phone sinfidan foydalanadi:

```
#include <iostream.h>
#include <string.h>
class phone
```

```
{
public:
 void dial(char *number) { cout << "Nabor nomera " << number << endl; }
 void answer(void) { cout << "Ojidanie otveta" << endl; }
 void hangup(void) { cout << "Zvonok vihpolnen - povesit' trubku" << endl;
}
 void ring(void) { cout << "Zvonok, zvonok, zvonok" << endl; }
 phone(char *number) { strcpy(phone::number, number); };
private:
 char number[13];
};
void main(void)
{
 phone telephone("555-1212");
 telephone.dial("212-555-1212");
}
```

Agar tugmali va diskli telefon yaratish lozim bo’lsa va qo’ng’iroq qilish uchun 25 tsent to’lash lozim bo’lsa vorislik yordamida touch_tone va pay_phone schinflarini yaratish mumkin:

```
class touch_tone : phone
{
public:
 void dial(char * number) { cout << "Pik pik Nabor nomera " << number <<
endl; }
 touch_tone(char *number) : phone(number) { }
};
class pay_phone : phone
{
public:
```

```
 void dial(char * number)
 {
 cout << "Pojaluysta, oplatite " << amount << " tsentov" << endl;
 cout << "Nabor nomera " << number << endl;
}
 pay_phone(char *number, int amount) : phone(number) { pay_phone::amount = amount; }
private:
 int amount;
};
```

Yangi touch_tone va pay__phone sinflari hususiy dial usulidan foydalanadi. Qo'yidagi NEWPHONE.CPP dasturida shu sinflardan foydalanilgan

```
#include <iostream.h>
#include <string.h>
class phone
{
public:
 void dial(char *number) { cout << "Nabor nomera " << number << endl; }
 void answer(void) { cout << "Ojidanie otveta" << endl; }
 void hangup(void) { cout << "Zvonok vihpolnen - povesit' trubku" << endl; }
 void ring(void) { cout << "Zvonok, zvonok, zvonok" << endl; }
 phone(char *number) { strcpy(phone::number, number); };
protected:
 char number[13];
);
class touch_tone : phone
{
public:
```

```
 void dial(char *number) { cout << "Pik pik Nabor nomera " << number << endl; }
 touch_tone(char *number) : phone(number) { }
};
class pay_phone : phone
{
public:
 void dial(char * number) { cout << "Pojaluysta, oplatite " << amount << " tsentov" << endl; cout << "Nabor nomera " << number << endl; }
 pay_phone(char * number, int amount) : phone(number) { pay_phone::amount = amount; }
private:
 int amount ;
};
void main (void)
{
 phone rotary("303-555-1212");
 rotary.dial("602-555-1212");
 touch_tone telephone("555-1212");
 telephone.dial("212-555-1212");
 pay_phone city_phone("555-1111", 25);
 city_phone.dial("212-555-1212");
}
```

13.2 Polimorf ob'ekt-telefon yaratish

Dastur bajarilganda ekranga qo’yidagi ma'lumotlar chiqadi:

```
S:\> NEWPHONE <Enter>
Nabor nomera 602-555-1212
```

Pik pik Nabor nomera 212-555-1212

Pojaluysta, oplatite 25 tsentov

Nabor nomera 212-555-1212

Bu misolda polimorf ob'ektlarpdan foydalanilmagan.

POLIMORF OB'EKT-TELEFON YARATISH

Bir qo'ng'iroqdan ikkinchisiga telefon o'z shaklini o'zgartirishi lozim bo'lsin. Polimorf ob'ekt yaratish uchun avval virtual suzi yordamida virtual usullar yaratiladi:

```
class phone
{
public:
 virtual void dial(char •number) { cout << "Nabor nomera " << number << endl; }
 void answer(void) { cout << "Ojidanie otveta" << endl; }
 void hangup(void) { cout << "Zvonok vihpolnen - povesit' trubku" << endl; }
 void ring(void) { cout << "Zvonok, zvonok, zvonok" << endl; }
 phone(char *number) { strcpy(phone::number, number); };
protected:
 char number[13];
};
```

So'ngra dasturda asosiy sinf ob'ektiga ko'rsatkich yaratasiz.

```
phone *poly_phone;
```

Shaklni o'zgartirish uchun bu ko'rsatkiyach qiymatiniga hosilaviy sinf ob'ekti adresiga teng qilinadi:

```
poly_phone = (phone *) &home_phone;
```

Bu misolda (phone *), tiplarni keltirish operatoridir.

Qo'yidagi POLYMORP.CPP dasturda shu usuldan foydalanilgandir:

```
#include <iostream.h>
```

```
#include <string.h>
class phone
{
public:
 virtual void dial(char *number) { cout << "Nabor nomera " << number << endl; }
 void answer(void) { cout << "Ojidanie otveta" << endl; }
 void hangup(void) { cout << "Zvonok vihpolnen - povesit' trubku" << endl; }
 void ring(void) { cout << "Zvonok, zvonok, zvonok" << endl; }
 phone(char *number) { strcpy(phone::number, number); };
protected:
 char number[13] ;
};
class touch_tone : phone
{
public:
 void dial(char * number) { cout << "Pik pik Nabor nomera " << number << endl; }
 touch_tone(char *number) : phone(number) { }
};
class pay_phone: phone
{
public:
 void dial(char *number) { cout << "Pojaluysta, oplatite " << amount << " tsentov" << endl; cout << "Nabor nomera " << number << endl; }
 pay_phone(char *number, int amount) : phone(number) { pay_phone::amount = amount; }
private:
```

```
 int amount;
};
void main(void)
{
 pay_phone city_phone("702-555-1212", 25);
 touch_tone home_phone("555-1212");
 phone rotary("201-555-1212") ;
 // Sdelat' ob'ekt diskovihm telefonom
 phone *poly_phone = &rotary;
 poly_phone->dial("818-555-1212");
 // Zamenit' formu ob'ekta na knopochnihy telefon
 poly_phone = (phone *) &home_phone;
 poly_phone->dial("303-555-1212");
 // Zamenit' formu ob'ekta na platnihy telefon
 poly_phone = (phone *) &city_phone;
 poly_phone->dial("212-555-1212");
}
```

Dastur bajarilishi natijasida ekranga kuyidagi ma'lumotlar hosil buladi:

S:\> POLYMORP <ENTER>

Nabor nomera 818-555-1212

Pik pik Nabor nomera 303-555-1212

Pojaluysta, oplatite 25 tsentov

Nabor nomera 212-555-1212

Nazorat savollari

1. C++da funksiya qanday ishlaydi?
2. funksiyaga kutubxona kerakmi?
3. For operatori funksiyada qanday ishlatiladi?
4. Matematik funksiyalar qanday ishlaydi?

5. Funksiya parametrlar nima?
6. Funksiya qanday chaqiriladi?
7. Funksiya parametrlari orqali nima uzatiladi?
8. If operatorining nechta turi bor?
9. O’zgaruvchilar nima uchun qo’llaniladi?

14-Ma’ruza. Polimorfizm

Ma’ruza rejasi:

14.1 Polimorfizm nima?;

14.2 Ob’yektga yo’naltirilgan dasturlash prinsiplari;

14.3 O’zgaruvchilarning toifasi va aniqlanishi

Kalit so’zlar:*, ro’yxat, manzil, nolinchi ko’rchsatkich, tugun, adres olish &, bo’shatish, ko’rsatkich, virtual destruktor, xotira, xotira chiqishi, destruktor, toifani o’zlashtirish, Inkapsulyasiya, inline, polimorfizm, vorislik, meros sinflar, virtual, MSINF a’zo destruktori.*

14.1. Polimorfizm nima?

Polimorfizm – bu kodning, bajarilish paytidan yuzaga keladigan holatga bog’liq ravishda o’zini turlicha amal qilish xususiyatidir. Polimorfizm – bu faqat ob’yektlar xususiyati bo’lmasdan, balki funksiyalar-a’zolar xususiyatidir va ular xususan, bitta nomdagi funksiya-a’zoni, har xil turdagi argumentlarga ega va bajaridagan amali unga uzatiladigan argumentlar turiga bog’liq bo’lgan funksiyalar uchun (o’rnida) foydalanish imkoniyatida namoyon bo’ladi. Bu holatga *funksiyalarni qayta yuklash* deyiladi. Polimorfizm amallarga ham qo’llanishi mumkin, ya’ni amal mazmuni (natijasi) operand (berilgan) turiga bog’liq bo’ladi. Polimorfizmning bunday turiga *amallarni qayta yuklash* deyiladi.

Polimorfizmning yana bir ta’rifi quyidagicha: polimorfizm – bu tayanch sinfga ko’rsatgichlarning (murojaatlarning), ularni virtual funksiyalarni chaqirishdagi turli shakl (qiymatlarni) qabul qilish imkoniyatidir. C++ tilining bunday imkoniyati *kechiktirilgan bog’lanish* natijasidir. Kechiktirilgan bog’lanishda chaqiriladigan funksiya-a’zolar adreslari dastur bajarilishi jarayonida dinamik ravishda aniqlanadi.

14.2. Ob’yektga yo’naltirilgan dasturlash prinsiplari;

An’anaviy dasturlash tillarida esa bu adreslar statik bo’lib, ular kompilyasiya paytida aniqlanadi (*oldindan bog’lanish*). Kechiktirilgan bog’lanish faqat virtual funksiyalar uchun o’rinli.

C++ tili ob’yektga yo’naltirilgan dasturlash prinsiplarini qo’llab quvvatlaydi. Bu prinsiplar quyidagilardan iborat:

1. Inkapsulyasiya
2. Vorislik
3. Polimorfizm

Polimorfizm. Poli – ko’p, morfe – shakl degan ma’noni bildiradi. C++ tili bir xil nomdagi funksiya turli ob’yektlar tomonidan ishlatilganda turli amallarni bajarish imkoniyatini ta’minlaydi. Polimorfizm – shaklning ko’p xilligidir.

14.3. O’zgaruvchilarning toifasi va aniqlanishi

Dasturda ishlatiladigan har bir o’zgaruvchi o’z toifasiga ega va u quyidagilarni aniqlaydi:

1. Xotiradagi o’lchovini;
2. Unda saqlanayotgan ma’lumotlarni;
3. Uning yordamida bajarilishi mumkin bulgan amallarni.

C++ tilida dasturchi o’ziga kerakli ixtiyoriy toifani hosil qilishi mumkin. Bu yangi toifa ichki toifalarning xossalari va ularning funksional imkoniyatlarini o’zida ifodalaydi. Yangi toifa sinfni e’lon qilish orqali tuziladi. Sinf bu – bir-biri bilan funksional bog’langan o’zgaruvchilar va usullar (funksiyalar) to’plamidir.

Nazorat savollari:

1. Polimorfizm nima?
2. Ob’yektga yo’naltirilgan dasturlash prinsiplari nimalarni qo’lllab quvvotlaydi?
3. Sinfdagi o’zgaruvchilar – sinf a’zolarining qo’llanishi.

15-Ma’ruza. Inkapsulyatsiya

Ma’ruza rejasi:

15.1. Inkapsulyatsiya tushunchasi.

15.2. Ob’yektga yo’naltirilgan dasturlash prinsiplari.

15.3. Sinflar hosil qilish va inkapsulyatsiyaning ishlatilishi.

Kalit so’zlar:*, ro’yxat, manzil, nolinchi ko’rchsatkich, tugun, adres olish &, bo’shatish, ko’rsatkich, virtual destruktor, xotira, xotira chiqishi, destruktor, toifani o’zlashtirish, Inkapsulyasiya, inline, polimorfizm, vorislik, meros sinflar, virtual, MSINF a’zo destruktori.*

15.1. Inkapsulyatsiya tushunchasi

Inkapsulyasiya – bu berilganlar va ularni qayta ishlovchi kodni birlashtirish mexanizmidir. Inkapsulyasiya berilganlar va kodni tashqi ta’sirdan saqlash imkonini beradi.

Yuqoridagi ikkita konsepsiyani amalga oshirish uchun C++ tilida *sinflar* ishlatiladi. *Sinf* termini bilan Ob’yektlar turi aniqlanadi. Sinfning har bir vakili (nusxasi) *Ob’yekt* deb nomlanadi. Har bir Ob’yekt o’zining alohida holatiga ega bo’ladi. Ob’yekt holati uning *berilganlar-a’zolarning* ayni paytdagi qiymati bilan aniqlanadi. Sinf vazifasi uning *funksiya-a’zolarining* sinf ob’yektlari ustida bajaradigan amallar imkoniyati bilan aniqlanadi.

Berilgan sinf ob’yektini yaratish *konstruktor* deb nomlanuvchi maxsus funksiya-a’zo tomonidan, o’chirish esa destruktor deb nomlanuvchi maxsus funksiya–a’zo orqali amalga oshiriladi.

Sinf ichki berilganlarini murojaatni cheklab qo’yishi mumkin. Cheklov berilganlarni ochiq (public), yopiq (private) va himoyalangan (protected) deb aniqlash bilan tayinlanadi.

Sinf, shu turdagi ob’yektning tashqi dunyo bilan o’zaro bog’lanishi uchun qat’iy muloqot shartlarini aniqlaydi. Yopiq berilganlarga yoki kodga faqat shu ob’yekt ichida murojaat qilish mumkin. Boshqa tomondan, ochiq berilganlarga va kodlarga, garchi ular ob’yekt ichida aniqlangan bo’lsa ham, dasturning ixtiyoriy joyidan murojaat qilish mumkin va ular ob’yektni tashqi olam bilan muloqotni yaratishga xizmat qiladi. Yaratilgan ob’yektlarni, ularni funksiya–a’zolariga oddiygina murojaat orqali amalga oshiriluvchi *xabarlar* (yoki *so’rovlar*) yordamida boshqarish mumkin. Keyinchalik Windows xabarlari bilan adashtirmaslik uchun so’rov termini ishlatiladi.

C++ tili ob’yektga yo’naltirilgan dasturlash prinsiplarini qo’llab quvvatlaydi. Bu prinsiplar quyidagilardan iborat:

- Inkapsulyasiya
- Vorislik
- Polimorfizm

Inkapsulyasiya. Agar muxandis ishlab chiqarishda diod, triod yoki rezistorni ishlatsa, u bu elementlarni yangitdan ihtiro qilmaydi, balki do’kondan sotib oladi. Demak, muxandis ularning qanday tuzilganligiga e’tiborini qaratmaydi, bu elementlar yaxshi ishlasa yetarli. Aynan shu tashqi konstruksiyada ishlaydigan yashirinlik yoki avtonomlik xossasi inkapsulyasiya deyiladi.

Sinfdagi o’zgaruvchilar – sinf a’zolari yoki sinf xossalari deyiladi.

Sinfdagi funksiyalar odatda o’zgaruvchilar ustida biror bir amal bajaradi. Ularni sinf usullari (metodlari) deb ham ataladi.

Sinfni e’lon qilish uchun class so’zi , { } ichida esa shu sinfning a’zolari va usullari keltiriladi. Masalan:

```
class non
{ int ogirlik ;
  int baho ;
  void yasash ( );
```

```
  void yopish ( );
  void eyish ( );
}
```

Sinfni e’lon qilishda xotira ajratilmaydi. Sinf e’lon qilinganda kompilyator faqat shunday (non) sinf borligini, hamda unda qanday qiymatlar (ogirlik, baho) saqlanishi mumkinligini, ular yordamida qanday amallarni (yasash, yopish, yeyish) bajarish mumkinligi haqida xabar beradi. Bu sinf ob’yekti hammasi bo’lib 4 bayt joy egallaydi (2 ta int).

Ob’yekt sinfning biror bir nusxasi hisoblanadi.

15.2. Ob’yektga yo’naltirilgan dasturlash prinsiplari

C++ tilida toifalarga qiymat o’zlashtirilmaydi, balki o’zgaruvchiga o’zlashtiriladi. Shuning uchun to’g’ridan-to’g’ri int = 55 deb yozib bo’lmaganidek non.baho=1200 deb ham bo’lmaydi. O’zlashtirishda xatolikka yo’l qo’ymaslik uchun oldin non sinfiga tegishli patir ob’yektini hosil qilamiz keyin esa unga kerakli qiymatlarni beramiz.

Masalan:

```
int a;     // butun toifali a o’zgaruvchisi, ob’yekti
non patir; //
```

Endi non sinfining real ob’yekti aniqlanganidan so’ng uning a’zolariga murojaat

```
patir.baho = 1200;
patir.ogirlik = 500;
patir.yasash ( ) ;
```

Sinfni e’lon qilishda quyidagilardan foydalaniladi:

public - ochiq

private – yopiq

15.3. Sinflar hosil qilish va inkapsulyatsiyaning ishlatilishi

Sinfning barcha usul va a’zolari boshlang’ich holda avtomatik ravishda yopiq bo’ladi. Yopiq a’zolarga esa faqat shu sinfning usullari orqaligina murojaat qilish mumkin. Ob’yektning ochiq a’zolariga esa dasturdagi barcha funksiyalar murojaat qilishi mumkin. Lekin sinf a’zolariga murojaat qilish ancha mushkul ish hisoblanadi. Agar to’g’ridan to’g’ri:

```
non patir;
patir.baho = 1200;
patir.og`irlik = 500;  deb yozsak xato bo’ladi.
```

A’zolarga murojaat qilishdan oldin uni ochiq deb e’lon qilish kerak:

```
# include < iostream.h >
class non
{ public :
  int baho;
  int ogirlik;
  void yasash ( );  };
void main ( )
{ non patir;
patir.baho = 1200;  patir.ogirlik = 500;
cout <<“men olgan patir” <<patir.baho <<“so’m”<<endl;
cout <<“uning og’irligi =”<<patir.og’irlik <<endl ;   }
```

Nazorat savollari

1. Inkapsulyasiya nima?
2. Inkapsulyasiya qanday ishlatiladi?
3. Ob’yektga yo’naltirilgan dasturlash prinsiplar qaysilar?

16-Ma’ruza. Virtual va abstract funksiyalar

Ma’ruza rejasi:

16.1. Virtual funksiyalar.

16.2. Abstract funksiyalar.

Kalit so’zlar:*, ro’yxat, manzil, nolinchi ko’rchsatkich, tugun, adres olish &, bo’shatish, ko’rsatkich, virtual destruktor, xotira, xotira chiqishi, destruktor, toifani o’zlashtirish, Inkapsulyasiya, inline, polimorfizm, vorislik, meros sinflar, virtual funksiyalar, MSINF a’zo destruktori.*

16.1. Virtual funksiyalar

Синф иерархиясидги турли сифларнинг объектларини тўплаганингизда, ва компонентлик функциясини фаоллаштирганингизда, тўғри бўлган компонентлик функциясини қўлланиллишини истайсиз. Масалан, display компонентлик функциясини Question* кўрсатгичи учун чақирганингизда, бу ChoiceQuestionга ишора қилинганда содир бўлиб, вариантлар намойиш этилишини хохлайсиз.

Самарадорлик нуқтаи назаридан, бу дефолт C++ эмас. Дефолт :

quiz[i]->display();

чақириғи доимо Question::displayни чақиради,чунки quiz[i] тури Question*га оид.

Шунга қарамай, бу холатда сиз ҳақиқатдан ҳам quiz[i]га ишора қилувчи объектнинг муҳим турини аниқлашни истайсиз, у ё Question ёки ChoiceQuestion объекти бўлади, кейин эса тўғри келадиган функцияни чақираерасиз. C++да, сиз компиляторни, функциянинг чақирилиши мос келувчи функция танловидан олдин бўлиши кераклиги ҳақтда огоҳлантиришингиз шарт, бу эса циклдаги ҳар бир итерация учун турличадир. Бу мақсадда, virtual яширин сўзни ишлатинг:

```
class Question
{
public:
 Question();
 void set_text(string question_text);
 void set_answer(string correct_response);
 virtual bool check_answer(string response) const;
 virtual void display() const;
private:
 ...
};
```

virtual яширин сўз *асосий синфда* ишлатилиши керак. Ёрдамчи синфда бир ном ва турли параметр турлариги эга бўлган функцияларнинг барчаси кейинчалик автоматик равишда виртул холатга ўтади. Бироқ, virtual яширин сўзни ёрдамчи синф функцияларида ҳам қўллаш яхши ҳисобланади.

```
class ChoiceQuestion : public Question
{
public:
 ChoiceQuestion();
 void add_choice(string choice, bool correct);
 virtual void display() const;
private:
 ...
};
```

virtual яширин сўзни функция тавифида ишлатманг:

```
void Question::display() const // No virtual reserved word
{
 cout << text << endl;
```

}

Виртуал функция чақирилган ҳар қандай вақтда, компилятор дастур бажарилаётган вақтда ўзига хос чақириқнинг яширин параметр турини аниқлайдит. Сўнгра объектга мос келувчи функция чақирилади. Масалан, display функцияси виртуал деб эълон қилинганда, чақириқ

quiz[i]->display();

доимо объектнинг актуал турига тегишли бўлган функцияни чақиради, бунда dis-play[i]—ёки Question::displayга ё бўлмаса, ChoiceQuestion::displayга ишора қилади.

quiz массиви саволларнинг иккиси аралашмасини тўплайди. Бундай тўплам **poly-morphic** деб номланади (яъни, “бир қанча шакллар” дегани). Полиморфик тўпламдаги объектлар бир оз ўхшашликка эга бўлсаларда, бир турга мансуб бўлишлари шарт эмас. Меросхўрлик ушбу ўхшашликни ифодалаш учун ишлатилади,ва виртуал функциялар ҳаракатдаги вариацияларга имкон беради.

Виртуал функциялар дастурларга ўта қайишқоқлик хусусиятини берадилар. Савол тақдимоти цикли фақатгина умумий механизмни тасвирлайди: “Саволни намойиш этинг, жавоб олинг, ва уни текширинг”. Ҳар бир объект, махсус вазифаларни қандай бажаришни ўзи билади: “Саволни намойиш этинг” ва “Жавобни текширинг”.

Виртуал функциялардан фойдаланиш дастурларни *кенгайтиришга онсонлик* яратади. Тасаввур қилинг, бизга ҳисоб китоб учун тахминий жавоб олишимиз мумкин бўлган янги савол тури керак. Қилишимиз керак бўлган нарса бу- ўзиниг check_answer функциясига эга бўлган NumericQuestion янги синфини ўрнатишдир. Сўнгра quiz массивини аниқ саволлар, вариантли саволларва рақамли саволлар аралашмаси билан тўлдиришимиз мумкин. Саволларни намойиш қилувчи код умуман ўзгартирилмаслиги керак! Виртуал функциялардаги чақириқлар автоматик равишда янги ўрнатилган синфлардан тўғри компонентлик функцияларини танлаб олади.

16.2. Abstract funksiyalar

Бу ерда тест дастурининг охирги версияси берилган бўлиб, унда кўрсатгичлар ва виртуал функциялар ишлатилган. Дастурни ишлатганингизда, виртуал функцияларга мос келувчи версиялар чақирилганлигини кўрасиз. (question.cpp ва choicequestion.cpp файллари китобингизнинг иккинчи қисмидаги кодларга киритилган.)

ch10/quiz3/question.h

```
1 #ifndef QUESTION_H
2 #define QUESTION_H
3
4 #include <string>
5
6 using namespace std;
7
8 class Question
9 {
10 public:
11 /**
12 Constructs a question with empty question and answer.
13 */
14 Question();
16 /**
17 @param question_text the text of this question
18 */
19 void set_text(string question_text);
20
21 /**
```

```
22 @param correct_response the answer for this question 23 */
24 void set_answer(string correct_response);
25
26 /**
27 @param response the response to check
28 @return true if the response was correct, false otherwise 29 */
30 virtual bool check_answer(string response) const;
31
32 /**
33 Displays this question.
34 */
35 virtual void display() const;
36 private:
37 string text;
38 string answer;
39 };
40
41 #endif
ch10/quiz3/choicequestion.h 1 #ifndef CHOICEQUESTION_H
2 #define CHOICEQUESTION_H
3
4 #include <vector>
5 #include "question.h"
6
7 class ChoiceQuestion : public Question
8 {
9 public:
```

```
10 /**
11 Constructs a choice question with no choices.
12 */
13 ChoiceQuestion();
14
15 /**
16 Adds an answer choice to this question.
17 @param choice the choice to add
18 @param correct true if this is the correct
choice, false otherwise 19 */
20 void add_choice(string choice, bool correct);
21
22 virtual void display() const;
23 private:
24 vector<string> choices;
25 };
26
27 #endif
```

ch10/quiz3/test.cpp

```
1 #include <iostream>
2 #include "question.h"
3 #include "choicequestion.h"
4
5 int main()
6 {
7 string response;
8 cout << boolalpha;
9
10 // Make a quiz with two questions
11 const int QUIZZES = 2;
```

```
12 Question* quiz[QUIZZES];
13 quiz[0] = new Question;
14 quiz[0]->set_text("Who was the inventor of C++?");
15 quiz[0]->set_answer("Bjarne Stroustrup");
16
17 ChoiceQuestion* cq_pointer = new ChoiceQuestion;
18 cq_pointer->set_text(
19 "In which country was the inventor of C++ born?");
20 cq_pointer->add_choice("Australia", false);
21 cq_pointer->add_choice("Denmark", true);
22 cq_pointer->add_choice("Korea", false);
23 cq_pointer->add_choice("United States", false);
24 quiz[1] = cq_pointer;
25
26 // Check answers for all questions
27 for (int i = 0; i < QUIZZES; i++)
28 {
29 quiz[i]->display();
30 cout << "Your answer: ";
31 getline(cin, response);
32 cout << quiz[i]->check_answer(response) << endl; 33
}
34
35 return 0;
36 }
```

Nazorat savollari:

1. Inkapsulyasiya nima?
2. Inkapsulyasiya qanday ishlatiladi?
3. Ob’yektga yo’naltirilgan dasturlash prinsiplar qaysilar?

4. Virtual funksiyalar qanday ishlaydi?
5. Virtual funksiyani qolganlaridan farqini sanang.

17-Ma’ruza. Operator va funksiyalarni qayta yuklash

Ma’ruza rejasi:

17.1 Funksiyalarni qayta yuklash;

17.2 Operatorlarni qayta yuklashda qo’llanilishi.

Kalit so’zlar:*, ro’yxat, manzil, nolinchi ko’rchsatkich, tugun, adres olish &, bo’shatish, ko’rsatkich, virtual destruktor, xotira, xotira chiqishi, destruktor, toifani o’zlashtirish, Inkapsulyasiya, inline, polimorfizm, vorislik, meros sinflar, virtual funksiyalar, MSINF a’zo destruktori.*

17.1. Funksiyalarni qayta yuklash

C++ 3-darslik FUNKSIYALAR C++ da dasturlashning asosiy bloklaridan biri funksiya-lardir. Funksiyalarning foydasi shundaki, katta masala bir necha kichik bo'laklarga bo'linib, har biriga alohida funksiya yozilganda, masala yechish algoritmi ancha soddalashadi. Bunda dasturchi yozgan funksiyalar C ++ ning standart kutubhonasi va boshqa firmalar yozgan kutub- honalar ichidagi funksiyalar bilan birlashtiriladi. Bu esa ishni osonlashtiradi. Ko'p holda dasturda takroran bejariladigan amalni funksiya sifatida yozish va kerakli joyda ushbu funksiyani chaqirish mumkin. Funksiyani programma tanasida ishlatish uchun u chaqiriladi, yani uning ismi yoziladi va unga kerakli argumentlar beriladi. () qavslar ushbu funksiya chaqirig'ini ifodalaydi. Masalan: foo(); k = square(l); Demak, agar funksiya argumentlar olsa, ular () qavs ichida yoziladi. Argumentsiz funksiyadan keyin esa () qavslarning o'zi qo'yiladi. MA'LUMOTLAR TIPI (DATA TYPES) Shu paytgacha ma'lumotlar tipi deganda butun son va kasrli son bor deb kelgan edik. Lekin bu bo'limda maylumotlar tipi tushunchasini yahshiroq ko'rib chiqish kerak bo'ladi. Chunki funksiyalar bilan ishlagandaargument kiritish va qiymat qaytarishga to'g'ri keladi.

Agar boshidan boshlaydigan bo'lsak, kompyterda hamma turdagi ma'lumotlar 0 va 1 yordamida kodlanadi. Buning sababi shuki, elektr uskunalar uchun ikki holat tabiyi-dir, tok oqimi bor yoki yo'q, kondensatorda zaryad bor yoki yo'q va hakozo. Demak biz bu holatlarni oladigan jihozlarni bir quti deb faraz qilsak, quti ichida yo narsa bo'ladi, yo narsa bo'lmaydi. Mantiqan buni biz bir yoki nol deb belgilaymiz. Bu kabi faqat ikki holatga ega bo'lishi mumkin bo'lgan maylumot birligiga biz BIT deymiz. Bu birlik kichik bo'lgani uchun kompyuterda bitlar guruhi qo'llaniladi. Bittan keyingi birlik bu BAYT (byte). Baytni sakkizta bit tashkil etadi. Demak bir bayt yordamida biz 256 ta holatni kodlashimiz mumkin bo'ladi. 256 soni ikkining sakkizinchi darajasiga tengdir. Bitimiz ikki holatga ega bo'lgani uchun biz kompyuterni ikkili arifmetikaga asoslangan deymiz. Ammo agar kerak bo'lsa, boshqa sistemaga asoslangan mashinalarni ham qo'llash mumkin. Masalan uchli sanoq sistemasiga asoslangan kompyuterlar bor. Informatika faniga ko'ra esa, hisoblash mashinasi uchun eng optimal sanoq sistemasi e ga teng bo'lar ekan. Demak amaldagi sistemalar ham shu songa iloji borisha yaqin bo'lishi kerakdir. C/C++ da baytga asoslangan tip char dir. char tipi butun son tipida bo'lib, chegaraviy qiymatlari -128 dan +127 gachadir. Bu tip lotin alifbosi harflarini va y ana qo'shimcha bir guruh simvollarni kodlashga qulay bo'lgan. Lekin hozirda milliy alifbelarni kodlash uchun 16 bitlik UNICODE qo'llanilmoqda. U yordamida 65536 ta simvolni ko'rsatish mumkin. char tipida o'zgaruvchi e'lon qilish uchun dasturda char g, h = 3, s; kabi yozish kerak. O'zgaruvchilar vergul bilan ayriladi. E'lon bilan bir vaqtning o'zida boshlang'ich qiymat ham berish imkoni bor. Mashina ichida baytdan tashkil topgan boshqa kattaliklar ham bor. Ikki baytdan tuzilgan kattalik so'z (word) deyiladi, unda 16 bit bo'ladi. 4 ta bayt guruhi esa ikkili so'z (double word) bo'ladi. Bu birlik 32 bitli mashimalarda qo'llaniladi. Hozirda qo'llanilmoqda bo'lgan mashinalar asosan 32 bitlidir, masalan Pentium I/II/III sistemalari. C ++ da butun sonlarning ikki tipi bor.

17.2. Operatorlarni qayta yuklashda qo’llanilishi.

Biri char - uni ko'rib chiqdik. Ikkinchisi int dir. Mashinalarning arhitekturasi qanday kattalikda bo'lsa, int tipining ham kattakigi huddi shunday bo'ladi. 16 bitlik mashinalarda int 16 bit edi. Hozirda esa int ning uzunligi 32 bitdir. int (integer - butun son) tipi charga o'hshaydi. Farqi bir baytdan kattaligidadir. 16 bitli int ning sig'imi -32768 dan +32767 gachadir. 32 bitli int esa -2 147 483 648 dan +2 147 483 647 gacha o'rin egallaydi. Bu ikki butun son tipidan tashqari C++ da ikki tur vergulli, (nuqtali) yani haqiqiy son tipi mavjud. Bulardan biri float, hotirada 4 bayt joy egallaydi.

Ikkinchisi esa double, 8 bayt kattalikka ega. Bularning harakteristikalari quyidagi jadvalda berilgan. Ushbu tiplar bilan ishlaganda unsigned(ishorasiz, +/- siz), signed (ishorali) long (uzun) va short (qisqa) sifatlarini qo'llasa bo'ladi. unsigned va signed ni faqat butun son tiplari bilan qo'llasa bo'ladi. unsigned

Nazorat savollari:

1. Inkapsulyasiya nima?
2. Inkapsulyasiya qanday ishlatiladi?
3. Ob’yektga yo’naltirilgan dasturlash prinsiplar qaysilar?
4. Virtual funksiyalar qanday ishlaydi?
5. Virtual funksiyani qolganlaridan farqini sanang.

18-Ma’ruza. O’yinlar yaratish va foydalanish

Ma’ruza rejasi:

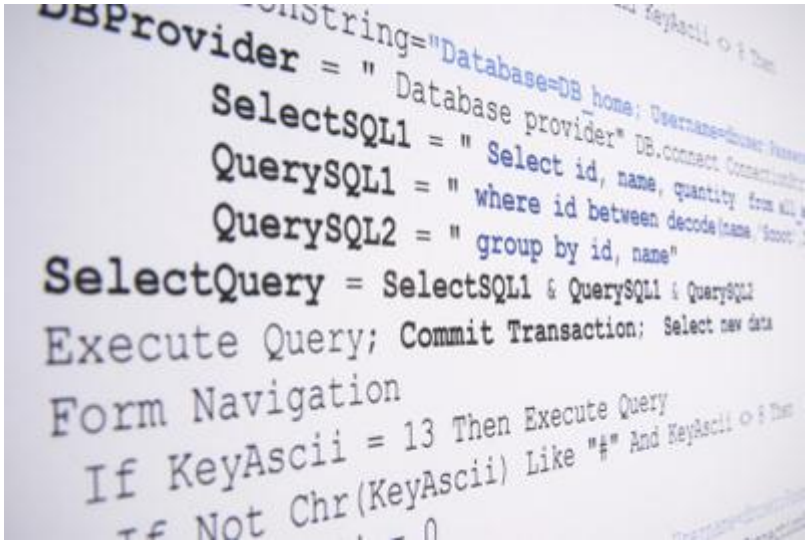
18.1. O’yinlar yaratish texnologiyasi.

18.2. O’yinlar yaratish usullari.

18.3. O’yinlada dasturlashning imkoniyati.

Kalit so’zlar:*, ro’yxat, manzil, nolinchi ko’rchsatkich, tugun, adres olish &, bo’shatish, ko’rsatkich, virtual destruktor, xotira, xotira chiqishi, destruktor, toifani o’zlashtirish, Inkapsulyasiya, inline, polimorfizm, vorislik, meros sinflar, virtual funksiyalar,o’yin direktivalari, o’yin texnologiyalari, MSINF a’zo destruktori.*

18.1 O’yinlar yaratish texnologiyasi



Dasturlash – kompyuterlar va boshqa mikroprosersorli elektron mashinalar uchun dasturlar tuzish, sinash va o‘zgartirish jarayonidan iborat.

Odatda dasturlash yuqori saviyali dasturlash tillari (Delphi, Java, C++, Python) vositasida amalga oshiriladi. Bu dasturlash tillarining semantikasi odam tiliga yaqinligi tufayli dastur tuzish jarayoni ancha oson kechadi.

Biz sizga xozirgi kunda keng tarqalgan desktop dasturlashda ishlatiladigan dasturlash tillaridan bazilari xaqida aytib o'tamiz:

Delphi (talaff. délfi) — dasturlash tillaridan biri. Borland firmasi tomonidan ishlab chiqarilgan. Delphi dasturlash tili ishlatiladi va avvaldan Borland Delphi paketi tarkibiga kiritilgan. Shu bilan bir qatorda 2003-yildan hozirgacha qo‘llanilayotgan shu nomga ega bulgan. Object Pascal — Pascal tilidan bir qancha kengaytirishlar va to‘ldirishlar orqali kelib chiqqan bo‘lib, u ob’yektga yo‘naltirilgan dasturlash tili hisoblanadi.

Avvaldan ushbu dasturlash muhiti faqatgina Microsoft Windows amaliyot tizimi uchun dasturlar yaratishga mo'ljallangan, keyinchalik esa GNU/Linux hamda Kylix tizimlari uchun moslashtirildi, lekin 2002-yilgi Kylix 3 sonidan so'ng ishlab chiqarish to'xtatildi, ko'p o'tmay esa Microsoft.NET tizimini qo'llab quvvatlashi to'g'risida e'lon qilindi.

Lazarus proekti amaliyotidagi (Free Pascal) dasturlash tili Delphi dasturlash muhitida GNU/Linux, Mac OS X va Windows CE platformalari uchun dasturlar yaratishga imkoniyat beradi.

Delphida bajarilgan taniqli dasturlar ichidan quyidagilarni sanash mumkin:

Borland mahsulotlari: Borland Delphi, Borland C++ Builder, Borland JBuilder 1 va 2-sonlari

Administratorlik/ma'lumotlar bazasi yaratish: MySQL Tools (Administrator, Query Browser), TOAD Muhandislik dasturiy mahsulotlari: Altium Designer/Protel (Elektronika pyektlash) Grafik munitlari: FastStone Image Viewer, FuturixImager, Photofiltre

Internetda ma'lumotlar uzatish: Skype (VoIP va IM), QIP va QIP Infium, The Bat! (elektron pochta mijoz dasturi), PopTray (elektron pochtani tekshirish), FeedDemon (RSS/Atom yangiliklar lentasini ko'rish), XanaNews (Yangiliklar guruhini o'qish), Xnews (Yangiliklar guruhini o'qish)

Musiqa yaratish: FL Studio (avval FruityLoops)

Dasturlash muhitlari yaratish: Dev-C++ (IDE), DUnit, Game Maker (o'yinlar yaratish) Help & Manual, Inno Setup (Dastur o'rnatuvchi dastur)

Veb-dasturlash: Macromedia HomeSite (HTML-taxrirlagich), TopStyle Pro (CSS-taxrirlagich), Macromedia Captivate

Veb-brauzerlar (MSIE uchun asos): Avant Browser, Netcaptor

Yordamchi dasturlar(utilitalar): Spybot — Search & Destroy, Ad-Aware, jv16 PowerTools, Total Commander, Copernic Desktop Search, PowerArchiver, MCubix, Download Master [ko'chirishlar ustasi]

Visual Basic (talaffuzi: "Vijual Beysik") – Microsoft korporatsiydan dasturlash tili va uning uchun dasturlash muhitdir. U BASICdan ko`p tushunchalar oldi va tez rasmli interfeys bilan dasturlar taraqqiyot ta`minlaydi.

Oxirgi versiya 6.0 1998 yilda reliz kelishdi. Microsoftdan voris Visual Basic .NET 2002 yilda paydo bo`ldi.

Java dasturlash tili - eng yaxshi dasturlash tillaridan biri bo'lib unda korporativ darajadagi mahsulotlarni(dasturlarni) yaratish mumkin.Bu dasturlash tili Oak dasturlash tili asosida paydo bo'ldi. Oak dasturlash tili 90-yillarning boshida Sun Microsystems tomonidan platformaga(Operatsion tizimga) bog'liq bo'lmagan holda ishlovchi yangi avlod aqlli qurilmalarini yaratishni maqsad qilib harakat boshlagan edi. Bunga erishish uchun Sun hodimlari C++ ni ishlatishni rejalashtirdilar, lekin ba'zi sabablarga ko'ra bu fikridan voz kechishdi.Oak muvofaqiyatsiz chiqdi va 1995-yilda Sun uning nomini Java ga almashtirdi, va uni WWW rivojlanishiga hizmat qilishi uchun ma'lum o'zgarishlar qilishdi.

Java Obyektga Yo'naltirilgan Dasturlash(OOP-object oriented programming) tili va u C++ ga ancha o'xshash.Eng ko'p yo'l qo'yildigan xatolarga sabab bo'luvchi qismalari olib tashlanib, Java dasturlash tili ancha soddalashtirildi.

Java kod yozilgan fayllar(*.java bilan nihoyalanuvchi) kompilatsiyadan keyin bayt kod(bytecode) ga o'tadi va bu bayt kod interpretator tomonidan o'qib yurgizdiriladi.

O’yinlar yaratish usullari;

C++ (*talaffuzi: si plyus plyus*) — turli maqsadlar uchun mo‘ljallangan dasturlash tili. 1979-yili Bell Labsda Biyarne Stroustrup tomonidan C dasturlash tilining imkoniyatlarini kengaytirish va OOP(object Oriented Programming) xususiyatini kiritish maqsadida ishlab chiqarilgan. Boshida „C with Classes“ deb atalgan, 1983-yili hozirgi nom bilan ya’ni C++ deb o‘zgartirilgan. C++ C da yozilgan dasturlarni kompilyatsiya qila oladi, ammo C kompilyatori bu xususiyatga ega emas. C++ tili operatsiyon tizimlarga aloqador qisimlarni, klient-server dasturlarni, EHM o‘yinlarini, kundalik ehtiyojda qo‘llaniladigan dasturlarni va shu kabi turli maqsadlarda ishlatiladigan dasturlarni ishlab chiqarishda qo‘llaniladi.

wikipedia.org malumotlaridan foydalanildi

Ushbu maqolaning 1-qismida kompilyasiya jarayoni va undagi birinchi bosqich – kiritilgan matnni bo’laklash haqida gapirib o’tgan edim. Ushbu maqolada mavzuni davom ettiramiz.

Oldin ham aytilganidek har qanday (dasturlash) til o’zining alifbosiga va qoidalar to’plamiga egadir. So’zlarning qanday tartibda kelishi tilning grammatikasini belgilaydi. Avvalgi maqolada erkin-kontekstli dasturlash tillarini grammatikasini yozish uchun foydalaniladigan Bekus-Naur yozuvi haqida ham gapirgandim. Kompilyasiya jarayonining keyingi bosqichi kiritilgan dastur matnidan, ya’ni biz o’tgan safar bo’laklarga ajratib olgan qismlardan til grammatikasiga mos ravishda tuzilmaviy daraxt hosil qilishidan iborat bo’ladi. Ushbu jarayonni sintaktik daraxt hosil qilish yoki parsing ham deyiladi. Sintaktik daraxtda dasturdagi dasturlash tilining harbir so’zi va buyruqlar ketma-ketligi shoxchalar yordamida ifodalanadi.

Masalan:

2 + 3 * 5

ifodani sintaktik daraxti quyidagicha ko’rinishda bo’lishi mumkin:

Yoki murakkabroq daraxt tahminan mana bunday bo’ladi:

Parsing tushunchasi umumiy ma’nodagi so’z bo’lib, sodda qilib aytganda har qanday matnli ahborotni tahlil qilish, qayta ishlash deb ham tushunish mumkin.

Sintaktik daraxt hosil qilishda qo’llaniladigan parsing jarayoni turlari bir necha hil bo’lib ulardan eng ko’p ishlatiladigani – Yuqoridan pastga qarab ishlovchi va Pastdan yuqoriga qarab ishlovchi parsing turlaridir (top-down va bottom-up). Bularning bir biridan farqi sintaktik daraxtning qay tomondan boshlab yaratishidadir. Bular haqida batafsil gapirib o’tirmayman, bularni bitta maqolada tushuntirishni iloji ham yo’q. Biz faqatgina Yuqoridan pastga qarab ishlovchi turga kiruvchi **Rekursiv parserni** ko’rib chiqamiz. Qolgan parser turlari haqida balki keyingi maqolalarda ham yozishga harakat qilib ko’raman.

Rekursiv tushuvchi parserning qolgan parserlardan asosiy afzalligi uni bevosita dasturlash mumkin. Qolgan parser turlarida esa steklar, mos kelish jadvallardan foydalaniladi.

Rekursiv parserni tuzishni boshlaymiz.

Deylik, bizning tilimizda faqatgina bitta qoida mavjud:

<Son> := ButunSon | HaqiqiySon

Ya’ni, dastur kodi faqat bitta butun yoki haqiqiy sondan iborat bo’lishi mumkin bo’lsin. “|” belgisi “yoki” degan ma’noni anglatadi (BN yozuvi). Grammatikada har doim boshlang’ich qoidani ham ko’rsatish kerak bo’ladi. Ushbu boshlang’ich qoida dastur qaysi qoidadan boshlanishini bildiradi. Bizda bitta qoida bor (yuqoridagi) va boshlang’ich qoida ham shunga teng deb olamiz:

<Dastur boshi> := <Son>

Bizda tayyor bo’laklar mavjud, shulardan foydalanib parserimizni yozamiz. Avvalo Parser nomli class yaratamiz.

```
class Parser
{

    string text; // kiritilgan matn
    Lexer lex; // ajratilgan bo’laklar, Lexer

    public Parser (string _text)
```

```
        {

            this.text = _text;

        }

        public void Parse ()
        {

            lex = new Lexer(text);

            StartProgram(); // boshlang'ich qoida

        }

    }
```

O'yinlada dasturlashning imkoniyati

Dastur matni faqatgina 1 ta sondan tashkil topishi mumkinligidan birinchi bo'lak sonligini va undan boshqa bo'lak yo'qligini tekshirsak parserimiz tayyor))

```
    // <Son> qoidasi
    // Joriy bo'lakni butun yoki haqiqiy sonligini tekshiradi

    public bool NumValue()
    {

      // agar keyingi bo'lak son bo'lsa

      if (lex.LookNextToken().Type == Token.TokenType.NUMLITERAL)
      {
```

```
        lex.ReadToken();

        return true;

    } else

        return false;

}

// <Dastur boshi>

public void StartProgram()
{

    // NumValue() ijobiy qiymat qaytarsa va keyingi bo'lak dastur
    // oxiri belgisi bo'lsa unda hammasi OK

    if (NumValue() && lex.ReadToken().Type==Token.TokenType.EOP )

        Console.WriteLine ("Parsing OK.");

    else

        Console.WriteLine ("Syntax Error!");

}
```

Ko’rib turganingizdek oddiy qoidamizga juda oddiy parser hosil bo’ldi. Endi til qoidalarini (mos ravishda parserni ham) biroz murakkablashtiramiz. Matn faqat bitta son emas sonlar va faqat qo’shish amalidan iborat arifmetik ifodadan iborat degan qoidani kiritamiz:

<Qo’shish> := <Son> | <Son> “+” <Qo’shish>

<Son> := <Qo’shish>

<Qo’shish> qoidasi faqat 1 ta sonning o’zi yoki son, “ ” belgisi va yana <Qo’shish> qoidasidan tashkil topgan. Bu rekursiv bog’lanish bizga <Son> <Son> <Son> … ko’rinishidagi ifodalarni ko’rsatishga yordam beradi.

```
// <Qo’shish> qoidasi

public bool Add()
{

   if (NumValue())
   {

      Token t = lex.LookNextToken();

      // Agar keyingi bo’lak amali bo’lsa

      if ((t.Type == Token.TokenType.OPERATOR) && (t.Text == "+"))
      {

         // amalini tashlab yuborish, chunki LookNextToken
         // joriy bo’lakni o’zgartirmaydi

         lex.ReadToken();
```

```
            // Qushishni rekursiv chaqirish va funksiyadan chiqish
            // funksiya qiymati rekursiv chaqiruv qiymatiga teng

            return Add();

        }

        // faqat sonni o'zidan iborat

        return true;

    }

    return false;

}
```

StartProgram proseduralida NumValue() ni Add() ga almashtiramiz. Bu qoidani ham dasturlashtirdik. Natijani dasturni ishlatib tekshirib ko'rishimiz mumkin.

Endi qoidamizni yana o'zgartiramiz bu safar matn to'liq arifmetik ifodadan iborat bo'lishi mumkin bo'ladi:

```
<Qo'shish>        := <Ko'paytirish>
                  | <Ko'paytirish> "+" <Qo'shish>
                  | <Ko'paytirish> "-" <Qo'shish>
<Ko'paytirish>    := <Daraja>
                  | <Daraja> "*" <Ko'paytirish>
                  | <Daraja> "/" <Ko'paytirish>
<Daraja>          := <Son>
                  | <Son> "^" <Daraja>
```

<Son> := ButunSon
| “(“ ButunSon “)”
| HaqiqiySon
| “(“ HaqiqiySon “)”

<Dastur boshi> := <Qo’shish>

Amallar qoidasi har doim deyarli shunday ketma-ketlikda yoziladi. Bunda qoidalar amallarning qaysi biri birinchi qaysi oxiri bajarilishiga qarab yoziladi. Masalan ko’paytirish amali qo’shishdan oldin bajarilgani sababli qo’shish qoidasida avval ko’paytirish qoidasiga murojaat bo’ladi va xok.

Ushbu qoida uchun yuqoridagi dasturimizning Add() funskiyasiga biroz o’zgartirish kiritib va qolgan qoidalarni ham dasturga qo’shsak kifoya.

```
// <Qo'shish> qoidasi

public bool Add()
{

   if (Mul())
   {

      Token t = lex.LookNextToken();

      if (t.Type == Token.TokenType.OPERATOR)
      {

         if ((t.Text == "+") || (t.Text == "-"))
         {

            lex.ReadToken();
```

```
                return Add();

            }

        }

        return true;

    }

    return false;

}

// <Ko'paytirish> qoidasi

public bool Mul()
{

    if (Pow())
    {

        Token t = lex.LookNextToken();

        if (t.Type == Token.TokenType.OPERATOR)
        {

            if ((t.Text == "*") || (t.Text == "/"))
```

```
            {

                lex.ReadToken();

                return Mul();

            }

        }

        return true;

    }

    return false;

}

// <Daraja> qoidasi

public bool Pow()
{

    if (NumValue())
    {

        Token t = lex.LookNextToken();

        if (t.Type == Token.TokenType.OPERATOR)
```

```
        {

            if (t.Text == "^")
            {

                lex.ReadToken();

                return Pow();

            }

        }

        return true;

    }

    return false;

}

// <Son> qoidasi

public bool NumValue()
{

    if (lex.LookNextToken().Type == Token.TokenType.NUMLITERAL)
    {
```

```
            lex.ReadToken();

            return true;

        // <Son>:= "(" <Qo'shish> ")"

        } else if (lex.LookNextToken().Text == "(") {

            lex.ReadToken();

            if (Add() && lex.LookNextToken().Text == ")")
            {

                lex.ReadToken();
                return true;

            } else

                return false;

        } else

            return false;

    }
```

Balki “Sintaktik daraxt qani?” degan savol tug’ilgandir? Sintaktik daraxt aynan shuni o’zidir. Aniqrog’I ushbu rekursiv chaqirilayotgan qoida funksiyalari sintaktik daraxtning shoxchalarini tashkil etadi. Endi sintaktik daraxtni ifodalovchi class yaratib, yuqoridagi funksiyalar tanasida daraxt haqidagi ma’lumotni yozib

boruvchi kod qo’shilsa yetarli. Dasturni ishlatib ko’ramiz va biror to’g’ri ifodani kiritib ko’ramiz:

2 * 3 + 4 / (1 - 2 ^ (1 + 1) * 2)

Parsing OK.

Demak ifoda to’g’ri. Endi biror noto’g’ri ifoda kiritamiz:

2 + 5 * 1 - (1 * 2

Syntax Error: 10

Demak parserimiz matnni to’g’ri tahlil qilyapti. Maqolaning keyingi qismida ushbu parserimizga (va mos ravishda tilimizga) o’zgartirishlar kiritib uni mukammallashtirib boramiz. Shuningdek kompilyasiyaning oxirgi bosqichi – dasturni mashina kodiga tarjima qilish, virtual mashinalar haqida ham fikr almashamiz.

P.S. Maqolada yozilgan kodlar deyarli tekshirilmagan, unda ba’zi “qo’polliklar”, “chiroyli yozish”ga amal qilinmagan holatlar yoki mantiqiy hatolar uchrashi mumkin.

Foydalanuvchi kiritmasidan qiymatlarni yig‘uvchi dasturni yozganingizda nechta qiymatlar bo‘lishini bilmaysiz. Afsuski, *dastur tuzilganda* jadval hajmi aniq bo‘lishi kerak.

Bo‘limda bu muammoni qisman to‘ldirilgan jadvallar qanday yo‘naltirishni ko‘rdingiz. biz quyidagi bo‘limlarda muhokama qiladigan bu vektor tuzilishi qulayroq echimni taklif etadi. Vektor qiymatlar ketma ketligini xuddi jadval qilgandek yig‘adi lekin uning hajmi o‘zgarishi mumkin.

Vektorni aniqlaganingizda burchak kronshteyndagi elementlar turini aniqlashtirasiz. Quyidagicha:

```
vector<double> values;
```

Boshlang‘ich hajmni erkin aniqlashingiz mumkin. Masalan, Bu erda boshlang‘ich

hajmi 10 ga teng bo‘lgan vektorning ta’rifi:

vector<double> values(10);

agar vektorni boshlang‘ich hajmsiz aniqlasangiz uning hajmi 0 ga teng.

Jadvalning 0 hajmini aniqlashda hech qanday nuqta bo‘lmaganda boshlang‘ich hajmi 0 ga teng vektorlarga ega bo‘lish foydali va keyin ular keraklicha o‘stiriladi. Dasturingizda vektorlardan foydalanish uchun siz vektor sarlavhasini ham kiritishingiz zarur

defining Vectors

vector<int> numbers(10); 10 butun vektor

vector<string> names(3); 3 trosli vektor

vector<double> values; 0 hajmli vektor

vector<double> values(); Xato**:** Vektorni aniqlamadi.

Agar qo‘shimcha elementlar kerak bo‘lsa elementni vektorning oxiriga qo‘shish uchun push_back funksiyasini qo‘llaysiz. bunda uning hajmi 1 ga ortadi. Push_back funksiyas i quyidagi nuqtali notatsiya bilan chaqirishingiz shart bo‘lgan a’zo funksiyasidir:

values.push_back(37.5);

Bu chaqiruvdan so‘ng 14 shakldagi vektor qiymati 3 hajmga teng va qiymatlar

values[2] qiymat 37.5 ga teng.

Bo‘sh vektor bilan boshlash va push_back funksiyasi bilan uni to‘ldirish ommalashgan. Masalan,

vector<double> values; // Dastlabki bo‘sh

values.push_back(32); // endi qiymatla 1 hajmga va element esa 32 hajmga ega values.push_back(54); // endi qiymatla 2 hajmga va element esa 32,54 hajmga ega

values.push_back(37.5); // endi qiymatla 3 hajmga va element esa 32,54,37.5 hajmga ega

push_back a’zo funksiyasi uchun boshqa bir foydalanish bu vektorni kiritma qiymatlari bilan to‘ldirish.

```
vector<double> values; // Dastlabki bo‘sh
double input;
while (cin >> input)
{
 values.push_back(input);
}
```

Bu kiritmassikl 6.2.10 bo‘limdagidan osonroq va oddiyroq ekanini qayd eting. Boshqa a’zo funksiyasi, pop_back, vektorning oxirgi elementini olib tashlash, uning xajmini bittaga kichiklashtirish. (15 shaklga qarang):

```
values.pop_back();
```

Nazorat savollari:

1. Inkapsulyasiya nima?
2. Inkapsulyasiya qanday ishlatiladi?
3. Ob’yektga yo’naltirilgan dasturlash prinsiplar qaysilar?
4. Virtual funksiyalar qanday ishlaydi?
5. Virtual funksiyani qolganlaridan farqini sanang.

II. TAJRIBA MASHG'ULOTLARI MATERIALLARI

1-SEMESTR UCHUN

Tajriba ish №1

Mavzu: Chiziqli algoritmlarni dasturlash

Ishdan maqsad. C++ dasturlash tilining ma'lumot tiplari bilan tanishish, o'zgaruvchilarni nomlash, kiritish/chiqarish, format bilan chiqarish va matematik funksiyalardan foydalanishni o'rganish.

Nazariy qism.

Ma'lumot tiplari. Dasturda biror ma'lumotni saqlash va qayta ishlashda har bir ma'lumot ma'lumot qaydaydir tipda saqlanadi. **C++** da sonlarni saqlash uchun birnechta ma'lumot tiplari mavjud. Sonlar 2 xil bo'lishi mumkin: butun va haqiqiy sonlar.

C++ da son tiplari:

Tip	Hotira	Chegarasi
Butun sonlarni saqlovchi o'zgaruvchi tiplari		
Int	4 bayt	-2 147 483 648 dan 2 147 483 647 gacha
unsigned int	4 bayt	0 dan 4294967295 gacha
short int	2 bayt	-32768 dan 32767 gacha
unsigned short int	2 bayt	0 dan 65,535 gacha
long long	8 bayt	-9 223 372 036 854 775 808 dan 9 223 372 036 854 775 807 gacha
unsigned long long	8 bayt	0 dan 18 446 744 073 709 551 615 gacha
Xaqiqiy sonlarni saqlovchi o'zgaruvchi tiplari		
Float	4 bayt	+/- 3.4e +/- 38 (~7 raqam)
Double	8 bayt	+/- 1.7e +/- 308 (~15 raqam)
long double	12 bayt	+/- 3.4e +/- 4932 (~15 raqam)

Butun sonlar uchun **unsigned** (ishorasiz) faqat manfiy bo'lmagan sonlarni saqlash uchun hizmat qiladi.

O'zgaruvchini nomlash.

O’zgaruvchilar bilan ishlash uchun ularning har biriga alohida nom berish lozim. O’zgaruvchini nomlashda uning maqsadini tushintirib beradigan ingliz tilida nom qo’yish lozim. Masalan kub hajmining qiymatini saqlaydigan o’zgaruvchini `cube_volume` deb nomlagan maqsadga muvofiq.

C++ da o’zgaruvchini nomlashda bir nechta qoidalar mavjud:

1. O’zgaruvchi nomi harf yoki pastki chiziq(_) belgisi bilan boshlanishi kerak va harflar, raqamlar va pastki chiziq belgisidan iborat bo’lishi lozim.

2. Boshqa belgilarni, masalan . yoki % qo’llash mumkin emas. Probel ham qatnashmasligi lozim. Ma’noga ega qismlari bir-biridan `cube_volume` ga o’xshab pastki chiziqcha bilan ajratilishi lozim. Masalancan_volume.

3. O’zgaruvchini nomlashda kata va kichik harflar farq qiladi. `cube_volume` va `Cube_volume` o’z garuvchilari boshqa-boshqa.

4. C++ da maxsus ma’noga ega bo’lgan masalan `int, return, for` kabi so’zlarni o’zgaruvchi nomi sifatida qo’llash mumkin emas.

C++ da oz’garuvchini e’lon qilish.

O’zgaruvchini e’lon qilishda dastlab o’zgaruvchi tipi yoziladi, undan so’ng o’zgaruvchi nomi yoziladi.

```
    int a;
```

-2 147 483 648 dan 2 147 483 647 gacha bo’lgan butun sonni saqlashi mumkin bo’lgan `a` deb nomlangan o’zgaruvchi e’lon qilindi. O’zgaruvchini tavsiflashda unga qiymat berishimiz va birnechta o’zgaruvchini birgalikda tavsiflashimiz mumkin:

```
int main() {
    int a = 10, b, c, d = 20;
    double g = 9.81, s;
}
```

Bunda `a, b, c` va int tipidagi o’zgaruvchilar tavsiflandi. `a` ning qiymati 10, g ning qiymati `9.81, d` ning qiymati 20, `b, c va s` ga qiymat berilmagan.

Kiritish va Chiqarish

O’zgaruvchilarga qiymatni klaviatura orqali kiritish. Bu o’zgaruvchilardan foydalanib amallar bajarish va chiqarish mumkin.
Ikki sonning yig’indisini topadigan dastur tuzamiz. Kiruvchi o’zgaruvchilarni saqlash uchun ikkita o’zgaruvchi kerak.

```
#include <iostream>
```

```
using namespace std;

int main() {
     int a, b;
     cin>>a>>b;
     int c = a + b;
     cout<<c;
}
```

Bu dasturni ko’rib chiqamiz.

`#include <iostream>` - kiritish chiqarish`(cin, cout)`bilan ishlash uchun e’lon qilinadigan kutibxona.

`using namespace std` – standart kutibxonalardan foydalanganda har bir dasturda uni yozish lozim

`int a, b` – `a va b` o’zgaruvchilarni tavsiflash

`cin>>a>>b cin` – oz’garuvchilar qiymatlarini kiritish uchun xizmat qiladi. Dastlab kiritilgan sonni `a` o’zgaruvchi , keying kiritilgan sonni `b` o’zgaruvchi o’zida saqlaydi. Sonlarni probel yoki enter bilan ajratilagan holda kiritish mumkin.

`int c = a + b` c o’zgaruvchi a va b larning qiymatlari yig’indisini o’zlashtiradi. Bunda a va b larning qiymatlari o’zgarmaydi.

`cout<<c` – `c` o’zgaruvchining qiymatini chiqarish.

Barcha amallar yuqoridan pastga, bir qatorda chapdan o’ngga bajarilib keladi.

C++ da matematik funksiyalar.

Matemetik funksiyalardan foydalanish uchun `math.h` jutibxonasini e’lon qilish lozim.

Funksiya	Tavsifi	Misol
abs(a)	**a** ning moduli	abs(-3)= 3 abs(5)= 5
sqrt(a)	**a** ning kvadrat ildizi	sqrt(9)=3.0
pow(a, b)	**a** ni **b** darajaga ko’tarish	pow(2,3)=8
ceil(a)	**a** ni o’zidan kichik bo’lmagan eng kichik butun songa yaxlitlash	ceil(2.3)=3.0 ceil(-2.3)=-2.0
floor(a)	**a** ni o’zidan katta bo’lmagan eng kichik butun songa yaxlitlash	floor(12.4)=12 floor(-2.9)=-3
fmod(a, b)	a/b ni hisoblashdagi qoldiqni olish	fmod(4.4, 7.5) = 4.4 fmod(7.5, 4.4) = 3.1
exp(a)	e^a ni hisoblash	exp(0)=1
sin(a)	***sina,*** **a** radiyanda beriladi.	

Funksiya	Tavsifi	Misol
cos(a)	***cosa*, a** radiyanda beriladi.	
log(a)	**a** natular logarifmi	log(1.0)=0.0
log10(a)	**a** ning o'nlik logarifmi	Log10(10)=1
asin(a)	**arcsin*a***, bunda **-1.0 < a < 1.0.** Natija radiyanda xosil bo'ladi	asin(1)=1.5708

Bo'linmaning haqiqiy qismi kerak bo'lga, agar o'zgaruvchilar butun son bo'lsa bo'lish amaliga e'tibor qaratish lozim.

Misol. Asosining uzunligi a va balandligi h ga teng bo'lgan uchburchakning yuzasini hisoblovchi dastur tuzing.

Yechimi.

Kiruvchi ma'lumot a va h butun sonlari. Uchburchak yuzasi formulasi: s=$\frac{ah}{2}$.

a va b sonlari butun, lekin s soni haqiqiy son.

```
#include <iostream>
using namespace std;
int main() {
     int a, h;
     cin>>a>>h;
     double s = a * h / 2;
     cout<<s;
}
```

Dasturda hatolik mavjud. Bu hatolik shundan iboratki, butun sonlarni bo'lganda bo'linmaning butun qiymati hisoblanadi. Bo'linmaning haqiqiy qiymatini hisoblash uchun bo'linuvchilardan birining qiymati haqiqiy bo'lishi kerak. Yuqoridagi masalada buni

```
     double s = a * h / 2.0;
```

yoki

```
     double s = 1.0 * a * h / 2;
```

ko'rinishida yozish orqali to'girlash kiritishimiz mumkin.

Murakkab topshiriq bo'yicha na'muna:

$$AF = 2^{-x} \cdot \sqrt{x + \sqrt[4]{|y|+2}} \cdot \sqrt[3]{e^{x-1} / \sin(z+2) + 2};$$

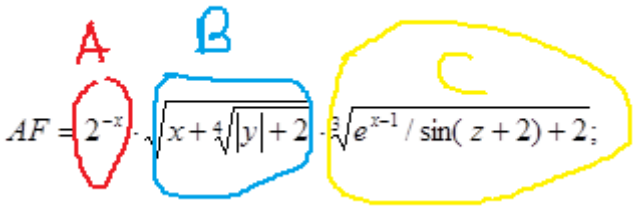
Bunda kiruvchi ma'lumotlar x, y, z haqiqiy sonlari.
Chiquvchi ma'lumot AF.

```
#include <iostream>
#include <math.h>
```

```
#include <stdio.h>

using namespace std;

int main() {
	double x, y, z;
	cin>>x>>y>>z;
	double AF = pow(2, -x) * sqrt(x + sqrt(sqrt(fabs(y)+2))) *
pow(exp(x-1) / sin(z+2) + 2, 1. / 3);
	printf("%.2f", AF);
}
```

`printf()` funksiyasi xaqiqiy sonni nuqtadan so'ng biror xona aniqlikda chiqarish uchun hizmat qiladi. Agar sonning qiymati 3.5689 ga teng bo'lsa yaxlitlab chiqarilganda 3.57 soni chiqariladi.

Katta ifodani yozishni o'rniga uni qismlarga ajratishimiz ham mumkin:




```
#include <iostream>
#include <math.h>
#include <stdio.h>

using namespace std;

int main() {
	double x, y, z;
	cin>>x>>y>>z;
	double A = pow(2, -x);
	double B = sqrt(x + sqrt(sqrt(fabs(y)+2)));
	double C = pow(exp(x-1) / sin(z+2) + 2, 1. / 3);
	double AF = A * B  * C;
	printf("%.2f", AF);
}
```

Topshiriqlar.

1-Topshiriq

1. O`lchami x bo`lgan qubik berilgan. Uning hajmini toping.
2. Radyuslari r1, r2, r3 bolgan 3 to doira radyuslari berilgan. Doiralarni yuzini hisoblang.

3. Yuzasi s va balandligi h bo`lgan uch burchag berilgan. Uni asosini toping.
4. Radyusi r bo`lgan sharing yuzini toping.
5. a,b va c tamonli uch burchag berilgan. Uch burchagning pirimetiri topilsin.
6. Asoslariavab, balandligihbo`lgang`olayuzinitoping.
7. Qo`nisni balandligi h va radyusi r bo`lsa uni hajmi nimaga teng bo`ladi.
8. Tezligi v bo`lgan avtomabil s masofani qancha vaqtda bosib o`tadi.
9. h balandlikqan erkin tushayotgan jism qancha vaqtdan keyin erga uriladi.
10. Jo`mrakdan 1 s da 1 milli litr suv tomsa x yilda necha litr suv tomadi.
11. 1 dan n gacha sonlar berilgan. Berilgan sonlarni yig`indisini toping.
12. Massasi m bo`lgan jismni og`irligini toping.
13. m massali jismga a tezlanish berilganda unga qanchali kuch tasir qiladi.

2-Topshiriq

1. $Z = \frac{ax}{by^2} + \frac{a}{b^2} + c^3 + \frac{a^2 + 6a^2b^3 + 2c^2}{\sqrt[3]{a^3 + b^3 - 2ab}}$

2. $Q = \frac{Xa\sin^2 y^2 + ab\cos^2 x^3}{by^2} + \frac{a^2 + 2b^2 + 3c^3}{a + 2c^2 + \sqrt[3]{7a}}$

3. $y = 2x^2 + b^2x^3 + \sqrt[3]{a + b^2} - \frac{a^3 + b^3 + 3cd}{\sqrt[5]{2|a - b| + c^2}}$

4. $Z = \frac{a + b^3 + c^2}{2abc} + \sin^2 x^3 + (a + b)^{2\sin^2 x} + b\cos^2 y^3$

5. $y = \frac{2a^2 + bc^3 + 2d}{\sqrt[3]{a^3 - b^3}\,3abc} - \frac{(a + b)^2 + 2ac^2}{a^2 + \sqrt[3]{b^2} + \ln x}$

6. $Q = \frac{2^{a+b} + c^2 + \sqrt[3]{x^2}}{(a - b)^2} - \frac{2x^2 + 7a^2b^3 + c^2}{a^2 + 8b^3 + c^2}$

7. $y = \frac{y\sqrt[5]{7{,}2631} + x\sqrt[3]{71{,}8672}}{\sqrt[3]{a} + \sqrt[5]{b}} - \frac{2a + 2b^2}{c^2 + d^2}$

8. $y = \frac{a + bx + c^2}{a^2 - b^2} + \frac{\ln x^2 - \ln ax}{2ab + c^2}$

9. $Z = \sqrt[5]{\frac{a}{x^2}} + \sqrt[3]{\frac{c^2}{b^2}} - \frac{2ab + c^2}{\sqrt[3]{a + b} - 2ab^2}$

10. $Z = \frac{2x^3 + \sqrt[3]{a + x^2}}{a^2 + bc} + \frac{\lg b + \ln c^2}{\cos^2 ax^3}$

11. $y = ax^2 + b + \frac{\sin x^2 + c}{\sqrt[3]{a + b^2}}$

12. $y = \frac{ax^3 + 7bx^2 + 8c^3}{x^2 + 3a^3} + \frac{a + 2ab + c^2}{\sqrt{a^2 - b^2}}$

13. $Q = \frac{ax^2 + \sqrt[3]{bc}}{2ab} + \frac{\sin^2 x + b^3}{\ln ax^2}$

3-Topshiriq

1. $Q = \frac{2x^2 + 7a^2b^3 + c^2}{a^2 + 8b^3 + c^2}$

2. $y = \sqrt[3]{a + b^2} - \frac{a^3 + b^3 + 3cd}{\sqrt[5]{2|a - b| + c^2}}$

3. $Q = \frac{Xa \sin^2 y^2 + ab \cos^2 x^3}{by^2}$

4. $y = \frac{a + bx + c^2}{a^2 - b^2}$

5. $y = \frac{2a^2 + bc^3 + 2d}{\sqrt[3]{a^3 - b^3}\, 3abc}$

6. $Z = \sin^2 x^3 + (a + b)^{2\sin^2 x} + b \cos^2 y^3$

7. $Z = \sqrt[3]{\frac{c^2}{b^2}} - \frac{2ab + c^2}{\sqrt[3]{a + b} - 2ab^2}$

8. $Z = \frac{a}{b^2} + \frac{a^2 + 6a^2b^3 + 2c^2}{\sqrt[3]{a^3 + b^3 - 2ab}}$

9. $y = \frac{y\sqrt[5]{7{,}2631} + x\sqrt[3]{71{,}8672}}{\sqrt[3]{a} + \sqrt[5]{b}}$

10. $y = \frac{ax^3 + 7}{x^2 + 3a^3} + \frac{ab + c^2}{\sqrt{a^2 - b^2}}$

11. $Q = \frac{ax^2}{2ab} + \frac{\sin^2 x}{\ln ax^2}$

12. $Z = \dfrac{2x^3 + \sqrt[3]{a + x^2}}{a^2 + bc}$

13. $y = b + \dfrac{\sin x^2 + c}{\sqrt[3]{a + b^2}}$

4-Topshiriq

1. $c1 = \dfrac{x + y}{y^2 + \left|\dfrac{y^2 + 2}{x + x^3/5}\right|} + e^{y+2}$ x - butun, y - haqiqiy.

2. $f1 = \dfrac{2tg(x + \pi/6)}{1/3 + \cos^2(y + x^2)} + \log_2^{(x^2+2)}$ x, y - haqiqiy

3. $f2 = \dfrac{1/(x + 2/x^2 + 3/x^3) + e^{x^2+3x}}{arctg(x + y) + |5 + x|^2} - \cos^2(y^2 + x^2/2);$

4. $z = \ln\left|(x + y)^2 + \sqrt{|y| + 2} - (x - \dfrac{xy}{x^2/2 - 5})\right| + \dfrac{\cos^2(x + y)}{(x + y)^{1/3}};$ x,y- butun

5. $T11 = \dfrac{x^2 + 1}{x^2 + \dfrac{xy + y^2}{y^2 + \dfrac{y + xy}{|xy| + 5}}} + \dfrac{1}{1 + \cos x + \dfrac{1}{\sin|x|}}$ x, y- haqiqiy son

6. $T = \sqrt[5]{a} + \sqrt[4]{b \cdot \dfrac{ax^2 + b}{2 \cdot b + a \cdot b}} \cdot (a^2 + x^2 + b^2 + 2)$ a, b- haqiqiy, x - butun son

7. $F = \left|\dfrac{\sin^2|cx_2^3 + dx_1^3 - cd|}{\sqrt{(cx_1^2 + dx_2^2 + 5) + 2}}\right| + tg(x_1 \cdot x_2^2 + d^3);$ x_1, x_2 - haqiqiy, s, d - butun.

8. $y2 = \dfrac{ax^2 + bx + c}{xa^3 + a^2 + a^{b-c}} + \cos\left|\dfrac{ax + b}{cx + d + 2^c}\right|$ a, b, c, d - butun, x - haqiqiy.

9. $W1 = 0.75 + \dfrac{8.2x^2 + \sqrt{|x^3 + 3x| + \cos(x-2)}}{a/4 + b/3 + c/2 + 1}$ a, b, c - butun, x - haqiqiy.

10. $TT = \dfrac{\sqrt{x-1} + \sqrt{x+2} + \lg(\sqrt{ax^2} + 2)}{\sqrt[2]{\sqrt{x+2} + \sqrt{x+24} + x^5}}$ x - haqiqiy, a - butun.

11. $W2 = \sqrt{e^{xy} - x \cdot \sin(ax) - \dfrac{x^2+2}{|x|+5}} + \sqrt{\ln(x^2+2)+5}$ a - butun,x, y - haqiqiy.

12. $AA = \sqrt{\dfrac{2tg(x+2) - \cos(x+2^x)}{1+\cos^2(x+2)}} + \dfrac{\sin x^2}{x^2+3}$ x - haqiqiy son.

13. $BB1 = x \cdot \sin(x/2 + x/3 + x/4) + \dfrac{\lg(x^2-2) + 3^a}{\cos(x+3) \cdot \sin(x+3) + 8}$ a - butun, x - haqiqiy.

14. $TT = \sqrt{y^2 + e^x + \sqrt{e^x + \dfrac{a}{x^2+2}}} + \dfrac{\cos^2 x}{\sin x^2} + \cos^3 x;$ a - butun, x - haqiqiy.

Tajriba ish №2

Mavzu: Tarmoqlanuvchi algoritmlarni dasturlash

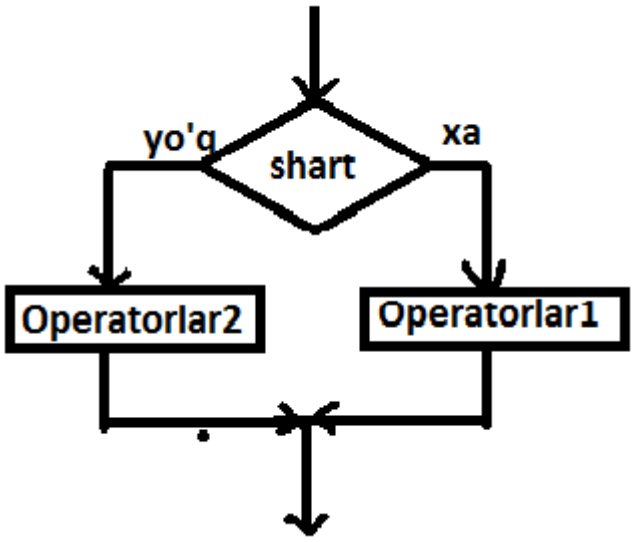
Ishdan maqsad. if shart, case tanlash operatorlarini tog'ri qo'llashni o'rganish, tarmoqlanuvchi algoritmga doir masalalarni dasturini tuzishni o'rganish.

Nazariy qism.

Agar algoritm qadamlari ketma-ket bajarilish jarayonida qandaydir shartga bo'gliq ravishda o'zgarsa, bunday algoritm tarmoqlanuvchi algoritm deb nomlanadi. Shart bu mantiqiy ifoda bo'lib, faqat rost yoki yolg'on qiymatni qabul qiladi. Agar shart rost bo'lsa Xa, yolg'on bo'lsa Yo'q tarmog'i bo'yicha algoritm qadami davom etadi.

Tarmoqlanuvchi algoritm to'liq tarmoqlanuvchi va to'liqmas tarmoqlanuvchi turlariga bo'linadi.

To’liq tarmoqlanuvchi algoritmda shart bajarilganda va bajarilmaganda ikkalasida ham amallar bajariladi.



Agar shart bajarilsa Operatorlar1 bajariladi, aks holda Operatorlar2 bajariladi.

Tarmoqlanish shart asosida bo’ladi. Shart manitiq ifoda bo’ladi. Mantiqiy ifoda mantiqiy o’zgaruvchi, taqqoslash amallari yoki ularning inkor, konyuksiya, dizyunksiya amallaridan iborat bo’lsihi mumkin. Shart operatori C++ da shart operatori quyidagicha yoziladi:

```
if (shart) {
        Operatorlar1;
}
else {
        Operatorlar2;
}
```

C++ da taqqoslash amallari:

№	Matematika	C++
1	>	>
2	<	<
3	≥	>=
4	≤	<=

5	=	==
6	≠	!=

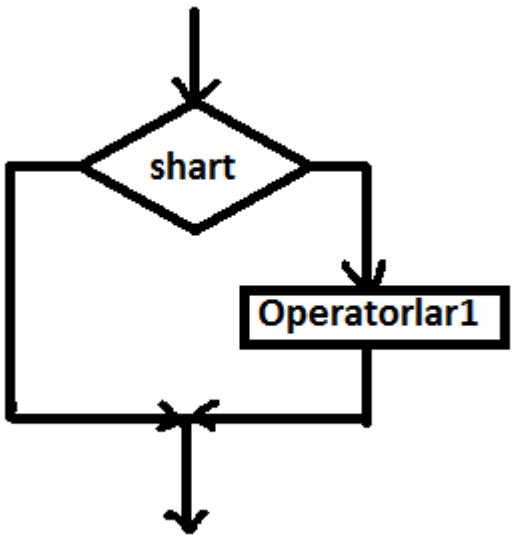
Misol1. $y=\begin{cases} x^2 \ agar \ x \geq 0 \\ 2x \ agar \ x < 0 \end{cases}$

Yechimi: y ning qiymati x ga bog'liq ravishda yoki x^2 formula, yoki $2x$ formula bo'yicha hisoblanadi. Tekshirilishi kerak bo'lgan shart $x \geq 0$.

```
#include <iostream>

using namespace std;

int main() {
	double x, y;
	cout<<"x=";
	cin>>x;
	if (x >= 0) {
		y = x * x;
	}
	else {
		y = 2 * x;
	}
	cout<<"y="<<y;
}
```

To'liqmas tarmoqlanuvchi algoritmda shart bajarilganda bu shartga bog'liq amallar bajariladi, bajarilmagan holatda hech qanday amal bajarish shart emas.

C++ da to'liqmas tarmoqlanuvchida faqat **if** operatori ishlatiladi, **else** ishlatilmaydi.



Misol2. *a* va *b* sonlari berilgan. Ulardan kattasini topuvchi dastur tuzing.

Yechimi: Dastavval *a* sonni maksimal deb tasavvur qilamiz. Agar b soni undan katta bo'lsa u holda *b* soni maksimal bo'ladi.

```
#include <iostream>

using namespace std;

int main() {
	double a, b;
	cout<<"Birinchi sonni kiriting: ";
	cin>>a;
	cout<<"Ikkinchi sonni kiriting: ";
	cin>>b;
	double max = a;
	if (b > max)
		max = b;
	cout<<a<<" va "<<b<<" sonlarining maksimali "<<max<<" ga teng";
}
```

if else ning boshqacha shaklda yozilishi.

if va else operatorlarini qisqacha shaklda ? va : belgilari orqali yozish mumkin.

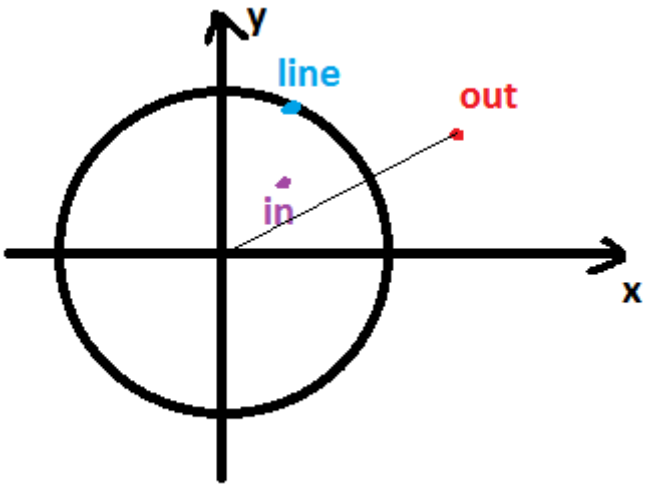
Misol3. n natural soni berilgan. Agar u toq bo'lsa "odd", juft bo'lsa "even" so'zini chiqaruvchi dastur tuzing.

Yechimi: n natural soni toq bo'lishi uchun uni ikkiga bo'lganda qoldiq 1 ga teng bo'lishi kerak, aks holda juft bo'ladi.

```
#include <iostream>

using namespace std;

int main() {
	int n;
	cin >> n;
	n % 2==1 ? cout<<"odd" : cout<<"even";
}
```

Murakkab tarmoqlanuvchi.
Agar biror shart asosida tarmoqlangandan so'ng yana shart asosida tarmoqlansa(ya'ni **else if**), bunday tarmoqlanish murakkab tarmoqlanish deyiladi.

Misol4. Markazi koordinatalar boshida va radiyusi R ga teng bo'lgan aylana berilgan. Tekislikdagi (x,y) nuqta bu aylanaga tegishliligini aniqlang. Agar aylana tashqarisida yotsa "out", chizig'ida yotsa "line", ichida yotsa "in" so'zini chiqaring.



Yechimi: Berilgan nuqatdan koordinata boshigacha masofani topamiz. Qaysi holat bo'lishi bu masofaga bo'g'liq. **Masofa d=**$\sqrt{x^2+y^2}$ ga teng**.** Agar **d>R** bo'lsa u holda nuqta aylanadan tashqarda, aks holda agar **d=R** bo'lsa u holda nuqta aylanaga tegishli, aks holda(ya'ni bu holatda faqat d<R shart qoldi) nuqta aylana ichkarisida yotadi. Bunga mos C++ da gi yechimi quyidagicha bo'ladi.

```
#include <iostream>

using namespace std;

int main() {
	int x, y, R;
	cout<<"x=";
	cin>>x;
	cout<<"y=";
	cin>>y;
	cout<<"R=";
	cin>>R;
	if (x*x+y*y > R*R) {
		cout<<"out";
	}
	else if (x*x+y*y==R*R) {
		cout<<"line";
	}
	else {
		cout<<"in";
	}
}
```

C++ da murakkabroq shartlarni yozish.

Murakkab shart sodda shartlarning konyuksiya, dizyunksiya va inkorlaridan tashkil topadi.

Berilgan sonning [a, b] intervalga tegishli ekanligini aniqlash uchun, **x≥a** va **x≤b** shartlari bir vaqtning o'zida o'rinli bo'lishi kerak. Shartlarning ikkalasi ham bajarilish shartini && (va - and) amali orqali yozamiz:

```
if (x >= a && x <= b)
     cout<<"Tegishli";
else
     cout<<"Tegishli emas";
```

Berilgan sonning [a, b] intervalga tegishli emas ekanligini aniqlash uchun, **x<a** yoki **x>b** shartlari istalgan biri bajarilishi kerak. Shartlarning istalgan biri bajarilishi yetarliligi shartini || (yoki - or) amali orqali yozamiz:

```
if (x < a || x > b)
     cout<<"Tegishli emas";
else
     cout<<"Tegishli";
```

Tanlash operatori.
Tanlash opratori switch tanlanuvchi ifoda qiymatini birnechtakonstantalar bilan taqqoslab chiqadi. switch case ko'plik tanlov operatori hisoblanadi. switch da ko'rsatilgan ifosa qiymati case so'zidan keyin yozilgan har bir qiymat bilan taqqoslab chiqiladi. Taqqoslanuvchi qiymat qaysidir qatordagi case operatoridan yozilgan qiymatga teng u holda uning davomida yozilgan amallar bajariladi.

Misol5. Hafta kuni raqamda barilgan. Uni so'zda chiqaruvchi dastur tuzing.
Yechimi:

```
#include <iostream>

using namespace std;

int main() {
     int n;
     cout<<"Hafta kunini raqamda kiriting: ";
     cin>>n;
     switch (n) {
          case 1: cout<<"Dushanba"; break;
          case 2: cout<<"Seshanba"; break;
          case 3: cout<<"Chorshanba"; break;
          case 4: cout<<"Payshanba"; break;
          case 5: cout<<"Juma"; break;
          case 6: cout<<"Shanba"; break;
          case 7: cout<<"Yakshanba"; break;
```

```
            default: cout<<"Hato kiritildi"; break;
      }
}
```

Agar har bir qatordan so'ng **break** yozilmasa u holda qaysidir shart bajarililadigan bo'lsa keying break operatori kelgunga qadar barcha holatdagi amallar bajariladi. Masalan quyidagi dasturda

```
      switch (n) {
            case 1: cout<<"Dushanba";
            case 2: cout<<"Seshanba";
            case 3: cout<<"Chorshanba";
            case 4: cout<<"Payshanba";
            case 5: cout<<"Juma"; break;
            case 6: cout<<"Shanba"; break;
            case 7: cout<<"Yakshanba"; break;
            default: cout<<"Hato kiritildi"; break;
      }
}
```

agar n=2 bo'lsa u holda ekranga SeshanbaChorshanbaPayshanbaJuma lar chiqadi.

1-topshiriq

1. *x* va *y* haqiqiy son berilgan. Xisoblang: max(*x*, *y*) va min(*x*, *y*)
2. *x*, *y* va *z* haqiqiy son berilgan. Xisoblang: max(*x*, *y*, *z*) va min(*x*, *y*, *z*)
3. *x*, *y*va *z* haqiqiy son berilgan. Xisoblang: max(x+y+z, x, y, *z*) va $\min^2(x+y/2, x, y, z)$
4. a, b va c haqiqiy son berilgan. Tekshiring: a<b<c tengsizlik bajariladimi?
5. a, b va c haqiqiy son berilgan. Agar $a \geq b \geq c$ tengsizlik bajarilsa, u xolda haqiqiy sonlarni ikkilantiring, aks xolda ularni modullari bilan almashtiring.
6. Ikkita haqiqiy son berilgan. Birinchi sonni chiqaring, agar u ikkinchisidan katta bo`lsa, aks xolda ikkalasini ham chiqaring.
7. Ikkita haqiqiy son berilgan. Birinchi sonni no`l bilan almashtiring, agar u ikkinchisidan kichik yoki teng bo`lsa, aks xolda o`zgartirishsiz koldiring.
8. Uchta haqiqiy son berilgan. Ulardan [1,3] intervalga tegishlilarini tanlang.
9. *x*, *y*(*x* va *y* teng emas) haqiqiy son berilgan. Ularning kichigini ularning yarim yig'indisi bilan, kattasini ularning ikkilangan ko`paytmasi bilan almashtiring.
10. Uchta haqiqiy son berilgan. Ularning musbatini kvadrati bilan almashtiring.
11. Agar uzaro farqli *x*, *y* ,*z* haqiqiy sonlar birdan kichik bo`lsa, u xolda bu uchta sondan eng kichigini boshka ikkitasining yarim yig'indisi bilan almashtiring. Aks xolda o`zgarishsiz qoldiring.
12. a, b, c va d haqiqiy sonlar berilgan. Agar $a \leq b \leq c \leq d$ tengsizlik bajarilsa, u xolda ularning har birini ularning kattasi bilan almashtiring.
13. *x*, *y*haqiqiy sonlar berilgan. Agar *x* va *y* manfiy bo`lsa, ularning har birini modullari bilan almashtiring; agar fakat bittasi manfiy bo`lsa ikkala sonning har birini 0.5 ga oshiring.
14. *x*, *y* , *z* haqiqiy musbat sonlar berilgan. *x*, *y*, *z* uzunlikka ega tomonli uchburchak mavjudmi?
15. a, b, c haqiqiy sonlar berilgan ($a \neq 0$). Aniqlanki, $ax^2+bx+c=0$ kvadrat tenglama haqiqiy echimga egami. Agar haqiqiy echimlari mavjud bo`lsa, u xolda ularni toping. Aks xolda haqiqiy echimlar mavjud emasligi xaqida xabar bering.

2-topshiriq

1 $Q = \begin{cases} \dfrac{ax^2 + \sqrt[3]{bc}}{2ab}, & \text{агар } ab \neq 0 \\ \dfrac{\sin^2 x + b^3}{\ln ax^2} & \text{агар } a > 0 \end{cases}$

2 $y = \begin{cases} \dfrac{\sin^2 ax + \cos^3 ax^2}{(a+b)^2 + c}, & \text{агар } a < 0 \\ \dfrac{\sqrt{a^2 - b^3} + 2ac}{\sin^2 a + bx^2}, & \text{агар } a^2 \geq b^3 \end{cases}$

3 $y = \begin{cases} \dfrac{a + b^2 + 2abx}{\sqrt[3]{c^2 + b^3}}, & \text{агар } x > 0 \\ \dfrac{a^2 + 2ab + c^2}{7a^2 + 8b^3}, & \text{агар } a \neq 0 \end{cases}$

4 $y = \begin{cases} \dfrac{a\sin x + b\cos x^2}{a^2 + b^2}, & \text{агар } ab \neq 0 \\ \dfrac{a^3 + 2ab^2 + c^3}{a + b^2}, & \text{агар } b > 0 \end{cases}$

5 $Z = \begin{cases} \dfrac{2x^3 + ax^2 + c^3}{a^2 - c^2}, & \text{агар } a > c \\ \dfrac{2ab + \sin ax^2}{\sqrt{a^2 + 2q^2}}, & \text{агар } q \neq 0 \end{cases}$

6 $y = \begin{cases} \dfrac{2c^2 + ab\cos x^2}{a\sin x^2 + b^3}, & \text{агар } a > 0 \\ \dfrac{a^2 + 2bc^3}{\sqrt{a^3 + 2c^2 + d^3}}, & \text{агар } d \neq 0 \end{cases}$

7 $Q = \begin{cases} \dfrac{a + b^2 + 2c^3}{a \cdot \sin^2 x^2}, & \text{агар } a > |b| \\ \dfrac{a^2 + 2ab + c^2}{a + 2c^2 + xy}, & \text{агар } xy > a \end{cases}$

8 $Z = \begin{cases} \sqrt[3]{cy^2 + a^3} - \ln x, & \text{агар } x > 0 \\ \dfrac{a^2 + bx + c^2}{\sqrt{a^3 - b^3}}, & \text{агар } a^3 > b^3 \end{cases}$

9 $y=\begin{cases}\dfrac{ax^2+cx^3}{\sqrt{a^2-b}+c}, \text{ agar } a^2>b\\ \dfrac{\sin^2 x^3}{\cos x^2+ab}, \text{ agar } x\neq 0\end{cases}$

10 $Q=\begin{cases}\dfrac{2x+ax^2+c^3}{a^2+b^2}, \text{ agar } x>c\\ \dfrac{a\left(x^2+y\right)+2xy^2}{a^2-b^2}, \text{ agar } a\neq b\end{cases}$

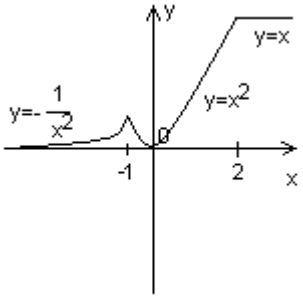
11 $y=\begin{cases}\dfrac{ax^2+b+2c^3}{x^2+g^2}, \text{ agar } x\neq g\\ \dfrac{2c^2+\sqrt[5]{a^2}}{c^3+ax^2+d}, \text{ agar } x<0\end{cases}$

12 $Z=\begin{cases}\dfrac{2x^2+ab^3+c^2}{2a+c^2}, \text{ agar } x\neq 0\\ \dfrac{2a^2+7b^3+12c}{2x+7ax^2+c^3}, \text{ agar } c>0\end{cases}$

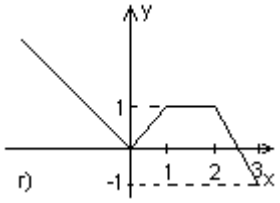
13 $U=\begin{cases}\dfrac{a^2+4abc+c^2}{a-b^2}, \text{ agar } a>b^2\\ \sqrt[3]{\dfrac{2a}{c^2}}+\sqrt[5]{\dfrac{c^2}{x^2}} \text{ agar } x\neq 0\end{cases}$

3-topshiriq

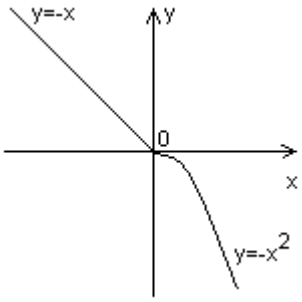
1. a xaqiqiys onberilgan. Quydagi rasmlardatasvirlangany(x)funktsiyauchuny(a)ni xisoblang.



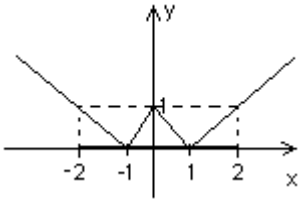
2. axaqiqiysonberilgan. Quydagirasmlardatasvirlanganf(x) funktsiyauchunf(a)ni xisoblang.



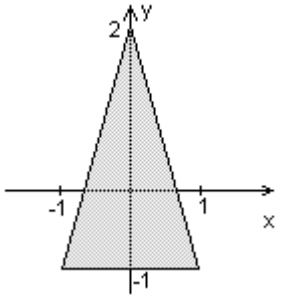
3. axaqiqiysonberilgan. Quydagi rasmlardatasvirlangany(x)funktsiyauchuny(a)ni xisoblang.



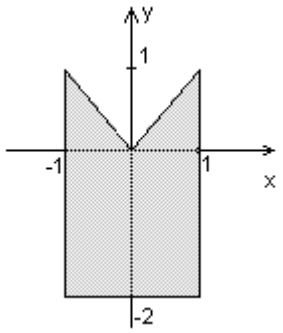
4. axaqiqiysonberilgan. Quydagi rasmlardatasvirlanganf(x) funktsiyauchunf(a)ni xisoblang.



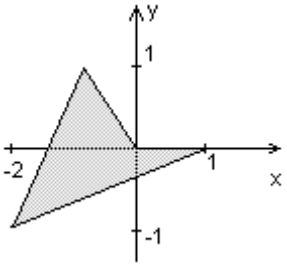
5. x, y xaqiqiysonlarberilgan. Koordinatalari (x, y) bo`lgannuqtaquyda keltirilgan rasmdagi tekislikningshtrixlanganqismigategishlimi? (ha/yo`q)



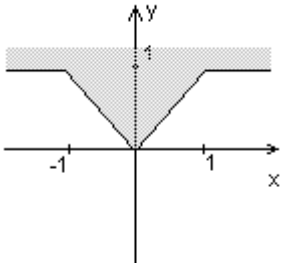
6. x, y xaqiqiysonlarberilgan. Koordinatalari (x, y) bo`lgannuqtaquyda keltirilgan rasmdagi tekislikningshtrixlanganqismigategishlimi? (ha/yo`q)



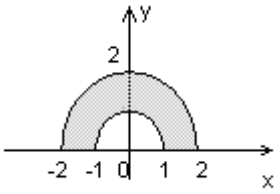
7. x, y xaqiqiysonlarberilgan. Koordinatalari (x, y) bo`lgannuqtaquyda keltirilgan rasmdagi tekislikningshtrixlanganqismigategishlimi? (ha/yo`q)



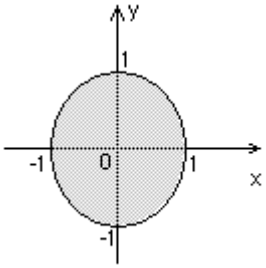
8. x, y xaqiqiysonlarberilgan. Koordinatalari (x, y) bo`lgannuqtaquyda keltirilgan rasmdagi tekislikningshtrixlanganqismigategishlimi? (ha/yo`q)



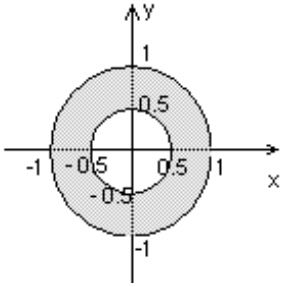
9. x, y xaqiqiysonlarberilgan. Koordinatalari (x, y) bo`lgannuqtaquyda keltirilgan rasmdagi tekislikningshtrixlanganqismigategishlimi? (ha/yo`q)



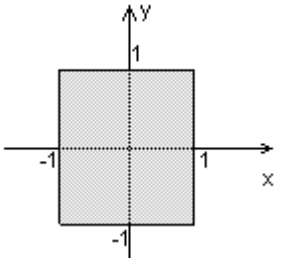
10. x, y xaqiqiysonlarberilgan. Koordinatalari (x, y) bo`lgannuqtaquyda keltirilgan rasmdagi tekislikningshtrixlanganqismigategishlimi? (ha/yo`q)



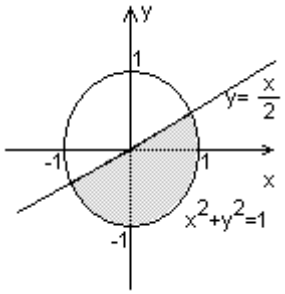
11. x, y xaqiqiysonlarberilgan. Koordinatalari (x, y) bo`lgannuqtaquyda keltirilgan rasmdagi tekislikningshtrixlanganqismigategishlimi? (ha/yo`q)



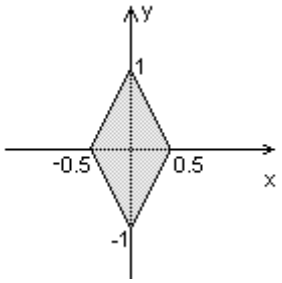
12. x, y xaqiqiysonlarberilgan. Koordinatalari (x, y) bo`lgannuqtaquyda keltirilgan rasmdagi tekislikningshtrixlanganqismigategishlimi? (ha/yo`q)



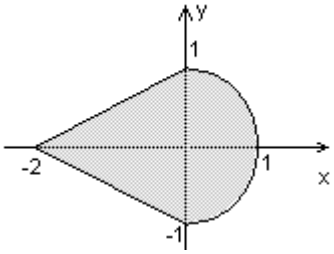
13. x, y xaqiqiysonlarberilgan. Koordinatalari (x, y) bo`lgannuqtaquyda keltirilgan rasmdagi tekislikningshtrixlanganqismigategishlimi? (ha/yo`q)



14. x, y xaqiqiysonlarberilgan. Koordinatalari (x, y) bo`lgannuqtaquyda keltirilgan rasmdagi tekislikningshtrixlanganqismigategishlimi? (ha/yo`q)



15. x, y xaqiqiysonlarberilgan. Koordinatalari (x, y) bo`lgannuqtaquyda keltirilgan rasmdagi tekislikningshtrixlanganqismigategishlimi? (ha/yo`q)



Tajriba ish №3

Mavzu: Takrorlanuvchi algoritmlarni dasturlash

Ishdan maqsad: C++ dasturlash tilining sikl operatorlari bilan tanishish. Sharti oldindan, shart oxiridan beriladigan takrorlanish va parametrik sikl operatorlarini o'rganish.

Nazariy qism.

Dastur kodining biror qismining ko'p marta bajalishi sikl hisoblanadi. Dastur kodining qandaydir qismini qandaydir shart asosida birnecha marta bajartirish uchun dasturlashda sikldan foydalaniladi. Agar shart rost bo'lsa sikl davom qiladi. Aks holda to'xtatiladi. Agar shart hamisha rost bo'lsa bunday sikl cheksiz sikl deb ataladi.

C++ da siklni tashkil qilish uchun *while*, *do while* va *for* operatorlari mavjud.

Siklni o'rganish uchun eng oson misol bu 1 dan *n* gacha natural sonlarning yig'indisini(1+2+3+…+*n*) topish dasturini tuzish. Bu yig'indini takrorlanish jarayoni orqali hisoblash uchun 1 dan n gacha sonlarni birma-bir qo'shib chiqish lozim. Yig'indining dastlabki qiymatini 0 ga tenglaymiz. Siklning har bir qadamida quyidagi amallar bajariladi:

s=0;
1-qadam. *s*=*s*+**1**=0+1=1;
2-qadam. *s*=*s*+**2**=1+2=3;
3-qadam. *s*=*s*+**3**=3+3=6;
4-qadam. *s*=*s*+**4**=6+4=10;
5-qadam. *s*=*s*+**5**=10+5=15;
…….
i-qadam. *s*=*s*+***i;***
…….
n-qadam. *s*=*s*+***n***;

Har bir qadamda bir xil amal bajariladi, ya'ni yog'indining yangi qiymatini hosil qilish uchun uning avvalgi qadamdagi qiymatiga navbatdagi natural son qo'shiladi.

1) *while* sikli.

Bu siklda shart oldindan qo'yiladi. Agar shart rost bo'lsa sikl tanasi bajariladi. Aks holda sikl to'xtab undan keyingi qadamga o'tiladi.

```
while (shart) {

    sikl tanasi
```

}

1 dan *n* gacha sonlar yig'indisini topish uchun har bir qadamda navbatdagi sonni qo'shib borish uchun *i* o'zgaruvchi e'lon qilamiz.

```
#include <iostream>

using namespace std;

int main() {
	int s = 0, i = 1, n;
	cout<<"n=";
	cin>>n;
	while (i <= n) {
		s += i;
		i++;
	}
	cout<<"s="<<s;
}
```

Dastur kodini bir boshdan qarab chiqamiz. Bizga uchta o'zgaruvchi kerak. Birinchi o'zgaruvchi n soni, ikkinchi o'zgaruvchi sanab borish uchun ishlatiladigan *i* o'zgaruvchisi, uchinchisi yig'indining qiymatini saqlash uchun *s* o'zgaruvchi. Siklni boshlashdan oldin yig'indining qiymatini nolga tenglaymiz, shunda unga qandaydir sonni birinchi marta qo'shganimizda uning o'zi hosil bo'ladi. *i* o'zgaruvchining dastlabki qiymatini 1 ga tenglaymiz, chunki 1 dan boshlab yig'indiga qo'shib borishimiz lozim. Agar $i \le n$ shart bajarilsa u holda *i* ni yig'indiga qo'shamiz(*s+=i bu s=s+i* ning qisqacha yozilishi) va *i* ning qiymatini orqali birga oshiramiz(*i*++ bu inkrement).

Cheksiz sikl.

while yordamida cheksiz sikl hosil qilish uchun shart ifodaga hamisha rost qiymat qabul qiladigan mantiqiy ifoda, o'zgaruvchi yoki rost konstanta qiymatini yozishimiz mumkin.

```
	while (1) {
		cout<<"Cheksiz sikl\n";
	}
```

2) *do while* sikli.

do while sikli while sikliga o'xshash, farqi shart sikl oxirida tekshiriladi va shart bajarilsin yoki bajarilmasin kamida bir marta(1-sikl) sikl bajariladi.

1 dan n gacha sonlar yig'indisi quyidagicha yoziladi:

```
#include <iostream>

using namespace std;

int main() {
     int s = 0, i = 1, n;
     cout<<"n=";
     cin>>n;
     do {
          s += i;
          i++;
     }while (i <= n);
     cout<<"s="<<s;
}
```

Bu siklda i o'zgaruvchinig qiymati qanday bo'lishidan qat'iy nazar sikl bir marta aylanadi. Bu siklni sonni kiritishda unig tog'riligini tekshirish va toki to'g'ri kiritilmaguncha kiritishni davom qildirish uchun foydalanishimiz mumkin. Masalan yuqoridagi masalamizda n soni natural bo'lishi kerak, agar natural son kiritilmasa yana kiritishni so'rash lozim:

```
     do {
          cout<<"n=";
          cin>>n;
     }while (n < 1);
```

3) for sikli.

for sikli sintaksisi quyidagicha:

for(sikl boshlanishidan oldingi amallar**;** sikl davom etish sharti**;** siklning har bir iteratsiyasi oxiridagi amallar) {
    sikl tanasi;
}

Iteratsiya deb siklning bir marta bajarilishiga aytiladi. Agar ma'lum qadam bilan bitta o'zgaruvchining qiymatini o'zgartirib takrorlanuvchi jarayon amalga oshirish lozim bo'lsa, u holda uni quyidagicha xususiy holda yozishimiz mumkin:

for(<o'zgaruvchi tipi> o'zgaruvchi =boshlang'ich qiymat**;** o'zgaruvchi <=oxirgi qiymat**;** o'zgaruvchi +=sikl qadami) {
    sikl tanasi;
}

1 dan *n* gacha sonlar yig'indisini topish uchun quyidagicha sikl amalga oshirishimiz mumkin:

```
#include <iostream>
using namespace std;
int main() {
```

```
    int s = 0, n;
    cout<<"n=";
    cin>>n;
    for (int i = 1; i <= n; i++) {
        s += i;
    }
    cout<<s;
}
```

Bu siklda *i* ning qiymati sikl boshlanishidan avval 1 ga teng qiymatni qabul qiladi. Yana bitta iteratsiya qilish uchun bajarilishi kerak bo'lgan shart i<=n, agar shart rost bo'lsa, yana bitta iteratsiya bajariladi, iteratsiya oxirida *i* ning qiymati birga oshiriladi(*i*++). Keyingi har bir iteratsiyada for siklining ikkinchi va uchinchi qismlari bajariladi, 1-qismi boshqa bajarilmaydi. Eng oxirgi iteratsiyadan oxirida *i* ning qiymati oshirilgach *n*+1 ga teng bo'ladi va keyingi iteratsiyada shart yolg'on qiymat qabul qilganligi sababli (*n*+1<=*n* yolg'on qiymat qabul qiladi) sikl aylanishi tugaydi.

Sikl o'zgaruvchisi *i* haqiqiy son ham bo'la oladi. Masalan 1 dan 10 gacha sonlarni 0.01 qadam bilan chiqarish uchun ya'ni 1, 1.01, 1.02, 1.03, …, 10 sonlarini chiqarish uchun quyidagicha sikl yoziladi.

```
    for (double x = 1; x <= 10; x += 0.01) {
        cout<<x<<" ";
    }
```

x sikl parametri bu safar haqiqiy qiymatni qabul qiladi va har bir iteratsiya oxirida qiymati 0.01 ga oshiriladi.

for siklining uchta qismidan istalgan qismini yozmaslik mumkin:

```
    double x = 1;
    for (; x <= 10; x += 0.01) {
        cout<<x<<" ";
    }
```

bu kod avvalgi yozilgani bilan bir xil, faqat *x*=1 dastlabki qiymatni o'zlashtirish *for* ichida yozilmadi.

```
    double x = 1;
    for (; ; x += 0.01) {
        cout<<x<<" ";
    }
```

Bu kod qismida *x* ning qiymati 1 dan boshlab 0.01 qadam bilan oshirib boriladi, lekin to'xtash sharti yozilmadi, shuning uchun cheksiz sikl hosil bo'ladi.

```
    double x = 1;
    for (; ;) {
        cout<<x<<" ";
    }
```

Bu holatda esa *x* ning qiymati iteratsiya oxirida o'zgartirilmadi shuning uchun cheksiz ko'p marta *x* ning dastlabki qiymati 1 chirariladi.

break operatori.

break operatori siklni uning bajarilish sharti rost qiymat qabul qilishiga qaramasdan to'xtatish uchun qo'llaniladi. Yuqoridagi *x* ning qiymati 1 dan 100 gacha 0.01 qadam bilan oshirib boradigan misolda

```
double x = 1;
for (; ;) {
     if (x > 100.000001)
          break;
     cout<<x<<" ";
     x += 0.01;
}
```

break operatorining ishlatishga misollardan biri berilgan sonning tub yoki tub emasligini aniqlaydigan dastur yozish.

Sonning tub ekanligini aniqlash uchun uni 2 dan $\lfloor\sqrt{n}\rfloor$ gacha bo'lgan sonlarga bo'linishini tekshiramiz. Agar ulardan biriga qoldiqsiz bo'linadigan bo'lsa, u holda bu son tub emas. 103 sonining tub ekanligini aniqlash uchun 2,3,4,5,6,7,8,9 va 10 sonlariga bo'linishini tekshiramiz.

```
#include <iostream>
using namespace std;
int main() {
     int n;
     cin>>n;
     bool is_prime = true;
     for (int i = 2; i*i <= n; i++) {
          if (n % i==0) {
               is_prime = false;
               break;
          }
     }
     if (n==1)
          is_prime = false;
     if (is_prime)
          cout<<"Tub";
     else
          cout<<"Tub emas";
}
```

Dastur kodini taxlil qilib chiqamiz. cin>>n – *n* sonini kiritish. `is_prime` o'zgaruvchisi - berilgan sonning tub ekanligining rost yoki yolg'onligini saqlovchi qiymat. Dastlab sonni tub deb tasavvur qilamiz(`is_prime = true`). 2 dan $\lfloor\sqrt{n}\rfloor$ gacha sonlarni ko'rib chiqish uchun for (int i=2; i<=sqrt(n); i++) ko'rinishida siklni amalga oshirish lozim. i<=sqrt(n) shartning ikkala tamonini kvadratga ko'tarib, uning o'rniga `i*i <= n` shartni yozish mumkin.

n soni *i* ga qoldiqsiz bo'linishi uchun *n* ni *i* ga bo'lgandagi qoldiq qiymati nolga teng bo'lishi kerak(`if (n % i==0)`). Agar bunday shart bajarilsa, u holda tekshiilayotgan son tub emas degan xulosaga kelinadi, ya'ni uning 1 dan kata va o'ziga teng bo'lmagan birorta bo'luvchisi bor. Endi qolgan sonlarga bo'linishini tekshirishning zaruriyati yo'q, siklni to'xtatish mumkin. Berilgan son tub emas degan xulosaga kelamiz(`is_prime = false`) va siklni to'xtatamiz(`break`).

Agar *n*=1 bo'lsa *n* soni 2 dan boshlab hech bir songa bo'linmaydi va `is_prime` true qiymatini saqlab qoladi.Buni alohida tekshirish lozim: agar *n* birga teng bo'lsa u holda u tub emas. Agar berilgan son tub bo'lsa `is_prime` o'zgaruvchisi ***true*** qiymatni saqlab qoladi.

continue operatori.

continue operatori siklni to'xtatmasdan, uni keyingi iteratsiyadan davom qildirib ketish uchun ishlatiladi. Masalan a dan b gacha sonlar yig'indisi va ular ichidan n ga qoldiqsiz bo'linmaydigan sonlar sonini topish dasturini for sikli yordamida quyidagicha yozish mumkin:

```
#include <iostream>
using namespace std;
int main() {
	int a, b, n;
	cin>>a>>b>>n;
	int sum = 0, cnt = 0;
	for (int i = a; i <= b; i++) {
		sum += i;
		if (i % n != 0)
			cnt++;
	}
	cout<<a<<" dan "<<b<<" gacha sonlar yig'indisi:
"<<sum<<endl;
	cout<<n<<" ga bo'linmaydigan sonlar soni: "<<cnt<<endl;
}
```

a dan b gacha barcha sonlarni ko'rib chiqamiz, `sum += i` summaga barcha *i* larni qo'shib boramiz, agar navbatdagi son *i* ga qoldiqsiz bo'linsa `if (i % n != 0)`, *i* ga bo'linadigan sonlar sonini birga oshiramiz(`cnt++`). Siklni **continue** operatori bilan quyidagi shaklda ham yozish mumkin:

```
	for (int i = a; i <= b; i++) {
		sum += i;
		if (i % n==0)
			continue;
		cnt++;
	}
```

Bu shaklda yozilganda `sum += i` hamisha bajariladi. Agar n %i==0 shart bajarilsa u holda siklning navbatdagi iteratsiyasiga o'tiladi. Ya'ni bizga *n* ga bo'linmaydigan

sonlar soni kerak. Agar n %i==0 shart bajarilmasa, u holda sikl tanasining navbatdagi amali ya'ni `cnt++` bajarilib bo'linmaydigan sonlar soni birga oshiriladi.

Topshiriqlar

Topshiqlarni unda ko'rsatilgan sikldan foydalanib yozing.

1-Topshiriq. *while* sikli

1. $S = \frac{x-1}{1} - \frac{(x-1)^2}{2} + \frac{(x-1)^3}{3} - \frac{(x-1)^4}{4} + ... + \frac{(-1)^{n-1}(x-1)^n}{n}$
2. $P = 1 - \frac{x^2}{1} + \frac{x^4}{2} - \frac{x^6}{3} + ... + \frac{(-1)^n x^{2n}}{n}$
3. $S = 1 - x^2 + x^4 - x^6 + ... + (-1)^n \cdot x^{2n}$
4. $SS = x - \frac{x^2}{2} + \frac{x^3}{3} - ... + \frac{(-1)^{n-1} x^{2n-1}}{n}$
5. $S = x + \frac{x^2}{2} + \frac{x^3}{3} + ... + \frac{x^{2n-1}}{n}$
6. $P = x - \frac{x^3}{3} + \frac{x^5}{5} - ... + \frac{(-1)^{n+1} x^{2n-1}}{2n-1}$
7. $S = x + \frac{x^3}{3} + \frac{x^5}{5} + ... + \frac{x^{2n-1}}{(2n-1)}$
8. $PP = 1 + \frac{1}{1} + \frac{1}{2} + \frac{1}{3} + ... + \frac{1}{n}$
9. $S = k + \frac{k^2}{2} + \frac{k^3}{3} + ... + \frac{k^n}{n}$
10. $S = 1 + \frac{x^1}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + ... + \frac{x^n}{n!}$
11. $S = \frac{x}{1!} + \frac{x}{2!} + \frac{x}{3!} + ... + \frac{x}{n!}$
12. $S = \frac{\pi}{10} + \frac{\pi^3}{10^3} + \frac{\pi^5}{10^5} + ... + \frac{\pi^{2n+1}}{10^{2n+1}}$
13. $S = \frac{\pi}{10} + \frac{\pi}{10^3} + \frac{\pi}{10^5} + ... + \frac{\pi}{10^{2n+1}}$

2-Topshiriq. *do while* sikli

1. $y = \sqrt[3]{\frac{\sin ax + b^{2c}}{b^2 + \cos^2 x}} - \frac{\sin x^2}{ab}, \quad c \le x \le d, n = 25$
2. $y = \sqrt[3]{\frac{ax + b}{b^2 + \cos^2 x}} - \frac{\sin x^2}{ab}, \quad a \le x \le c, h = 0.3$
3. $y = \sqrt[3]{a^a} + x^2 \cos ax, \quad -\frac{\pi}{2} \le x \le \pi; h = \frac{\pi}{19}, a = 3.26$
4. $y = \frac{a^2 + bx + x^c}{a^2 + b^2 + x^2}, \quad 5 \le x \le 10, h = 0.4$

5. $y = a^2 \cos x + \frac{\sin x}{2} + bx^2, \quad c \le x \le в, h = 0.2$

6. $y = \sqrt[3]{\frac{\sin ax + b^{2c}}{b^2 + \cos^2 x}} - \frac{\sin x^2}{ab}, \quad -1 < x < 1, n = 50$

7. $y = a^2 + \sqrt[5]{\frac{b + \sin x}{a^3 + \cos^2 x^3}}, \quad 1 \le a \le 12; n = 20$

8. $y = \frac{ax^2}{b} + \frac{x}{c}, 1 \le x \le 10, h = 8, a = 3, b = 12, c = 6$

9. $y = a\cos x - \sin x^2, 0 \le x \le 10, h = 0{,}5$

10. $y = \sqrt[k]{\frac{ax + b}{b^2 + \cos^2 x}} - \frac{\sin x^2}{ab}, \quad d \le x \le c, n = 15$

11. $y = \sqrt{\frac{\sin ax + b^{2c}}{b^2 + \cos^2 x}} - \frac{\sin x^2}{ab}, \quad 0 \le x \le 1, n = 50$

12. $y = \frac{\log a^{2\sin x} + e^{2x}}{\arctan x + 2}, -\pi \le x \le \pi, n = 10$

13. $y = \frac{a^b + b^x + c^a}{2x^2 + 3a^{x+c}}, \quad 3 \le x \le 5, h = 0.2$

14. $y = \frac{ax^2 + bx + 4}{a^2 + b^2 + x^2}, \quad 1 \le x \le 20, n = 100$

15. $y = 2\sqrt[3]{a^{\sin 2x}} + x^2 \cos ax, \quad -\frac{\pi}{2} \le x \le \pi; n = 10$

3-Topshiriq. *for* sikli

1. $S = \frac{\sin 1}{2^1} + \frac{\sin 2}{2^2} + \ldots + \frac{\sin n}{2^n}$
2. $S = \frac{\sin 1^1}{2^1} - \frac{\sin 2^2}{2^2} + \ldots + (-1)^{2n} \frac{\sin n^n}{2^n}$
3. $S = \frac{1}{1!} - \frac{1}{3!} + \frac{1}{5!} - \frac{1}{7!} + \ldots + (-1)^{n-1} \frac{1}{(2n-1)!}$
4. $S = \frac{1}{x^2} - \frac{1}{x^4} + \frac{1}{x^6} - \ldots + (-1)^{2n} \frac{1}{x^{2n}}$
5. $S = \frac{1}{x^2} + \frac{2}{x^4} + \frac{3}{x^6} + \ldots + \frac{n}{x^{2n}}$
6. $S = \sin x - \frac{1}{2}\sin 2x + \frac{1}{3}\sin 3x - \frac{1}{4}\sin 4x + \ldots + (-1)^{2n} \frac{1}{n}\sin nx$
7. $S = \frac{x^1}{\sqrt{1}} + \frac{x^2}{\sqrt{2}} + \ldots + \frac{x^n}{\sqrt{n}}$
8. $S = 1 + \frac{x^1}{1!} + \frac{x^2}{2!} + \ldots + \frac{x^n}{n!}$

9. $S = 1 - \frac{x^1}{1!} + \frac{x^2}{2!} - ... + (-1)^{2n-1} \frac{x^n}{n!}$
10. $S = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + ... + \frac{(-1)^{n-1} x^{2n-1}}{(2n-1)!}$
11. $S = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + ... + \frac{(-1)^{n-1} x^{2n-2}}{(2n-2)!}$
12. $S = 1 + \frac{x^2}{2!} + \frac{x^4}{4!} + \frac{x^6}{6!} + ... + \frac{x^{2n-2}}{(2n-2)!}$
13. $S = x + \frac{x^2}{2} + \frac{x^3}{3} + ... + \frac{x^{2n-1}}{2n-1}$
14. $S = x + \frac{x^3}{3!} + \frac{x^5}{5!} + ... + \frac{x^{2n-1}}{(2n-1)!}$
15. $S = 1 - \frac{k^1}{1!} + \frac{k^2}{2!} - \frac{k^3}{3!} + ... + \frac{(-1)^{2n-1} k^n}{n!}$

Tajriba ish №4

Mavzu: Murakkab takrorlanuvchi algoritmlarni dasturlash

Ishdan maqsad: C++ dasturlash tilida ichma-ich joylashgan sikllarni yozishni o’rganish. Ichma-ich joylashgan sikllarni sindirish va goto operatori.

Nazariy qism.

Murajkkab takrorlanuvchi jarayonlar uchun ichma-ich sikllar yozish talab qilinadi. Ichma-ich sikllarda bir sikl boshqa sikl ichida bo’ladi. Ichma-ich sikllarni jadvallar bilan ishlaganda qo’llash mumkin. Tashqi sikl barcha qatorlar bo’yicha, ichki sikl joriy qatordagi ustunlar bo’yicha iteratsiya qiladi.

Quyidagicha ko’paytirish jadvalidagi sonlarni chiqarish lozim bo’lsin:

```
 1   2   3   4   5   6   7   8   9  10
 2   4   6   8  10  12  14  16  18  20
 3   6   9  12  15  18  21  24  27  30
 4   8  12  16  20  24  28  32  36  40
 5  10  15  20  25  30  35  40  45  50
 6  12  18  24  30  36  42  48  54  60
 7  14  21  28  35  42  49  56  63  70
 8  16  24  32  40  48  56  64  72  80
 9  18  27  36  45  54  63  72  81  90
10  20  30  40  50  60  70  80  90 100
```

Siklni qatorma-qator *i* o'zgaruvchini bilan, har bir qatorda ustun bo'yicha *j* o'zgaruvchi bilan iteratsiya qilamiz. *i*-qator va *j*-ustunida i*j ni chiqaramiz. Har bir qatorni chiqarib bo'lgach, keyingi qatorga o'tamiz:

```
#include <iostream>
#include <stdio.h>
using namespace std;

int main() {
      for (int i = 1; i <= 10; i++) {
            for (int j = 1; j <= 10; j++) {
                  printf("% 4d", i*j);
            }
            cout<<endl;
      }
}
```

Ichma-ich siklga misollar.

Ichma-ich sikl	Chiquvchi ma'lumot	Tushintirish

`for (int i = 1; i <= 3; i++) {`
`for (int j = 1; j <= 4; j++) {`
`cout << "*";`
`}`
`cout << endl; }`
	****
	****
	****
		Har biri 4 ta * dan iborat 3 ta qator chiqaradi

`for (int i = 1; i <= 4 i++) {`
`for (int j = 1; j <= 3; j++) {`
`cout << "*";`
`}`
 	
`cout << endl; }`
	***
	***
	***
	***
		Har biri 3 ta * dan iborat 4 ta qator chiqaradi

`for (int i = 1; i <= 4; i++){`
`for (int j = 1; j <= i; j++) {`
`cout << "*";`
`}`
`cout << endl;}`
	*
	**
	***
	****
		Uzunliklari 1,2,3,4 ga teng bo'lgan 4 ta qator chiqariladi

for (int i = 1; i <= 3; i++)
{
for (int j = 1; j <= 5; j++)
{
if (j % 2 == 0) { cout << "*"; }
else { cout << "-"; }
}
cout << endl;
}
	-*-*-
	-*-*-
	-*-*-
		Juft ustunlarda * chiqaradi, toq ustunlarda - chiqariladi.

for (int i = 1; i <= 3; i++)
{
for (int j = 1; j <= 5; j++)
{
if ((i + j) % 2 == 0) { cout << "*"; }
else { cout << " "; }
}
cout << endl;
}
	* * *
	* *
	* * *
		Shaxmat doskasi

Misol. Ko'paytmalarning umumiy yi'gindisini hisoblang.

$$S=\sum_{i=1}^{32}\prod_{m=1}^{14}\frac{\ln i+tm^i}{rm^i+fn^{2i}}.$$

Bu masalani ko'paytmalarning yi'gindisi deb nomlanadi. *i* o'zgaruvchining har bir qiymatida m o'zgaruvchi 1 dan 14 gacha qiymatlarni barchasini ifodaga qo'yib, ko'paytmasini topiladi. Har bir ko'paytmaning yig'indisi masalaning javobi bo'ladi.

```
#include<iostream>
#include<cmath>
using namespace std;

int main(){
    float t,r,f;
    int i,m,n;
    cout<<"t=";
    cin>>t;
    cout<<"r=";
    cin>>r;
    cout<<"f=";
    cin>>f;
    cout<<"n=";
    cin>>n;
    double s=0;
    for(i=1;i<=32;i++){
      double p=1;
      for(m=1;m<=14;m++){
       p *= (log(i)+t*pow(m, 1.0*i))/(r * pow(m, 1.0*i)+f * pow(n,
2.0*i));
      }
      s=s+p;
    }
    cout<<"s="<<s<<endl;
    return 0;
    }
```

Murakkab siklni sindirish.

Masala. To'g'ri burchakli uchburchakning barcha tomonlari uzunliklari natural sonlar. Uning gipatenuzasi *c* berilgan. Uning katetlarini topish lozim. To'gri burchakli uchburchak shartini qanoatlantiradigan istalgan javoblardan birini topish yetarli. ($1\le c\le 10^3$).

Yechimi. $c\le 1000$ bo'lganligi uchun uchburchak katetlari uzunliklari ham 1000 dan oshmaydi. Katetlarni mumkin bo'lgan barcha variantlarini tekshirib chiqish uchun *a* katetni *i* paramertli sikl sifatida qaraymiz, *b* katetni *i* sikl ichida *j* parametrli sikl sifatida to'g'ri burchakli uchburchak bo'lish shartini qanoatlantirilishini tekshirish lozim($i^2+j^2=c^2$):

```
#include <iostream>
using namespace std;
int main() {
	int c;
	cout<<"c=";
	cin>>c;
	bool found = false;
	int a = 0, b = 0;
	for (int i = 1; i <= 1000; i++) {
		for (int j = 1; j <= 1000; j++) {
			if (i*i + j*j==c*c) {
				a = i;
				b = j;
				found = true;
			}
		}
	}
	if (found)
		cout<<a<<" "<<b;
	else
		cout<<"Bunday sonlar yo'q";
}
```

Dastur yechimin tahlil qilib chiqamiz. *c*-gipotenuza uzunligini kiritamiz. `found` mantiqiy o'zgaruvchi e'lon qilamiz, bu o'zgaruvchida bunday gipotenuzali to'gri burchakli uchburchak mavjud yoki yo'qligi haqidagi mantiqiy ma'lumotni saqlaymiz. Dastlab bunday uchburchak mavjud emas deb tasavvur qilamiz(`found = false`). *a* katet sifatida *i* o'zgaruvchi bilan 1 dan 1000 gacha bo'lgan(1000 ning o'rniga *c* gacha tekshirish ham mumkin) iteratsiya qilamiz, bu sikl ichida *b* katet sifatida *j* o'zgaruvchi bilan iteratsiya aylantiramiz. Agar *i*, *j* va *c* sonlari to'g'ri burchakli uchburchak shartini qanoatlantirsa(`if (i*i + j*j==c*c)`) , u holda *a* katet sifatida *i* o'zgaruvchini, *b* katet sifatida *j* o'zgaruvchini qiymatlarini o'zlashtiramiz (`a = i;b = j;`) va topilganligi haqidagi ma'lumotni saqlaydigan mantiqiy o'zgaruvchiga rost qiymatini o'zlashtiramiz(`found = true;`).

Lekin bu ichma-ich sikl barcha variantlarni ko'rib chiqadi. Bizga faqat bitta variant kerak bo'lganligi sababli agar katetlar topilsa, siklni to'xtatishimiz foydaliroq bo'ladi. Siklni to'xtatish uchun quyidagicha yozish mumkin:

```
for (int i = 1; i <= 1000; i++) {
	for (int j = 1; j <= 1000; j++) {
		if (i*i + j*j==c*c) {
			a = i;
			b = j;
			found = true;
			break;
		}
	}
}
```

Bu shaklda siklni sindiradigan bo'lsak faqat ichki sikl sinadi, tashqi sikl keyingi iteratsiyadan davom qiladi. Siklni butunlay sindirish uchun har bir tashqi sikl iteratsiyasidan oldin tekshiramiz, agar javob topilgan bo'lsa u holda tashqi siklni sindiramiz.

```
for (int i = 1; i <= 1000; i++) {
	if (found)
		break;
	for (int j = 1; j <= 1000; j++) {
		if (i*i + j*j==c*c) {
			a = i;
			b = j;
			found = true;
			break;
		}
	}
}
```

goto operatori.

Agar ichma-ich joylashgan sikllar soni ko'p bo'ladigan bo'lsa u holda siklni sindirish uchun har bir siklda `break` operatorini yozish lozim bo'ladi. Buning o'rniga sikldan keyingi amallar bajarilishiga o'tish uchun **goto** operatoridan foydalanish mumkin. **goto** operatoridan keyin nishon yoziladi. Agar **goto** operatori yozilsa, dastur nishon yozilgan nuqtadan bajarishni davom qildiradi. Nishon harflar, $(dollor belgisi), _(pastki chiziqcha), raqamlardan iborat bo'ladi va raqam bilan boshlanmaydi.

```
#include <iostream>
using namespace std;
int main() {
	int c;
	cout<<"c=";
	cin>>c;
	bool found = false;
	int a = 0, b = 0;
	for (int i = 1; i <= 1000; i++) {
		for (int j = 1; j <= 1000; j++) {
			if (i*i + j*j==c*c) {
				a = i;
				b = j;
				found = true;
				goto label_1;
			}
		}
	}
	label_1:
	if (found)
		cout<<a<<" "<<b;
	else
		cout<<"Bunday sonlar yo'q";
}
```

Topshiriqlar

1. Variant

a. $S=\sum_{m=1}^{19}\frac{3m^3+4m+5}{m^3+\ln(m-3)}$

b. $P=\prod_{k=1}^{46}\frac{k}{k^3+7k+5}.$

c. $S=\sum_{i=1}^{32}\prod_{m=1}^{14}\frac{\ln i+m^i}{m^i+n^{2i}}.$

2. Variant

a. $S=\sum_{a=1}^{27}\frac{a^2+2a}{a^3+a\cos^2 a+1}.$

b. $P=\prod_{i=1}^{20}\frac{i^2+1}{\sqrt[i]{i^3}+2}.$

c. $S=\sum_{i=1}^{17}\prod_{k>12}^{28}\ln\frac{k^i+\sqrt[i]{k}}{k^3+\sqrt[k]{i}}.$

3. Variant

a. $S=\sum_{x=5}^{10}(ax+b)^2.$

b. $P=\prod_{a=10}^{15}\frac{a+b}{\sqrt{a^2+x^2}}.$

c. $S=\sum_{k=1}^{5}\sum_{y=2}^{7}\frac{ak+by}{\sqrt{k^2+y^2}}.$

4. Variant

a) $P=\sum_{i=1}^{19}\frac{i^4+i^2+3}{\sqrt{i^1+e^i}}.$

b) $S=\sum_{k=1}^{20}\frac{k+1}{k^3+5k+7}.$

c) $S=\sum_{m=1}^{25}\prod_{n=1}^{30}\sqrt{\frac{|m^n-n^m|}{m^n+n^m}}.$

5. Variant

$a)\ P=\prod_{i=1}^{34}\frac{i^3+|i-9|}{\ln i+7i}.$

$b)\ S=\sum_{k=1}^{10}\frac{(-1)^k\cdot(k+1)}{k^3+k^2+1}.$

$c)\ P=\prod_{n=1}^{34}\sum_{m=1}^{35}(-1)^m\frac{\lg(m+5)}{m^{n+3}+n\cdot m}.$

6. Variant

$a.\ S=\sum_{n=1}^{10}\frac{1}{5-17n+n^3}.$

$b.\ P=\prod_{m=0}^{12}\cdot\frac{\sqrt[2]{|m-5|+1}}{m^3+4m+(-1)^3}.$

$c.\ S=\sum_{ii=1}^{33}\prod_{k=1}^{15}(-1)^i\frac{\sqrt[7]{\sin k+e^k}}{|4i^3-k^4|}.$

7. Variant

$a.\ S=\sum_{a=5}^{8}\frac{ax+bc}{a^2+x}.$

$b.\ P=\prod_{x=2}^{5}\frac{a+bx}{a^2+x^2}.$

$c.\ SP=\sum_{k=1}^{4}\prod_{a=2}^{6}\frac{ak+bx}{k^2+x^2}.$

8. Variant

$a.\ S=\sum_{k=2}^{8}(ak+bx).$

$b.\ P=\prod_{a=3}^{7}\frac{ax+b}{\sqrt{a^2+b^2}}.$

$c.\ SP=\sum_{x=1}^{5}\prod_{a=2}^{4}\frac{ax+b}{\sqrt{a^2+bx}}.$

9. Variant

$a.\ S=\sum_{a=1}^{7}\frac{ax+b}{\sqrt{a+b}}.$

$b.\ P=\prod_{x=2}^{6}\frac{ax^2+b}{\sin(ax)}.$

$c.\ PP=\prod_{x=2}^{5}\prod_{i=1}^{4}\frac{(a\cdot i+bx)}{\sqrt{(ax+b)^i}}.$

10. Variant

a. $S=\sum_{i=10}^{30}\sqrt{(ax+b)^i}$.

b. $P=\prod_{k=1}^{15}\frac{\sin^k(a+b)+3a}{\cos(ak)+2{,}78b}$.

c. $SP=\sum_{i=1}^{10}\prod_{k=1}^{5}\frac{ax^k+i\cdot b}{a\cdot i+b\cdot k}$.

11. Variant

a. $S=\sum_{x=5}^{10}\frac{kx+b^2}{\sqrt{x^2+ab}}$.

b. $P=\prod_{a=1}^{6}\frac{ax+c}{\sin^2 ax}$.

c. $PS=\prod_{k=1}^{4}\sum_{x=2}^{5}\frac{ax+kb}{\sqrt{x^2+k^2}}$.

12. Variant

a. $S=\sum_{k=1}^{5}(ax^2+bk)$.

b. $P=\prod_{a=3}^{6}(ax+\cos^2 ab)$.

c. $PS=\prod_{x=1}^{4}\sum_{a=3}^{6}\frac{ax+bk^2}{a^2+x^2}$.

13. Variant

a. $S=\sum_{x=2}^{12}\frac{ax^2+b}{\cos^2(a+bx)}$.

b. $P=\prod_{a=3}^{9}\sqrt[3]{\frac{ax^2+b}{2\cdot a}}$.

c. $S=\sum_{a=1}^{5}\sum_{x=2}^{7}\frac{a\cdot\cos^2(2x)}{b+c\cdot\sin(ax)}$.

14. Variant

a. $S=\sum_{i=1}^{81}\frac{i+2i+7}{i^2-1+\cos^2 i}$.

b. $P=\prod_{k=1}^{28}\sin\frac{k+1}{k+\sqrt{k+1}}$.

c. $S=\sum_{k=1}^{15}\prod_{n=8}^{21}\frac{n^k+k^n}{\sqrt[n]{R^k}+(n\cdot k)^5}$.

Tajriba ish №3

Mavzu: Funksiyalar yaratish

Ishdan maqsad: C++ dasturlash tilining sikl operatorlari bilan tanishish. Sharti oldindan, shart oxiridan beriladigan takrorlanish va parametrik sikl operatorlarini o’rganish orqali funksiyalarni tashkil etsh.

Nazariy qism.

Funksiya va uning turlari

Funksiya – yordamchi qism dastur bo‘lib, maxsus biror-bir jarayonni amalga oshirishga mo‘ljallangan bo‘ladi. Quyida funksiyani tuzilishi keltirilgan:

Type nomi (parameter 1, parameter 2, ...) {amallar}

- type – e’lon qilinayotgan funksiyani toyifasi;
- nomi – e’lon qilinayotgan funksiyani nomi;
- parameter – funksiyaga qo‘yilgan vazifani amalga oshirishda qatnashuvchi o‘zgaruvchilarni toyifasi bilan birga e’lon qilishni ta’minlaydi;
- amallar – bu qism funksiyaning tanasi bo‘lib, funksiyaga qo‘yilgan vazifani bajaruvchi amallardan iborat.

Funksiyalar xususiyatlari asosan ikki turga bo‘linadi.

Funksiya 1.1

Derektivalar va asosiy funksiya **main()** tarkibida ishlatiladi hamda quyidagicha:

```
#include<iostream>
#include<conio.h>
 using namespace std;
        int summ(int a, int b)  // funksiyani e’lon qilish
{               // funksiyani boshlanishi
    .....       // funksiya tanasi
 };             // funksiyani yopilishi
main ()
{   .....
    summ(a,b);      // funksiyani chaqirish
   .....
```

}

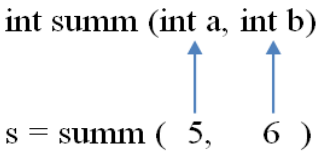
Quyida funksiyani shaklantirishning ikki xil varianti keltirilgan:

1-variant	***Natija:***

#include<iostream>
#include<conio.h>
using namespace std;
int summ(int a, int b) // *funksiyani e'lon qilish*
{ int r;
r = a + b;
return (r); };
 	
int main()
{ int s;
s = summ(5, 6); // *funksiyani chaqirib, ishga tushirish*
cout<<"Summ = "<<s; // *Natijani ekranga chiqarish*
getch();
return 0;
}
	a = 5
	b = 6
	Summ = 11

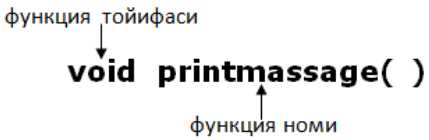
SHu o‘rinda funksiyani e’lon qilishdagi uning tarkibiy qismlariga to‘xtalib o‘tsak:

YUqorida keltirilgan misoldan ko‘rinib turibdiki, *summ* deb ataluvchi funksiyani e’lon qilindi va dasturning asosiy funksiyasi bo‘lgan *main* tarkibida ishga tushirilib natijada hisoblashni oshirish orqali qiymatni ekranga chiqardi. Guvohi bo‘lganimizdek, *summ* funksiya *int* – toyifasidagi parametrlari ustida qo‘shish amalini bajardi. Quyida funksiya o‘zi va uni dastur tarkida *s* orqali hisoblashni amalga oshirgan satrlarni keltiramiz:



Funksiya 1.2

Bu turdagi funksiyalar toyifasi bo‘sh bo‘lib, toyifa o‘rnida *void* – dan foydalanadi. SHu bilan birga funksiya 1 kabi qiymat qaytarmaydi. Funksiya 1.2 ning vazifasi jarayonni borishini ta’minlaydi. Ushbu funksiya 2 e’lon qilish quyida keltirilgan:



Misol:	**Natija:**

#include<iostream>
#include<conio.h>
using namespace std;
void printmessage ()
{ cout<<“Bu void bilan ishlovchi funksiya!”; }
int main()
{
printmessage();
getch();
return 0; }
	Bu void bilan ishlovchi funksiya!

Albatta, dasturni ishlab chiqish mobaynida dasturchi bir nechta funksiyalardan foydalanishimiz mumkin. YUqorida eng sodda funksiyaning tuzilishini foydalanuvchilar e’tiboriga havola qildik. Keltirilganlardan foydalanib, dasturlar tarkibida foydalanilishi lozim bo‘lgan funksiyalarni ishlab chiqish maqsadga muvofiq bo‘ladi.

Funksiya 2

YUqorida funksiyaning umummiy ko‘rinishi bilan tanishib chiqdik. Ularda qiymat kiritilib, funksiyaga qanday amal yuklangan bo‘lsa, shu amalni bajaradi. Buni yuqoridagi misollardagi a va b o‘zgaruvchilarga mos ravishda 5 va 6 qiymatlarni berish orqali funksiya o‘z funksiyasini bajardi. Biroq *a* va *b* o‘zgaruvchilarni boshqa modifikatsiyalarini yuqorida keltirilgan funksiyalar orqali bajarib bo‘lmadi. SHuning uchun funksiya ichida o‘zgaruvchilar manipulyasiyasi amalga oshirilib, tashqaridan kiritilgan qiymatlar o‘zgargan holda ekranga chiqadi. Bunda eng avvalo funksiya nomi e’lon qilinadi, so‘ngra har parametrlarini toyifasi bilan birga ampersand (&) belgisi qo‘yiladi. &-belgini vazifasi argumentni mos ravishda ko‘rsatilgan yoki tavsiya etilgan o‘zgaruvchi bilan o‘rnini almashtirishdan iborat. O‘zgaruvchilarni funksiya nusxasini emas, balki, natijada ularni boshqa modifikatsiyasini beradi. Bu holda dasturni asosiy tarkibida funksiya chaqirilganda foydalanuvchi tomonidan kiritilgan o‘zgaruvchilar o‘ziga o‘xshash kabi argumentni natija sifatida chiqarib beradi. Ushbu jarayon quyidagi misolda keltirilgan.

Misol:	**Natija:**

#include<iostream>
#include<conio.h>
using namespace std;
void nusxakochirish (int& a, int& b, int& c)
{
a* = 2;
b* = 2;
c* = 2;
}
int main()
{
int x=1, y=3, z=7;
	X = 2
	Y = 6
	Z = 14

nusxakochirish(x, y, z);
cout<<"\n X = "<<x;
cout<<"\n Y = "<<y;
cout<<"\n Z = "<<z;
 	
getch();
return 0;
}	

YAna bir shunday misol keltiramiz, bu misol & belgisining vazifasini yanada yaqqolroq ko‘rish mumkin:

Misol:	**Natija:**

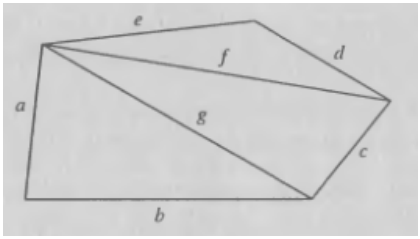
#include<iostream>
#include<conio.h>
using namespace std;
void prev_next (int x, int& prev, int& next)
{
prev = x - 1;
next = x + 1;
}
int main()
{
int x=100, y, z;
prev_next (x, y, z);
cout<<"\n Y = "<<y;
cout<<"\n X = "<<x;
cout<<"\n Z = "<<z;
	Y = 99
	X = 100
	Z = 101

getch();
return 0;
}	

2.2-variant. Funksiyaga doir variantlar

1. Sonni o‘nlik sanoq sistemasidan o‘n oltilik sanoq sistemasiga (hamda teskarisiga) o‘tkazuvchi funksiya tuzing.
2. Sonni o‘nlik sanoq sistemasidan sakkizlik oltilik sanoq sistemasiga o‘tkazuvchi funksiya tuzing.
3. Sonni o‘nlik sanoq sistemasidan ikkilik sanoq sistemasiga o‘tkazuvchi funksiya tuzing.
4. Sonni o‘n oltilik sanoq sistemasidan ikkilik sanoq sistemasiga o‘tkazuvchi funksiya tuzing.
5. Sonni o‘n oltilik sanoq sistemasidan sakkizlik sanoq sistemasiga o‘tkazuvchi funksiya tuzing.
6. Sonni sakkizlik sanoq sistemasidan ikkilik sanoq sistemasiga o‘tkazuvchi funksiya tuzing.
7. Q sonini P darajasini topuvchi funksiya tuzing.
8. P sonini oxiridan L sonini qo‘shuvchi funksiya tuzing.
9. Berilgan to‘g‘ri burchakli uchburchakning katetlari yordamida gipotenuzasini topuvchi funksiya tuzing
10. Berilgan ikki nuqtaning koordinatasi asosida ular orasidagi masofani topuvchi funksiya tuzing.
11. Berilgan sondagi qo‘shni raqamlarining raqamlarining yig‘indisiga teng bo‘lgan raqamni o‘chirivchi funksiya tuzing.
12. Berilgan N soni ikkita tub sonni yig‘indisi bo‘lishini tekshiruvchi funksiya tuzing.
13. Berilgan son 11 ga bo‘linsa 1 aks holda o chiqaruvchi funksiya tuzing.
14. Berilgan sonni K o‘rindagi raqamini N o‘rindagi raqami bilan almashtiruvchi funksiya tuzing.

15. Sonni raqamlarini o‘sish (kamayish) tartibida saralovchi funksiya tuzing.
16. Uchburchakni uchta uchining koordinatalari berilgan. Uning yuzasini topuvchi funksiya tuzing.
17. Tomonlari a, b, c, d va e haqiqiy sonlardan iborat bo‘lgan beshburchak berilgan (2.1-rasm). Beshburchakni va undagi uchta uchburchaklarni yuzasini topuvchi alohida funksiyalar tuzing.



2.1-rasm.

18. Besh burchakning uchlarining koordinatalari x1, y1, x2, y2, …, x5, y5 berilgan (2.1-rasm). Beshburchakdagi uchburchakni uchta uchining koordinatalari berilgan. Uning yuzasini topuvchi funksiya tuzing.
19. Sonni tublikka tekshiruvchi funksiya tuzing va barcha uch xonali tub sonlarni chiqaring.
20. Barcha tub sonlar ichidan “egizak”larini topuvchi funksiya tuzing. Tub sonlar “egizak” deyiladi, agarda ular 3 ga farq qilsa. Masalan, 41 va 43. Uch xonali sonlar ichidan barcha “egizak”larni chiqaring.
21. Ikkita son berilgan. Ularni raqamlari yig‘indisi kattasini toping. Sonni raqamlari yig‘indisini topuvchi funksiya tuzing.
22. Ikkita son berilgan. Ularning raqamlaridan soni ko‘pini toping. Sonni raqamlari sonini topuvchi funksiya tuzing.
23. 6 xonali barcha baxtli sonni toping. Agarda 6 xonali sonni dastlabki uchta raqamini yig‘indisi oxirgi uchta raqamini yig‘indisiga teng bo‘lsa baxtli son deyiladi. 6 xonali sonni raqamlarini uchtalab yig‘indisini topuvchi funksiya tuzing.

24. Berilgan sonni palindromlikka tekshiruvchi funksiya tuzing. Masalan, 1221. Son palindrom bo‘lsa “Palindrom”, aks holda “Palindrom emas” so‘zini chiqaring.
25. Berilgan a va b sonlarini EKUBini topuvchi funksiya tuzing.
26. Berilgan a va b sonlarini EKUKini topuvchi funksiya tuzing.
27. Berilgan a, b va c sonlarini EKUBini topuvchi funksiya tuzing.
28. So‘z berilgan. Shu so‘zni palindromlikka tekshiruvchi funksiya tuzing. Masalan, kiyik. So‘z palindrom bo‘lsa “Palindrom” aks holda “Palindrom emas” so‘zini chiqaring.
29. Berilgan so‘zning harflari sonini topuvchi funksiya tuzing.
30. Berilgan y (yil), o (oy) va k (kun) sonlari berilgan. Shu sonlarni kiritgan holda qaysi kunga to‘g‘ri kelishini aniqlovchi dastur tuzing. Masalan, 2014 2 7 uchun “Juma” chiqishi lozim.

Rekursiv funksiyalar

Funksiya tanasida o‘zini o‘zi chaqirsa **rekursiya** deyiladi. Rekursiya ikki xil bo‘ladi:

- **Oddiy** – agar funksiya o‘z tanasida o‘zini chaqirsa;
- **Vositali** – agar birinchi funksiya ikkinchi funksiyani chaqirsa, ikkinchisi esa o‘z navbatida birinchi funksiyani chaqirsa.

Masalan: Faktorialni hisoblash funksiyasini olamiz. U o‘zini ichida oldingilarini chaqiradi.

Dasturi	Matematik ifodasi

`long Faktorial(int n)`
`{`
`if (!n) return 1;`
`else return n * Faktorial (n - 1);`
`}`
	$n! = \begin{cases} 1, & \text{agar } n = 0; \\ n^* (n-1)!, & \text{agar } n > 0, \end{cases}$

Xuddi shunday darajani hisoblash funksiyasini ham misol keltirish mumkin.

Dasturi	Matematik ifodasi

double Daraja(double x, int n)
{
if (!n) return 1;
else return x * Daraja(x, n - 1);
}
	$x^n = \begin{cases} 1, & \text{agar } n = 0; \\ x * x^{(n-1)}, & \text{agar } n > 0, \end{cases}$

Namuna. Rekursiv funksiyadan foydalangan holda ikkita sondan raqamlari yig‘indisi katta bo‘lgan sonni topuvchi dastur tuzing.

```
    int sum, sum_1, sum_2 ;
    int raqam(int son)
    {
        sum  += son % 10;
        son = son / 10;
        if (son == 0) return sum;
        raqam (son);
        }
    int main()
    {
        int  sum_1 = 0, sum_2 = 0;
        int son_1, son_2;
        cin>>son_1>>son_2;
        sum_1 = raqam(son_1);
        sum_2 = raqam(son_2);
        if (sum_1 > sum_2) cout << son_1; else
cout<<son_2;
    getch();
    return 0;
    }
```

2.3-variant. Rekursiyaga doir variantlar

1. Nyuton Binom koeffisiyentlarini hisoblovchi rekursiv funksiyali dastur tuzing (M004).

Kiruvchi ma’lumotlar: Nyuton Binom koeffisiyenti n berilgan ($1 \le n \le 100$).

Chiquvchi ma’lumotlar: Nyuton Binom koeffisiyentlarini probel bilan ajratgan holda tartib bilan chiqaring.

Kiritishga misol	Chiqarishga misol

1	1 1
2	1 2 1
3	1 3 3 1

2. n natural sonini a-darajasini aniqlovchi rekursiv funksiya tuzing.
3. Sonni raqamlar yig‘indisini topuvchi rekursiv funksiya tuzing.
4. Sonni raqamlar sonini topuvchi rekursiv funksiya tuzing.
5. Berilgan sonning raqamli ildizini topuvchi rekursiv funksiya tuzing. Sonning raqamlar yig‘indisini topamiz va bu yig‘indini ham raqamlar yig‘indisi ustma-ust tushsa sonni raqamli ildizi deb ataladi.
6. Arifmetik progressiyani birinchi hadi va ayirmasi berilgan. N ta hadini aniqlovchi rekursiv funksiya tuzing.
7. Arifmetik progressiyani birinchi hadi va ayirmasi berilgan. N ta hadini yig‘indisini hisoblovchi rekursiv funksiya tuzing.
8. Fibonachi sonini k-hadini topuvchi rekursiv funksiya tuzing.
9. Fibonachi sonini k ta hadini yig‘indisini hisoblovchi rekursiv funksiya tuzing.
10. Massivning eng katta elementini topuvchi rekursiv funksiya tuzing.
11. Massivning eng katta elementini indeksini topuvchi rekursiv funksiya tuzing.
12. Manfiy bo‘lmagan n va m sonlari uchun Akkerman funksiyasini hisoblovchi rekursiv funksiya tuzing. Akkerman funksiyasi quyidagicha aniqlanadi:

    m+1, agar n = 0;

    A(n, m) = A(n – 1, 1), agar n ≠ 0, m = 0;

    A(n-1, A(n, m-1)), agar n>0, m>0.
13. Berilgan a va b sonlarining EKUBini topuvchi rekursiv funksiya tuzing.
14. Berilgan a va b sonlarining EKUKini topuvchi rekursiv funksiya tuzing.
15. Berilgan a, b va c sonlarining EKUBini topuvchi rekursiv funksiya tuzing.

16. Berilgan sonning raqamlarini teskarisiga yozuvchi rekursiv funksiya tuzing.
17. Soni noma’lum bo‘lgan sonlar ketma-ketligini massiv ishlatmagan holda teskarisiga yozuvchi rekursiv funksiya tuzing.
18. N-Fibonachi sonini oxirgi 17 xonasini aniqlovchi rekursiv dastur tuzing.
19. Berilgan S satrning i-elementidan j-elementigacha bo‘lgan elementlarini simmetrikligini aniqlovchi rekursiv dastur tuzing.
20. Maxraji n bo‘lgan [0;1] orasidagi barcha qisqarmas kasrlarni topuvchi rekursiv dastur tuzing.
21. Berilgan o‘nlik natural sonni N ($2 \leq N \leq 16$) sanoq sistemasiga o‘tkazuvchi rekursiv funksiya tuzing.
22. Berilgan sonni necha N faktorialga tengligini aniqlovchi rekursiv funksiya tuzing. Masalan, 6 = 3!.
23. Sonni o‘nlik sanoq sistemasidan o‘n oltilik sanoq sistemasiga (hamda teskarisi) o‘tkazuvchi rekursiv funksiya tuzing.
24. Sonni o‘nlik sanoq sistemasidan sakkizlik oltilik sanoq sistemasiga o‘tkazuvchi rekursiv funksiya tuzing.
25. Sonni o‘nlik sanoq sistemasidan ikkilik sanoq sistemasiga o‘tkazuvchi rekursiv funksiya tuzing.
26. Sonni o‘n oltilik sanoq sistemasidan ikkilik sanoq sistemasiga o‘tkazuvchi rekursiv funksiya tuzing.
27. Sonni o‘n oltilik sanoq sistemasidan sakkizlik sanoq sistemasiga o‘tkazuvchi rekursiv funksiya tuzing.
28. Berilgan massiv elementlarini saralovchi rekursiv funksiya tuzing. Elementlar soni [1; 10000] bo‘lishi mumkin.
29. 0110100110010110 tartibda berilgan ketma – ketlikning n – o‘rnida necha soni turganligini topuvchi dastur tuzing. Bu ketma – ketlik birinchi elementi 0 ga teng. Keyingi elementlari esa berilgan satrni 0 ni 1 ga 1 ni esa 0 ga o‘girgan holatga ko‘chirilgan ya’ni 0 1 10 1001 10010110

30. $2^{2\wedge n}$ ning 10^9 ga bo‘lgandagi qoldiqni hisoblovchi rekursiv funksiya tuzing. n soni [1; 1000000] oraliqda bo‘lishi mumkin.

Nazorat uchun savollar

1. Ko‘p o‘lchovli massivlar.
2. Funksiya nima?
3. Rekursiv funksiya nima?
4. Rekursiv funksiya va funksiyaning farqi nima?
5. Massivlar tasodifiy sonlar bilan qanday to‘ldiriladi?
6. Funksiya argumenti sifatida massivlarni ishlatish mumkinmi?

3.2 Satr

C++ tilida satr uchun maxsus tur aniqlanmagan. Satr char turidagi belgilar massivi sifatida qaraladi va bu belgilar ketma–ketligi satr terminatori deb nomlanuvchi nol kodli belgi bilan tugaydi (‘\0’).

Odatda, nol-terminator bilan tugaydigan satrlarni ASCII–satrlar deyiladi. Sart konstanta deb qo‘shtirnoqlar ichiga olingan belgilar ketma–ketligiga aytiladi:

```
“Ushbu belgilar ketma-ketligiga satr deyiladi.”
```

Satrli massivlar quyidagicha e’lon qilinadi:

```
char A[4] = { 'A', 'P', '[', 'D'};
char B[10];
```

Satr massivi e’lon qilinishida, satr oxiriga terminator qo‘yilishini va natijada satrga qo‘shimcha bitta bayt bo‘lishi inobatga olinishi kerak:

Massivning har bir simvol – alohida obyekt va massiv N uzunlikka ega.

Satrlar bilan ishlash uchun albatta `#include <string.h>` bibliotekasidan foydalanish lozim.

Masala. Berilgan matndagi barcha “a” harfini “b” ga almashtiruvchi dastur tuzing.

```
main()
{
  char q[80];//satrli massivni e’lon qilish
  int i;
  cout<<“Satr kiriting\n“;
  cin>>q;//Kiritilgan satrni o‘qib olish
  i = 0;// sikl boshi
  while ( q[i] != '\0' ) { //toki satr oxirigacha
    if ( q[i] == 'a' ) q[i] = ‘b'; // almashtirsin
    i ++; // siklni oshirish
    }
  cout<< “Natija: “<< q ;
}
```

Bu dasturning kamchilik joyi shundaki, u faqatgina bitta so‘zdagi a larni b ga almashtiradi, ya’ni probelgacha. Probellari bilan o‘qib olish uchun `gets(q);` operatoridan foydalanish lozim. Bunda `cin>>q;` ni `gets(q);` ga almashtirish lozim.

Sartlar bilan ishlovchi funksiyalar:

puts (q);	natijani bitta satrga chiqarish va keyingisiga o‘tkazadi;
strlen (q);	satr uzunligini aniqlaydi;
strcmp (q1, q2);	**q1** va **q2** satrlarni bir biridan farq qiluvchi kodlar orasidagi farqni aniqlaydi;
strcpy (q1, q2);	q2 satrni q1 satrga nusxalaydi. Bu funksiya strcpy (q1+2, q2), strcpy (q1+2, q2+3) va boshqa ko‘rinishda bo‘lishi mumkin.
strncpy (q1+2, q2, 2);	bir nechta simvollarni nusxalash
***strcat* (q1, q2);**	birinchi satr (q1) oxiridan ikkinchi satr (q2) ni qo‘shadi
strchr(q, 'b')	satr boshidan bitta simvolni topish
strstr(q, "bcd");	satr boshidan birinchi qism satrni topish

3.2- variant. Satrlar bo‘yicha variantlar

1. Matnli faylda nechta so‘z va gap qatnashayotganligini aniqlovchi dastur tuzing.

Kiruvchi	**Chiquvchi**
Bugun burunchi para C++ darsi. 2-para Matematika.	7 2
Toshkent O‘zbekiston poytaxti.	3 1
Mening Malibu rusumli mashinam bor edi. Uni sotdim.	8 2

2. Matnli faylda nechta “a” va “.” qatnashayotganligini aniqlovchi dastur tuzing.

Kiruvchi	**Chiquvchi**
Mana senga olam olam gul.	5 1
.Toshkent O‘zbekiston poytaxti.	1 2
Mening Malibu rusumli mashinam bor.	3 1

3. Berilgan gapdagi simvollar soni n tadan kam bo‘lgan so‘zlarni o‘chiring.

Kiruvchi	Chiquvchi
5 Mana senga olam olam gul.	senga
8 Toshkent O‘zbekiston poytaxti.	O‘zbekiston
7 Mening Malibu rusumli mashinam bor.	rusumli mashinam

4. Matnli satrda katta harflar bilan berilgan satr berilgan. Undagi qism satr necha marta qatnashayotganligini aniqlovchi dastur tuzing.

Kiruvchi	Chiquvchi
DEF DEFDEFABCABCZABCDEFDEF	4

5. Matnli fayldagi simvollar ketma-ketligi bir nechta probellar bilan ajratilgan holda berilgan. Undagi eng uzun so‘zning “a” harflarini “b” bilan almashtiring.

Kiruvchi	Chiquvchi
Mana senga olam olam gul!	Mana sengb olam olam gul!

6. Matnli fayldagi simvollar ketma-ketligi bir nechta probellar bilan ajratilgan holda berilgan. Undagi eng uzun va eng kalta so‘zlarni sonini probel bilan chiqaring. Masalan, “hi! say biggest” uchun natija “1 2”

Kiruvchi	Chiquvchi
Mana senga olam olam gul!	1 1

7. Matnli fayldan ortiqcha probellarni olib tashlovchi dastur tuzing. Har bir so‘z orasida bitta probel tursin.

Kiruvchi	Chiquvchi
Mana senga olam olam gul!	Mana senga olam olam gul!

8. Matnli fayldagi gaplarni birinchini ikkinchiga, uchinchini to‘rtinchiga va hokazo so‘zlarini o‘zgartiruvchi dastur tuzing.

Kiruvchi	Chiquvchi
Mana senga olam olam gul! Havo yaxshi.	senga Mana olam olam gul! yaxshi Havo.

9. Matnli fayldagi birinchini ikkinchiga, uchinchini to‘rtinchiga va hokazo gaplarni o‘rnini o‘zgartiruvchi dastur tuzing.

Kiruvchi	Chiquvchi
Mana senga olam olam gul! Havo yaxshi.	Havo yaxshi. Mana senga olam olam gul!

10. Matndagi eng ko‘p qatnashgan harfni topuvchi dastur tuzing. Sonini va harflarni matnda berilgan tartib bo‘yicha chiqaring.

Kiruvchi	Chiquvchi
Mana senga olam olam gul! Havo yaxshi.	`7 a`
Bugun imtixon. Demak, bayram ekan.	`4 a`

11. Matndagi eng kam qatnashgan harflarni topuvchi dastur tuzing. Sonini va harflarni matnda berilgan tartib bo‘yicha chiqaring.

Kiruvchi	Chiquvchi
Mana senga olam olam gul! Havo yaxshi.	`1 n i x y`
Bugun imtixon. Demak, bayram ekan.	`1 g t x o D y r`

12. So‘z berilgan. Shu so‘zni palindromlikka tekshiring. Boshidan va teskarisiga o‘qilganda bir xil bo‘lsa bu palindrom bo‘ladi. Masalan, aziza. Dasturingiz kiritilgan so‘z palindrom bo‘lsa “Yes”, aks holda “NO” chiqarsin.

Kiruvchi	Chiquvchi

kiyik	Yes
aziza	Yes
aziz	No

13. Berilgan so‘zga nechta harf qo‘shsa palindrom bo‘ladi. Shu sonni chiqaring. Masalan,

Kiruvchi	**Chiquvchi**
aziz	1
olim	3
aslo	3

14. Berilgan so‘z palindrom bo‘lishi uchun qanday bo‘lishi kerak. Regesrga e’tibor bering (1354).

Kiruvchi	**Chiquvchi**
Aziz	AzizA
Olim	OlimilO
OnLine	OnLineniLnO
KiyiK	KiyiK

15. Matnli fayldagi so‘zlarni ulardagi simvollar soni bo‘yicha kamayish va o‘sish tartibida chiqaring. Bunda satr uzunligi 200 va har bir so‘zning uzunligi 10 bo‘lishi mumkin. Masalan,

Kiruvchi	**Chiquvchi**
2014 yil Sog‘lom bola yili	Sog‘lom 2014 bola yili yil

16. Matnli fayldagi so‘zlar berilgan. Shu so‘zlardagi simvollar soni beshdan oshganlarini oxirgi uchta simvolini “$” bilan almashtiring. Masalan,

Kiruvchi	**Chiquvchi**
big	big
mouses	mou$
industry	indust$
dastur	das$

klaviatura	klaviat$

17. Matnli fayldagi so‘zlardan faqatgina boshi va oxiridagi simvollari bir xil bo‘lmagan sozlarni qoldiring (Katta va kichik harflar farq qilmaydi). Agarda bunday so‘z bo‘lmasa “NO” chiqaring.

Kiruvchi	**Chiquvchi**
Mana senga olam olam gul!	NO
Toshkent O‘zbekiston poytaxti.	O‘zbekiston poytaxti.
Mening Malibu rusumli mashinam bor.	Mening Malibu rusumli bor.

18. Berilgan so‘zdagi katta va kichik harflar nisbatini toping. Avval katta harflarni keyin kichik harflarni foizini nuqtadan keyin ikkita aniqlikda chiqaring.

Kiruvchi	**Chiquvchi**
Ola!	25.00 50.00
Salom	20.00 80.00
ToshkenT	25.00 75.00
Toshkent	12.50 87.50

19. Berilgan matndagi so‘zlarning o‘rtacha uzunligini toping.

Kiruvchi	**Chiquvchi**
Mana senga olam olam gul	4.00
Toshkent O‘zbekiston poytaxti	9.00
Tashkent capital of Uzbekistan	6.75

20. “One” va “puton” bir-biri bilan gaplashyapdi. “One” "out" va "output" so‘zlarini va “puton” so‘zini gapira oladi. “Puton” esa "in", "input" va "one" so‘zlarini gapira oladi. Va ular bir birini probelsiz yozilgan ushbu so‘zlar bilan bemalol tushunishadi. Sizga N ta satrda ularning dialogi berilgan. Shu dialoglardan qaysi biri dialog bo‘lishini aniqlang. Agar berilgan dialog faqat

“one” va “puton” ning so‘zlaridan iborat bo‘lsa “YES” aks holda “NO” chiqaring (1102). Masalan,

Kiruvchi	Chiquvchi
6 puton inonputin oneputonininputoutoutput oneininputwooutoutput outpu utput	YES NO YES NO NO NO

21. Siz maqolalarni qabul qilyapsiz. Maqolalardagi senzuraga oid so‘zlarni tekshirishingiz kerak. Berilgan senzuraga oid so‘zlarni maqoladan qidiring. Birinchi satrda $n\,(1 \leq n \leq 10000)$ soni berilgan. Keyingi n ta satrda esa senzuraga oid so‘zlar berilgan. Undan keyin esa maqoladagi satrlar soni m berilgan. Birinchi marta topilgan senzuraga oid so‘zni satrini va o‘rnini chiqaring. Agar bunday so‘zlar topilmasa, “NO” chiqaring (1269). Masalan,

Kiruvchi	Chiquvchi
5 dear sweetie angel dream baby 8 Had I the heavens' embroidered cloths, Enwrought with golden and silver light, The blue and the dim and the dark cloths Of night and light and the half-light, I would spread the cloths under your feet: But I, being poor, have only my dreams; I have spread my dreams under your feet; Tread softly because you tread on my dreams.	6 33

22. Berilgan matndagi takrorlanuvchi so‘zlarni olib tashlang.

Kiruvchi	Chiquvchi

Mana senga olam olam gul.	Mana senga olam gul.
I am going at home. They are going at home too.	I am going at home. They are home too.

23.Berilgan matndagi juft simvolli so‘zlarning birinchi yarmini chiqaring.

Kiruvchi	**Chiquvchi**
Mana senga olam olam gul.	Ma senga ol ol gul.
I am going at home. They are going at home too.	I a going a ho. Th are going t ho too.

24.Berilgan matndagi juft simvolli so‘zlarning ikkinchi yarmini chiqaring.

Kiruvchi	**Chiquvchi**
Mana bugun bayram.	na bugun ram.
I am going at home. They are going at home too.	I m going t me. ey are t me too.

25.Berilgan matndagi juft simvolli so‘zlarning birinchi va ikkinchi yarimlarini o‘rnini almashtiring

Kiruvchi	**Chiquvchi**
Mana bugun bayram.	naMa bugun rambay.
I am going at home. They are going at home too.	I ma going ta meho. eyTh are ta meho too.

26.Berilgan matndagi juft simvolli so‘zlarning birinchi va ikkinchi yarimlarini o‘rnini almashtiring

Kiruvchi	**Chiquvchi**
Mana bugun bayram.	naMa bugun rambay.
I am going at home. They are going at home too.	I ma going ta meho. eyTh are ta meho too.

27.Berilgan matndagi toq simvolli so‘zlarni teskari tartibda chiqaring.

Kiruvchi	**Chiquvchi**

Mana bugun bayram.	Mana nugub bayram
I am going at home. They are going at home too.	I am gniog at home. They era gniog at home oot.

28. Berilgan matnda necha xil unli tovush qatnashayotganligini aniqlovchi dastur tuzing

Kiruvchi	**Chiquvchi**
Mana bugun bayram.	2
Bugun rosa qiyin kun bo‘ldi.	4

29. Berilgan matnda necha xil undosh tovush qatnashayotganligini aniqlovchi dastur tuzing. Katta va kichik harflar farq qilmaydi.

Kiruvchi	**Chiquvchi**
Mana bugun bayram.	6
Bugun rosa qiyin kun bo‘ldi.	10

30. Berilgan matnda barcha probellarni “_” bilan almashtiring.

Kiruvchi	**Chiquvchi**
Mana bugun bayram.	Mana_bugun_bayram.
Bugun rosa qiyin kun bo‘ldi.	Bugun_rosa_qiyin_kun__bo‘ldi.

Nazorat savollari

1. Struktura deb nimaga aytiladi ?
2. Struktura nima uchun kerak?
3. Ochiq (public) va yopiq (private) o‘zgaruvchi-a’zolar orasida qanday farq bor ?
4. Sinfning funksiya a’zolari qachon yopiq bo‘lishi lozim?
5. Sinfning funksiya a’zolari qachon ochiq bo‘lishi lozim?
6. Agar sinf class so‘zi yordamida ta’riflangan bo‘lsa ko‘zda tutilgan komponentalari bo‘yicha qanday murojaat huquqiga ega bo‘ladi?

7. Qaysi holda sinf usullari joylashtiriluvchi funksiya hisoblanadi?
8. Agarda sinfning ikkita obyektini e’lon qilsak, ularning o‘zgaruvchi a’zolari qiymati turlicha bo‘lishi mumkinmi?
9. Konstruktorlar xossalarini ko‘rsating.
10. Sinf obyektini hosil qilishda qanday funksiya chaqiriladi?
11. Sinf obyekti uchun ajratilgan xotira maydonini tozalashda qanday funksiya chaqiriladi.
12. Statik maydonlar xususiy bo‘lishi mumkinmi?

Fayllar bilan ishlashning bitli rejimi.

Fayl bilan bitli almashish rejimi `gets()` va `puts()` funksiyalari yordamida tashkil etiladi. Bu funksiyalarga esa quyidacha murojaat etiladi:

```
c = gets(fp);
puts(c,fp);
```

Bu yerda fp-ko‘rsatkich

c-int turidagi o‘zgaruvchi

Namuna. Klaviaturadan simvol kiritib faylga yozing. Matn oxirini ‘#’ belgisi bilan ko‘rsating. Fayl nomi foydalanuvchidan so‘raladi. Agar <enter> klavishasi bosilsa faylga CR va LF (qiymatlari 13 va 10) konstantalar yoziladi. Keyinchalik fayldan simvollarni o‘qishda bu konstantalar satrlarni ajratishga imkon beradi.

```
#include <stdio.h>
int main()
{
FILE *fp;
char c;
const char CR = '\015';
const char LF = '\012';
char fname[20];
puts("fayl nomini kiriting:\n");
gets(fname);
if((fp = fopen(fname, "w")) == NULL)
{
perror(fname);
return 1;
}
while ((c = getchar())!= '#')
{
if (c == '\n')
{ putc(CR,fp);
putc(LF,fp);
}
else putc (c,fp);
}
fclose(fp);
}
```

5.1-variant. Matnli fayllarga oid masalalar

5.1-jadval

№	**Masala sharti**
1.	**1.** N va K butun musbat sonlar va fayl nomi berilgan. Yangi matnli fayl hosil qilinsin va unga N ta satr va har bir satr K ta "*" (yulduzcha) belgisidan iborat bo‘lsin. 2. 10 ta raqamdan iborat S satr va shifrlangan matnli fayl berilgan. Text59 masaladagi algoritm bo‘yicha shifrlangan matnni deshifrlovchi programma tuzilsin.
2.	**1.** N (0<N<27) butun son va fayl nomi berilgan. Berilgan nomdagi matnli fayl hosil qilinsin va unga: birinchi satri "a" kichik lotin harfi, ikkinchisiga"ab", uchinchisiga "abc" va h.k satrlarni saqlovchi N ta satr yozilsin. **2.** 10 ta raqamdan iborat S satr va lotin harflaridan iborat matnli fayl berilgan. Matnli fayli quyidagicha shifrlang: Matnli fayl satrining K - belgisini, shu belgining kodiga S satridagi K -raqamini qo‘shishdan hosil bo‘lgan kod belgisiga almashtiring. Agar K=11 bo‘lsa, ya’na satrning birinchi raqamidan boshlang.
3.	1. N (0<N<27) butun son va fayl nomi berilgan. Berilgan nomdagi fayl hosil qilinsin va unga uzunligi N ga teng bo‘lgan N ta satr quyidagicha yozilsin; K-nomerdagi satr (K=1N) katta lotin harflarining boshlang‘ich K ta harfini va undan o‘ngda "*" belgisidan iborat bo‘lsin. Masalan N=**4** uchun fayl quyidagi satrlardan iborat bo‘lishi kerak. "A***", "AB**", "ABC*", "ABCD". 2. Matnli fayl berilgan. Undagi har bir uchragan kichik lotin harflarining uchrashlar miqdori sanalsin va tarkibi quyidagi ko‘rinishda bo‘lgan matnli fayl hosil qilinsin "<harf>-<uchrashlar soni>"(masalan, "a-25"). Matnda uchramagan harflar hisobga olinmasin. Satrlarni harflarning uchrash sonini kamayish bo‘yicha, teng sondagi uchrashlarni esa ularning kodlari bo‘yicha o‘sish tartibida joylashtirilsin.
4.	1. Matnli fayl berilgan. Uning tarkibiga kiruvchi satrlar va belgilar soni (miqdori) chop qilinsin. (satrning oxiri EOLN va EOF fayl oxirlari markerlari belgilarni sanayotganda hisobga olinmasin). 2. Matnli fayl berilgan. Undagi har bir uchragan kichik lotin harflarining uchrashlar miqdori sanalsin va tarkibi quyidagi ko‘rinishda bo‘lgan matnli fayl hosil qilinsin "<harf>-<uchrashlar soni>"(masalan, "a-25"). Matnda uchramagan harflar hisobga olinmasin. Satrlar kodi bo‘yicha o‘sish tartibida joylashsin.
5.	1. Satr va matnli fayl berilgan. S satr fayl oxiriga qo‘shilsin. 2. Matnli fayl berilgan. Matnli fayldagi barcha uchragan belgilarni, probel va tinish belgilariga ega bo‘lgan (takrorlanishsiz) belgili fayl hosil qilinsin. Belgilar kodi bo‘yicha kamayish tartibida joylashsin.
6.	1. Ikkita matnli fayl berilgan. Birinchi fayl oxiriga ikkinchi fayl qo‘shilsin. 2. Matnli fayl berilgan. Matnli fayldagi barcha uchragan belgilarni, probel va tinish belgilariga ega bo‘lgan (takrorlanishsiz) belgili fayl hosil qilinsin. Belgilar kodi bo‘yicha o‘sish tartibida joylashsin.
7.	1. S satr va matnli fayl berilgan. S satr fayl boshiga qo‘shilsin.

	2. Matnli fayl berilgan. Matnli fayldagi barcha uchragan belgilarni, probel va tinish belgilariga ega bo‘lgan (takrorlanishsiz) belgili fayl hosil qilinsin. Belgilar matnda birinchi joylashgan tartibida joylashtirilsin.
8.	1. Ikkita matnli fayl berilgan. Birinchi fayl boshiga ikkinchi fayl qo‘shilsin. 2. Matnli fayl berilgan. Matnli fayldagi barcha uchragan tinish belgilariga ega bo‘lgan belgili fayl hosil qilinsin.
9.	1. K butun soni va matnli fayl berilgan. K- nomerdagi satrdan oldin bo‘sh satr qo‘yilsin. Agar bunday nomerli satr mavjud bo‘lmasa, u holda fayl o‘zgartirishsiz qoldirilsin. 2. Butun sonlardan iborat uchta ustunli jadvalga ega bo‘lgan matnli fayl berilgan. Jadvaldagi har bir ustunining boshiga va oxitiga hamda ular orasiga ajratuvchi belgi joylashtirilgan. Jadvaldagi ustunlar kengligi va ularning tekislanishi hamda ajratuvchi belgilar ko‘rinishi ixtiyoriy. Boshlang‘ich jadvalning har bir satridagi sonlar yig‘indisiga ega bo‘lgan yangi butun sonlar fayli hosil qilinsin.
10.	1. K butun soni va matnli fayl berilgan. K- nomerdagi satrdan keyin bo‘sh satr qo‘yilsin. Agar bunday nomerli satr mavjud bo‘lmasa, u holda fayl o‘zgartirishsiz qoldirilsin. 2. Haqiqiy sonlardan iborat uchta ustunli jadvalga ega bo‘lgan matnli fayl berilgan. Ustun kengligi, tekislash usullari ixtiyoriy ravishda berilgan, maxsus ajratuvchi belgilarga ega emas. Uchta haqiqiy sonlar fayli hosil qilinsin va har bir fayl jadvalning mos ustunidagi sonlarni o‘zida saqlasin.
11.	1. Matnli fayl berilgan. Undagi barcha bo‘sh satrlar ikkilantirilsin. 2. Matnli fayl berilgan. Har bir satrning birinchi 30 ta belgisi matndan, qolgani esa haqiqiy sonlardan iborat. Boshlang‘ich faylning barcha matn qismiga ega bo‘lgan mantli fayl va boshlang‘ich faylning barcha haqiqiy sonlar qismiga ega bo‘lgan haqiqiy sonlar fayli hosil qilinsin.
12.	1. S satr va matnli fayl berilgan. Fayldagi barcha bo‘sh satrlar S satrga o‘zgartirilsin. 2. Butun sonlar fayli va matnli fayl berilgan. Matn faylining har bir satrining oxiriga butun sonlar faylidagi mos sonlari joylashtirilsin. Agar butun sonlar fayli matn faylidan qisqa bo‘lsa, u holda matn faylidagi qolgan satrlar o‘zgartirmasdan qoldirilsin.
13.	1. Bo‘sh bo‘lmagan matnli fayl berilgan. Undagi birinchi satr o‘chirilsin. 2. Har bir satrida probellar bilan ajratilgan bir nechta sonlarni tasvirlovchi matnli fayl berilgan (haqiqiy sonlar nol bo‘lmagan kasr qismiga ega). Tarkibida boshlang‘ich faylning barcha butun sonlariga ega bo‘lgan butun sonlar fayli hosil qilinsin.
14.	1. Bo‘sh bo‘lmagan matnli fayl berilgan. Undagi oxirgi satr o‘chirilsin. 2. Har bir satri o‘ng va chap tomonidan bir nechta probellar bialn to‘ldirilgan butun yoki haqiqiy sonlarni tasvirlovchi matnli fayl berilgan. (haqiqiy sonlar nol bo‘lmagan kasr qismiga ega). Butun sonlar miqdori va ularning yig‘indisi chop qilinsin.

15.	1. K butun soni va matnli fayl berilgan. Undaki K-nomerdagi satr o‘chirilsin. Agar faylda bunday nomerdagi satr mavjud bo‘lmasa, u holda fayl o‘zgartirishsiz qoldirilsin. 2. Har bir satrida probellar bilan ajratilgan bir nechta sonlarni tasvirlovchi matnli fayl berilgan (haqiqiy sonlarning kasr qismi noldan farqli). Boshlang‘ich faylning barcha noldan farqli kasr qismiga ega bo‘lgan sonlaridan iborat (taribini o‘zgartirmagan holda) haqiqiy sonlar fayli hosil qilinsin.
16.	1. Matnli fayl berilgan. Undagi barcha bo‘sh satrlar o‘chirilsin. 2. Har bir satrida bittadan butun yoki haqiqiy son bo‘lgan matnli fayl berilgan. Sonlar o‘ng va chap tomonidan bir nechta probellar bilan to‘ldirilgan. (haqiqiy sonlarning kasr qismi noldan farqli). Kasr qismi nol bo‘lmagan sonlar miqdori va ularning yig‘indisi chop qilinsin.
17.	1. Ikkita matnli fayl berilgan. Birinchi faylning har bir satridan so‘ng ikkinchi fayldagi mos satrlar qo‘shilsin. Agar ikkinchi fayl birinchi fayldan kalta bo‘lsa, u holda qolgan satrlar o‘zgartirishsiz qoldirilsin. 2. Har bir satrida bittadan butun son bo‘lgan matnli fayl berilgan. Butun sonlar o‘ng va chap tomonidan bir nechta probellar bilan to‘ldirilgan. Shu sonlar miqdori va ularning yig‘indisi chop qilinsin.
18.	1. K butun soni va matnli fayl berilgan. Faylning har bir satridan birinchi к ta belgi o‘chirilsin. (agar satr uzunligi K dan kichik bo‘lsa, u holda satrning hamma belgilari o‘chirilsin). 2. N butun soni va A, B haqiqiy sonlari berilgan. Sin(x) va cos(x) funksiyasining [A, B] oraliqdagi (B-A)/N qadam bilan hosil bo‘luvchi qiymatlari jadvalini saqlovchi matnli fayl hosil qilinsin. Jadval uchta ustundan iborat: x argumentli (8 ta pozitsiya va uning 4 tasi kasr qismi) va sin(x) hamda cos(x) ning qiymatlari (12 ta pozitsiyadan va ulardan 8 tasi kasr qismi). Ustunlar o‘ng tarafdan tekislanadi.
19.	1. Matnli fayl berilgan. Fayldagi hamma katta lotin harflari kichik harflarga va aksincha, barcha kichik lotin harflari katta harflarga almashtirilsin. 2. N butun soni va A, B haqiqiy sonlari berilgan. funksiyasining [A,B] oraliqdagi (B-A)/N qadam bilan hosil bo‘luvchi qiymatlari jadvalini saqlovchi matnli fayl hosil qilinsin. Jadval ikkita ustundan iborat: x argumentli (10 ta pozitsiya va uning 4 tasi kasr qismi) va uning qiymatlari (15 ta pozitsiya va ulardan 8 tasi kasr qismi). Ustunlar o‘ng tarafdan tekislanadi.
20.	1. Matnli fayl berilgan. Undagi barcha ketma-ket kelgan probellar bitta probelga almashtirilsin. 2. Bir xil o‘lchamdagi butun sonlardan iborat ikkita fayl berilgan. Shu sonlardan, kengligi 30 ta belgidan iborat bo‘lgan ikkita ustunga ajratilgan matnli fayl hosil qilinsin. Birinchi ustunda birinchi boshlang‘ich fayl sonlari, ikkinchisida esa ikkinchi boshlang‘ich fayl sonlari joylashadi. Matnli fayldagi har bir satrning boshi va oxiriga "|" ajratuvchi (kod 124) qo‘shilsin. Sonlar ustunning o‘ng tarafiga tekislanadi.

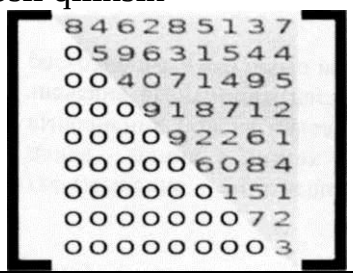
21.	1. Uchtadan ko‘p bo‘lgan satrga ega bo‘lgan matnli fayl berilgan. Shu fayldan oxirgi uchta satr o‘chirilsin. 2. Bir xil o‘lchamdagi butun sonlardan iborat ikkita fayl berilgan. Shu sonlardan, kengligi 30 ta belgidan iborat bo‘lgan ikkita ustunga ajratilgan matnli fayl hosil qilinsin. Birinchi ustunda birinchi boshlang‘ich fayl sonlari, ikkinchisida esa ikkinchi boshlang‘ich fayl sonlari joylashadi. Matnli fayldagi har bir satrning boshi va oxiriga "|" ajratuvchi (kod 124) qo‘shilsin. Sonlar ustunning o‘ng tarafiga tekislanadi.
22.	1. K (0<K<10) butun son va K ta dan ko‘p bo‘lgan satrga ega matnli fayl berilgan. Shu faylning oxirgi K ta satri o‘chirilsin. 2. K (>25) butun soni va chap tarafdan tekislangan matnli fayl berilgan. Abzas qizil satr (Text26 masalaga qarang) orqali ajratiladi, bo‘sh satrlar esa mavjud emas. Matnni shunday formatlangki, uning kengligi K ta belgidan oshmasin va abzaslarga bo‘linganligini saqlagan holda chap tarafga tekislansin. Satrning oxiridagi probellar o‘chirilsin. Formatlangan matn yangi faylda saqlansin.
23.	1. K (0<K<10) butun son va K ta dan ko‘p bo‘lgan satrga ega matnli fayl berilgan. Boshlang‘ich faylning oxirgi K ta elementidan iborat bo‘lgan yangi matnli fayl hosil qilinsin. 2. K (>25) butun soni va chap tarafdan tekislangan matnli fayl berilgan. Matn abzasi bitta bo‘sh satr orqali ajratiladi. Matnni shunday formatlangki, uning kengligi K ta belgidan oshmasin va abzaslarga bo‘linganligini saqlagan holda chap tarafga tekislansin.
24.	1. Matnli fayl berilgan. Agar abzas bitta yoki bir nechta bo‘sh satrlar bilan ajratilgan bo‘lsa, u holda matndagi abzaslar soni aniqlansin. 2. Chap tomonidan tekislangan matnga ega bo‘lgan matnli fayl berilgan. Abzaslar bitta bo‘sh satr orqali ajratiladi. Har bir bo‘sh bo‘lmagan satrdagi eng oxirgi probeldan boshlab satr so‘zlari orasidagi probellarni qo‘shish orqali matn kenglik bo‘yicha tekislansin (ham chap, ham o‘ng tomondan tekislansin). Matn kengligi 50 ta belgiga teng deb olinsin.
25.	1. K butun son va matnli fayl berilgan. Fayldan K-nomerdagi abzas o‘chirilsin (abzas bir-biridan bitta yoki bir nechta bo‘sh satrlar bilan ajratiladi). O‘chirilgan abzasdan oldin va keyin keluvchi bo‘sh satrlar o‘chirilmasin. Agar berilgan nomerdagi abzas mavjud bo‘lmasa u holda fayl o‘zgartirishsiz qoldirilsin. 2. O‘ng tomonidan tekislangan matnga ega bo‘lgan matnli fayl berilgan. Har bir bo‘sh bo‘lmagan satrning boshlang‘ich probellarini yarmini o‘chirish orqali matn markazga tekislantirilsin. Toq uzunlikka ega bo‘lgan satrlarga, markazlashtirishdan oldin chap tomondan bitta probel o‘chirilsin.
26.	1. Matnli fayl berilgan. Agar fayl matnidagi har bir abzasning birinchi satri 5 ta probeldan boshlansa ("qizil satr"), u holda shu matndagi abzaslar soni aniqlansin. Abzaslar orasidagi bo‘sh satrlar hisobga olinmasin. 2. Chap tomonidan tekislangan matnga ega bo‘lgan matnli fayl berilgan. Har bir bo‘sh bo‘lmagan satr boshiga kerakli miqdorda probel qo‘shish orqali matnni markazga tekislanilsin. (Matn kengligi 50 ta belgiga teng deb olinsin). Toq

	uzunlikka ega bo‘lgan satrlarga, markazlashtirishdan oldin chap tomondan probel qo‘shilsin.
27.	1. Matnli fayl berilgan. Fayl matnidagi har bir abzasning birinchi satri 5 ta probeldan boshlangan. K butun soni va mantli fayl berilgan. Fayldan K-nomerdagi abzas o‘chirilsin. Abzas qizil satr orqali ajratiladi. 2. Chap tomonidan tekislangan matnga ega bo‘lgan matnli fayl berilgan. Har bir bo‘sh bo‘lmagan satr boshiga kerakli miqdorda probel qo‘shish orqali matnni o‘ng tomondan tekislanilsin. (Matn kengligi 50 ta belgiga teng deb olinsin).
28.	1. Matnli fayl berilgan. Fayl matnidagi har bir abzasning birinchi satri 5 ta probeldan boshlangan. Abzas *qizil satr* yordamida ajratiladi. Faylda bo‘sh satrlar mavjud emas. Har bir qo‘shni abzaslar orasiga bittadan bo‘sh satr joylashtirilsin. (faylning boshiga va oxiriga bo‘sh satr qo‘shilmasin). 2. Matnli fayl va lotin harflarining kichik harflaridan C belgi berilgan. Matnli fayl hosil qilinsin va unga boshlang‘ich fayldagi barcha shu C harfidan boshlanuvchi so‘zlar yozilsin (katta yoki kichik harflar bilan boshlanuvchi). So‘z deb, probellarga, tinish belgilariga ega bo‘lmagan va probellar, tinish belgilari yoki satrning boshi/oxiri bilan chegaralangan belgilar to‘plamiga aytiladi. Agar boshlang‘ich fayl tarkibida mos so‘zlar mavjud bo‘lmasa, u holda natijaviy fayl bo‘sh holda qoldirilsin.
29.	1. Matnli fayl berilgan. Matnning eng uzun bo‘lgan birinchi so‘zi chop qilinsin. So‘z deb, probellar bilan chegaralangan yoki satrning boshi/oxiri bo‘lgan belgilar to‘plamiga aytiladi. 2. Matnli fayl va lotin harflarining bosh harflaridan C belgi berilgan. Matnli fayl hosil qilinsin va unga boshlang‘ich fayldagi barcha shu C harfidan boshlanuvchi so‘zlar yozilsin (katta yoki kichik harflar bilan boshlanuvchi). So‘z deb, probellarga, tinish belgilariga ega bo‘lmagan va probellar, tinish belgilari yoki satrning boshi/oxiri bilan chegaralangan belgilar to‘plamiga aytiladi. Agar boshlang‘ich fayl tarkibida mos so‘zlar mavjud bo‘lmasa, u holda natijaviy fayl bo‘sh holda qoldirilsin.
30.	1. Matnli fayl berilgan. Matnning eng qisqa bo‘lgan oxirgi so‘zi chop qilinsin. So‘z deb, probellar bilan chegaralangan yoki satrning boshi/oxiri bo‘lgan belgilar to‘plamiga aytiladi. 2. K butun soni va matnli fayl berilgan. Yangi satrli fayl hosil qilinsin va unga boshlang‘ich fayldagi uzunligi K ga teng bo‘lgan barcha so‘zlar yozilsin. So‘z deb, probellarga, finish belgilariga ega bo‘lmagan va probellar, tinish belgilari yoki satrning boshi/oxiri bilan chegaralangan belgilar to‘plamiga aytiladi. Agar boshlang‘ich fayl K uzunlikdagi so‘z bo‘lmasa, u holda natijaviy fayl bo‘sh holda qoldirilsin.

5.2-variant. Binar fayllarga oid masalalar

5.2 -jadval

№	**Masala sharti**

1.	1. S satr berilgan. Agar S faylning mumkin bolgan nomi bo‘lsa, u holda shu nomdagi bo‘sh fayl hosil qilinsin va TRUE chop qilinsin. Agar S nomdagi faylni yaratish mumkin bo‘lmasa u holda FALSE chop qilinsin. 2. Butun sonlar fayli berilgan. Undagi barcha juft nomerdagi elementlari o‘chirilsin. 3. Tarkibi yuqori uchburchakli matritsadan iborat bo‘lgan haqiqiy sonlar fayli berilgan. Tarkibi ushbu berilgan matritsaning noldan farqli elementlaridan iborat bo‘lgan yangi fayl hosil qilinsin 
2.	1. Fayl nomi N butun soni berilgan (N>1). Berilgan nomdagi fayl hosil qilinsin va unga N ta birinchi musbat juft sonlari chop qilinsin (2,4,...). 2. 50 ta elementdan ortiq bo‘lgan elementlardan iborat butun sonlar fayli berilgan. Shu fayl elementlarini boshidan boshlab 50ta elementgacha o‘chirilsin. 3. Tarkibi quyi uchburchakli A va B matritsaning noldan farqli elementlaridan iborat bo‘lgan SA va SB nomli haqiqiy sonlar fayli berilgan. Tarkibi A va B matritsaning noldan farqli elementlarini ko‘paytmasidan hosil bo‘lgan matritsa elementlaridan iborat bo‘lgan yangi SC fayl hosil qilinsin. Agar A va B matritsalarni ko‘paytirish imkoni bo‘lmasa, SC fayl bo‘sh holda qoldirilsin.
3.	1. Fayl nomi va A va D haqiqiy sonlar berilgan. Shu nomdagi fayl hosil qilinsin va unga A boshlang‘ich hadi va D farqiga ega bo‘lgan arifmetik progressiyaning birinchi 10ta hadi yozilsin. A, A+D, A+2*D, A+3*D,… 2. Hech bo‘lmaganda bitta probel belgisi mavjud bo‘lgan belgili fayl berilgan. Shu fayldagi birinchi kelgan probeldan oldin joylashgan barcha elementlar o‘chirilsin. (Shu probelni ham hisobga olgan holda) 3. Tarkibi yuqori uchburchakli A va B matritsaning noldan farqli elementlaridan iborat bo‘lgan SA va SB nomli haqiqiy sonlar fayli berilgan. Tarkibi A va B matritsaning noldan farqli elementlarini ko‘paytmasidan hosil bo‘lgan matritsa elementlaridan iborat bo‘lgan yangi SC fayl hosil qilinsin. Agar A va B matritsalarni ko‘paytirish imkoni bo‘lmasa, SC fayl bo‘sh holda qoldirilsin.
4.	1. 4ta faylning nomi berilgan. Shu fayllarning nechtasi joriy katalogda joylashgani aniqlansin. 2. Hech bo‘lmaganda bitta probel belgisi mavjud bo‘lgan belgili fayl berilgan. Shu fayldagi oxirgi kelgan probeldan keyin joylashgan barcha elementlar o‘chirilsin. (Shu probelni ham hisobga olgan holda)

	3. Tarkibi uch dioganalli matritsaning faqat noldan farqli elementlaridan iborat bo‘lgan haqiqiy sonlar fayli berilgan. Yangi fayl hosil qilinsin va u berilgan matritsaning barcha elementlarini saqlasin.
5.	1. Butun sonlar fayli berilgan. Shu fayl tarkibiga kiruvchi elementlar soni aniqlansin. Agar bunday fayl mavjud bo‘lmasa u holda -1 chop qilinsin. 2. Hech bo‘lmaganda bitta probel belgisi mavjud bo‘lgan belgili fayl berilgan. Shu fayldagi birinchi kelgan probeldan keyin joylashgan barcha elementlar (probelni ham hisobga olgan holda) o‘chirilsin 3. Tarkibi quyi uchburchakli matritsaning faqat noldan farqli elementlaridan iborat bo‘lgan haqiqiy sonlar fayli berilgan. Yangi fayl hosil qilinsin va u berilgan matritsaning barcha elementlarini saqlasin.
6.	1.Manfiy bo‘lmagan butun sonlardan iborat fayl va K soni berilgan (K butun) Faylning K-elementi chop qilinsin (elementlar 1dan boshlab nomerlanadi). Agar bunday element mavjud bo‘lmasa, (-1) chop qilinsin. 2. S1, S2 satrlar va bir nechta fayllar berilganlarini saqlovchi butun sonlar fayl-arxivi berilgan. Yangi S1 va S2 butun sonlar fayli hosil qilinsin va ularning birinchisiga fayl-arxivdagi barcha fayllarning boshlang‘ich elementlari, ikkinchisida esa barcha fayllarning oxirgi elemetlari yozilsin (tartibini o‘zgartirmagan holda). 3. Tarkibi yuqori uchburchakli matritsaning faqat noldan farqli elementlaridan iborat bo‘lgan haqiqiy sonlar fayli berilgan. Yangi fayl hosil qilinsin va u berilgan matritsaning barcha elementlarini saqlasin.
7.	1. Elementlar soni 4dan ko‘p bo‘lgan butun sonlardan iborat fayl berilgan. Ushbu faylning birinchi, ikkinchi, oxirgi va oxiridan bitta oldingi elementlar chop qilinsin. 2. S satr, butun N (>0) soni va S1,…,SN nomli N ta butun sonlar fayl-arxivi berilgan. Fayl-arxivdan N nomerli fayl tiklansin va S nomi bilan saqlasin. Agar fayl-arxiv N fayldan kam bo‘lgan berilganlarni saqlasa, u holda natijaviy fayl bo‘sh holda qoldirilsin. 3. Tarkibi uch dioganalli matritsaning noldan farqli elemetlaridan iborat bo‘lgan haqiqiy sonlar fayli hamda I va J butun sonlar berilgan. l-satr va J-ustunda joylashgan matritsa elemeti va matritsa tartibi chop qilinsin. (satr va ustunlar 1 dan boshlab nomerlanadi). Agar talab qilingan element matritsaning nol qismida bo‘lsa, u holda 0 qiymati chop qilinsin; agar element mavjud bo‘lmasa, u holda -1 qiymati chop qilinsin.
8.	1. Ikkita haqiqiy sonlar fayli berilgan. Shu fayllarning birinchisi bo‘sh bo‘lmagani ma’lum va ikkinchisi joriy katalogda mavjud emas. Mavjud bo‘lmagan bo‘sh fayl yaratilsin va mavjud bo‘lgan faylning birinchi va oxirgi elementi shu faylga yozilsin. (tartibi buzilmagan holda). 2. S satr, butun N (>0) soni va binar formatdagi bir nechta fayllar (6 tadan ko‘p bo‘lmagan) berilganlarini saqlovchi butun sonlar fayl-arxivi berilgan. Arxivda saqlanuvchi har bir fayl uchun uning barcha elementlarini o‘rta arifmetigi

	(haqiqiy son) topilsin va topilgan sonlarni (tartibini buzmagan holda) S nomli haqiqiy sonlar fayliga yozilsin. 3. Tarkibi quyi uchburchakli matritsaning noldan farqli elemetlaridan iborat bo‘lgan haqiqiy sonlar fayli hamda I va J butun sonlar berilgan. l-satr va J-ustunda joylashgan matritsa elemeti va matritsa tartibi chop qilinsin. (satr va ustunlar 1 dan boshlab nomerlanadi). Agar talab qilingan element matritsaning nol qismida bo‘lsa, u holda 0 qiymati chop qilinsin; agar element mavjud bo‘lmasa, u holda -1 qiymati chop qilinsin.
9.	1. Ikkita haqiqiy sonlar fayli berilgan. Shu fayllardan biri (birinchisi bo‘lishi shart emas) bo‘sh bo‘lmagani ma’lum va boshqasi esa joriy katalogda mavjud emas. Mavjud bo‘lmagan fayl yaratilsin va mavjud bo‘lgan faylning birinchi va oxirgi elementi shu faylga yozilsin. (tartibi buzilmagan holda). 2. S satr, butun N(>0) soni va S1,...,SN nomli N ta butun sonlar fayl-arxivi berilgan. Fayl-arxivdan N nomerli fayl tiklansin va S nomi bilan saqlansin. Agar fayl-arxiv N fayldan kam bo‘lsa , u holda natijaviy fayl bo‘sh holda qoldirilsin. 3. Tarkibi uch dioganalli matritsadan iborat bo‘lgan haqiqiy sonlar fayli berilgan. Tarkibi ushbu berilgan matritsaning noldan farqli elementlaridan iborat bo‘lgan yangi fayl hosil qilinsin.
10.	1. Haqiqiy sonlar fayli berilgan. Berilgan fayl elementlarini teskari tartibda saqlovchi yangi fayl hosil qilinsin. 2. SO satri, butun N(<=4) soni va S1.....SN nomli N ta butun sonlar fayli berilgan. Ularning tarkibini quyidagi formatdan foydalangan holda SO nomli yangi fayl-arxivda birlashtirilsin: fayl -arxivning birinchi elementi sifatida N soni, keyingi N ta elementi esa har bir boshlang‘ich fayllarning o‘lchami (elementlar soni) va ulardan so‘ng ketma - ket har bir boshlang‘ich fayllarning berilganlari joylashtiriladi. 3. Tarkibi yuqori uchburchakli matritsaning noldan farqli elemetlaridan iborat bo‘lgan haqiqiy sonlar fayli hamda I va J butun sonlar berilgan. l-satr va J-ustunda joylashgan matritsa elemeti va matritsa tartibi chop qilinsin. (satr va ustunlar 1 dan boshlab nomerlanadi). Agar talab qilingan element matritsaning nol qismida bo‘lsa, u holda 0 qiymati chop qilinsin; agar element mavjud bo‘lmasa, u holda -1 qiymati chop qilinsin.
11.	1. Haqiqiy sonlar fayli berilgan. Ikkita yangi fayl hosil qilinsin. Ularning biriga boshlang‘ich faylning toq nomerdagi elementlarini (1, 3,...), ikkinchisi esa - juft nomerdagi elementlarini (2, 4...) saqlovchi programma tuzilsin. 2. Elementlari kamayish tartibida joylashgan S1, S2, va S3 haqiqiy sonlar fayli berilgan. Shu fayllarni yangi S4 faylga shunday birlashtirilsinki, natijada elementlar yana kamayish tartibida joylashsin. 3. Tarkibi quyi uchburchakli matritsadan iborat bo‘lgan haqiqiy sonlar fayli berilgan. Tarkibi ushbu berilgan matritsaning noldan farqli elementlaridan iborat bo‘lgan yangi fayl hosil qilinsin.

12.	1. Butun sonlar fayli berilgan. Ikkita yangi fayl hosil qilinsin. Ulardan birinchisi boshlang‘ich faylning juft sonlarini, ikkinchisi esa toq sonlarini o‘zida saqlasin. Agar boshlang‘ich faylda juft yoki toq sonlar mavjud bo‘lmasa u holda natijaviy fayl bo‘sh holda qoldirilsin. 2. Haqiqiy sonlar o‘sish tartibida S1 va S2 fayllariga joylashirilgan. Shu fayllarni yangi S3 fayliga shunday birlashtirilsinki, natijada elementlar yana o‘sish tartibida joylashsin. Ya’ni S1 faylda (1 2 3) S2 faylda (4 5 6) joylashgan bo‘lishi mumkin. Yoki aksincha S1 faylda (4 5 6) S2 faylda (1 2 3) joylashgan. Har ikkala holda ham S3 fayl elementlari (1 2 3 4 5 6) tartibida bo‘lishi kerak. 3. Tarkibi A va B to‘rtburchak matritsaning elementlarini saqlovchi ikkita SA va SB nomli haqiqiy sonlar fayli berilgan. Bunda har bir faylning birinchi elementi mos matritsalarning ustunlar o‘lchamini saqlaydi. Tarkibi A va B matritsalarning ko‘paytmasidan hosil bo‘lgan matritsa elementlaridan iborat bo‘lgan, xudda o‘sha strukturadagi SC fayl hosil qilinsin. Agar A va B matritsalarni ko‘paytirish imkoni bo‘lmasa, u holda SC fayl bo‘sh qoldirilsin.
13.	1. Butun sonlar fayli berilgan. Ikkita yangi fayl hosil qilinsin. Ularning birinchisi boshlang‘ich faylning musbat sonlaridan (teskari tartibda), ikkinchisi esa manfiy sonlaridan (teskari tartibda) iborat bo‘lsin. Agar boshlang‘ich faylning manfiy yoki musbat sonlari mavjud bo‘lmasa, u holda natijaviy fayl bo‘sh holda qoldirilsin. 2. Bir xil turdagi va bir xil o‘lchamdagi SA, SB, SC.SD butun sonlar fayli va SE satr berilgan. Yangi SE nomli fayl hosil qilinsin va uning elementlari boshlang‘ich fayl elementlarining bir xil nomerlilari bilan joylashsin. 3. Tarkibi to‘rtburchak matritsaning elementlari bo‘lgan haqiqiy sonlar fayli berilgan. Bunda faylning birinchi elementi matritsaning ustunlari sonini saqlaydi. Tarkibi boshlang‘ich matritsani transponerlanganidan hosil bo‘lgan matritsa elementlarini saqlovchi huddi o‘sha strukturadagi yangi fayl hosil qilinsin.
14.	1. Haqiqiy sonlar fayli berilgan. Shu fayl elementlarining o‘rta arifmetigi topilsin. 2. Bir xil turdagi va bir xil o‘lchamdagi S_A, S_B, S_c butun sonlar fayli va S_D satr berilgan.Yangi S_D nomli fayl hosil qilinsin va unda elementlar boshlang‘ich fayl elementlari bir xil nomerlilari bilan joylashsin: A1, B1, C1, A2, B2, C2 3. Ikkita I va J butun sonlari hamda tarkibi to‘rtburchak matritsaning elementlari bo‘lgan haqiqiy sonlar fayli berilgan. Bunda faylning birinchi elementi matritsaning ustunlari sonini saqlaydi. l-satr va J- ustunda joylashgan matritsa elementi chop qilinsin (satr va ustunlar 1 dan boshlab nomerlanadi). Agar talab qilingan element mavjud bo‘lmasa, nol chop qilisin.
15.	1. Haqiqiy sonlar fayli berilgan. Shu faylning juft nomerdagi elementlari yig‘indisi hisoblansin. 2. Bir xil turdagi ikkita fayl berilgan. Birinchi fayl tarkibiga ikkinchi faylniki, ikkinchi fayl tarkibiga birinchi fayl tarkibi qo‘shilsin.

	3. A va B kvadrat matritsaning elementlarini saqlovchi ikkita SA va SB nomli haqiqiy sonlar fayli berilgan. Tarkibi A va B matritsaning ko‘paytmasidan hosil bo‘lgan matritsa elementlaridan iborat bo‘lgan SC nomli yangi fayl hosil qilinsin. Agar A va B matritsani ko‘paytirish mumkin bo‘lmasa, u holda SC fayl bo‘sh holda qoldirilsin.
16.	1. Butun sonlar fayli berilgan. Shu fayl tarkibiga kiruvchi seriyalar soni topilsin Seriya deb, ketma - ket kelgan bir xil elementlar guruhiga aytiladi. Masalan, 1, 5, 5, 5, 4, 4, 5 elementga ega bo‘lgan fayl uchun natija 4. 2. Butun N soni va SO satri berilgan (N<=4) va bir xil turdagi N ta fayl berilgan. S1 SN fayllar nomlari. Shu fayllar tarkibi yangi SO nomli faylda birlashtirilsin (tartibini buzmagan holda). 3. Kvadrat matritsa elementlaridan iborat bo‘lgan haqiqiy sonlar fayli berilgan. Boshlang‘ich matritsaning tronspanerlanganidan hosil bo‘lgan matritsaning elementlaridan iborat bo‘lgan yangi fayl hosil qilinsin. $\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix}^T = \begin{bmatrix}1 & 3\\2 & 4\end{bmatrix}$
17.	1. Butun sonlar fayli berilgan. Boshlang‘ich faylning barcha seriyalari uzunligiga ega bo‘lgan yangi butun sonlar fayli hosil qilinsin. Seriya deb, ketma - ket kelgan bir xil elementlar guruhida aytiladi. Seriya uzunligi esa, bu elementlar soni. (seriya uzunligi 1 bo‘lishi mumkin). Masalan, 1,5,5,5,4,4,5 elementlariga ega bo‘lgan boshlang‘ich faylni elementlarini orqali yaratilgan yangi fayl tarkibi 1, 3, 2, 1 bo‘ladi. 2. Bir xil turdagi lekin o‘lchamlari turli bo‘lgan uchta fayl berilgan. Shu fayllar orasidan tarkibi eng kalta fayl bilan eng uzun tarkibdagisiga almashtirilsin. 3. Ikkita I va J butun sonlari, hamda kvadrat matritsa elementlaridan iborat bo‘lgan haqiqiy sonlar fayli berilgan. I - satr va J - ustunda joylashgan matritsa elementi chop qilinsin. (satr va ustunlar 1 dan boshlab nomerlanadi). Agar talab qilingan element mavjud bo‘lmasa, u holda nol qiymati chop qilisin.
18.	1. Haqiqiy sonlar fayli berilgan. Shu faylning birinchi lokal minimumi topilsin. (lokal minimum deb o‘z qo‘shnilaridan kichik bo‘lgan elementga aytiladi). 2. Bir xil turdagi lekin o‘lchamlari turli bo‘lgan uchta fayl berilgan. Shu fayllar orasidan tarkibi eng uzun fayl bilan eng kalta tarkibdagisiga almashtirilsin. 3. Tarkibida "kun/oy/yil" formatdagi sanalar bo‘lgan satrli fayl beilgan. Yangi satr fayli hosil qilinsin va boshlang‘ich fayldagi sanalar kamayish tartibida joylashtirilsin.
19.	1. Haqiqiy sonlar fayli berilgan. Shu faylning birinchi lokal maksimumi topilsin. (lokal maksimum deb o‘z qo‘shnilaridan katta bo‘lgan elementga aytiladi). 2. Ixtiyoriy fayl berilgan. Shu fayl nusxasi yangi nom bilan yaratilsin. 3. Tarkibida "kun/oy/yil" formatdagi sanalar bo‘lgan satrli fayl berilgan. Shu fayldan tarkibida kuzning eng kech sanasi bo‘lgan satr topilsin. Agar talab qilingan sana faylda mavjud bo‘lmasa, u holda bo‘sh satr chop qilinsin.

20.	1. Haqiqiy sonlar fayli berilgan. Shu fayldagi umumiy lokal ekstremumlari soni topilsin. (Ya'ni lokal minimumlar va lokal maksimumlarning umumiy soni. Lokal minimum deb o‘z qo‘shnilaridan kichik bo‘lgan elementga, lokal maximum esa o‘z qo‘shnilaridan katta bo‘lgan elementga aytiladi. 2. Ixtiyoriy ikkita fayl berilgan. Fayl ichidagi ma'lumotlar almashtirilsin. 3. Tarkibida "kun/oy/yil" formatdagi sanalar bo‘lgan satrli fayl berilgan. Shu fayldan tarkibida bahorning eng erta sanasi bo‘lgan satr topilsin. Agar talab qilingan sana faylda mavjud bo‘lmasa, u holda bo‘sh satr chop qilinsin.
21.	1. Haqiqiy sonlar fayli berilgan. Boshlang‘ich faylning barcha lokal maksimumlarining nomerlarini o‘sish tartibida joylashgan holda saqlovchi yangi butun sonlar fayli hosil qilinsin. (lokal maximum deb o‘z qo‘shnilaridan katta bo‘lgan elementga aytiladi). 2. Butun sonlar fayli berilgan. Undagi barcha musbat sonlarni uchta nolga almashtirilsin. 3. Tarkibida "kun/oy/yil" formatdagi sanalar bo‘lgan satrli fayl berilgan. Tarkibida boshlang‘ich fayldagi barcha qishki sanalarni saqlovchi yangi satrli fayl hosil qilinsin (teskari tartibda). Agar qo‘yilgan shartdagi sanalar boshlang‘ich faylda mavjud bo‘lmasa, u holda natijaviy fayl bo‘sh holda qoldirilsin.
22.	1.Haqiqiy sonlar fayli berilgan. Boshlang‘ich faylning barcha lokal ekstremumlarining nomerlarini kamayish tartibida joylashgan holda saqlovchi yangi butun sonlar fayli hosil qilinsin. (lokal ekstremumi deb lokal minimum va lokal maximumlar soniga aytiladi). 2. Butun sonlar fayli berilgan. Undagi juft nomerdagi elemetlar ikkita nolga almashtirilsin. 3. Tarkibida "kun/oy/yil" formatdagi sanalar bo‘lgan satrli fayl berilgan. Tarkibida boshlang‘ich fayldagi barcha yozgi sanalarni saqlovchi yangi satrli fayl hosil qilinsin (tartibini saqlagan holda). Agar qo‘yilgan shartdagi sanalar boshlang‘ich faylda mavjud bo‘lmasa, u holda natijaviy fayl bo‘sh holda qoldirilsin.
23.	1. Haqiqiy sonlar fayli berilgan. Boshlang‘ich faylning kamayib boruvchi elementlar ketma-ketliklari uzunligiga ega bo‘lgan yangi butun sonlar fayli hosil qilinsin. Masalan, 1.7, 4.5, 3.4, 2.2 elementlariga ega bo‘lgan boshlang‘ich fayl uchun natijaviy yaratilgan fayl tarkibi quyidagicha bo‘ladi: 3, 2. 2. Butun sonlar fayli berilgan. Undagi barcha toq nomerdagilari ikkilantirilsin. 3. Tarkibida "kun/oy/yil" formatdagi sanalar bo‘lgan satrli fayl berilgan. Ikkita butun sonlar fayli hosil qilinsin va ularning birinchisida boshlang‘ich satrli fayldagi oylar, ikkinchisida esa shu oylar uchun yillar qiymatlari saqlansin (tartibni saqlagan holda).
24.	1. Haqiqiy sonlar fayli berilgan. Boshlang‘ich fayl elementlarining barcha monoton ketma-ketliklariga ega bo‘lgan yangi butun sonlar fayli hosil qilinsin.

	Masalan, 1.7, 4.5, 3.4, 2.2, 8.5, 1.2 elementlariga ega bo‘lgan boshlang‘ich fayl uchun yaratilgan natijaviy fayl tarkibi quyidagicha bo‘ladi: 2, 3, 2, 2. 2. Butun sonlar fayli berilgan. Shu fayl oxiriga boshlang‘ich elementlarni yozish orqali fayl o‘lchami 2 martta orttirilsin.(teskari tartibda) 3. Tarkibida "kun/oy/yil" formatdagi sanalar bo‘lgan satrli fayl berilgan. Bunda kun va oy ikkita o‘rinni, yil esa 4 ta o‘rinni egallaydi (masalan, "05/07/2012"). Ikkita butun sonlar fayli hosil qilinsin va ularning birinchisi boshlang‘ich satrli fayldagi kunlar qiymatini, ikkinchisi esa shu kunlar uchun oylar qiymatini saqlasin. (tartibini saqlagan holda).
25.	1. Haqiqiy sonlar fayli berilgan. Undagi barcha elementlarni kvadratlariga almashtirilsin. 2. Butun sonlar fayli berilgan. Shu fayl oxiriga boshlang‘ich elementlarni yozish orqali fayl o‘lchami 2 martta orttirilsin.(tartibini buzmagan holda) 3. Satrli fayl berilgan. Boshlang‘ich fayldagi satrlarni liksografik tartibda, ya’ni satrdagi birinchi belgidan boshlab belgilarning kodlarini o‘sish tartibida joylashishidan hosil bo‘lgan yangi satrli fayl hosil qilinsin.
26.	1. Haqiqiy sonlar fayli berilgan. Undadi eng kata va eng kichik elementlar o‘rinlari almashtirilsin. 2. 50ta elementdan kam bo‘lgan butun sonlar fayli berilgan. Shu faylning boshiga kerakli nollar yozish orqali elementlar miqdorini 50tagacha ochirilsin. 3. Satrli fayl berilgan. Boshlang‘ich fayl tarkibidagi barcha eng katta uzunlikka ega bo‘lgan satrlarni saqlovchi yangi satrli fayl hosil qilisin. (Tartibini saqlagan holda).
27.	1. A1,A2,...An(n fayldagi elementlar soni) elementlardan iborat butun sonlar fayli berilgan. Shu faylning boshlang‘ich joylashishini elementlarning quyidagi joylashishiga almashtirilsin. A1,An,A2,An-1,A3... 2. Butun sonlar fayli berilgan. Undagi barcha manfiy sonlar o‘chirilsin. 3. Satrli fayl berilgan. Boshlang‘ich fayl tarkibidagi barcha eng kichik uzunlikka ega bo‘lgan satrlarni saqlovchi yangi satrli fayl hosil qilisin. (Tartibini saqlagan holda).
28.	1. Haqiqiy sonlar fayli berilgan. Fayldagi oxirgi va birinchi turgan elementdan boshqa barcha elementlarini o‘zidan oldingi va keying turgan elementlarning o‘rta arifmetigiga almashtirilsin. 2. Butun sonlar fayli berilgan. Undagi barcha juft nomerdagi elementlari o‘chirilsin. 3. Butun K (>0) soni va satrli fayl berilgan. 2 ta yangi fayl hosil qiling. 1 - fayl satrli, 2 - fayl belgili. Satrli fayl boshlang‘ich faylning har bir satridan dastlabki K ta belgini o‘zida saqlasin. Belgili fayl esa, faqat K - belgini o‘zida saqlasin. Qaralayotgan satr belgilari soni K dan kichik bo‘lsa, satrli faylga to‘liq yozilsin. Belgili faylga probel yozilsin.
29.	1. 50 ta elementdan ortiq bo‘lgan elementlardan iborat butun sonlar fayli berilgan. Shu fayl elementlarini oxiridan boshlab 50 ta elementgacha o‘chirilsin.

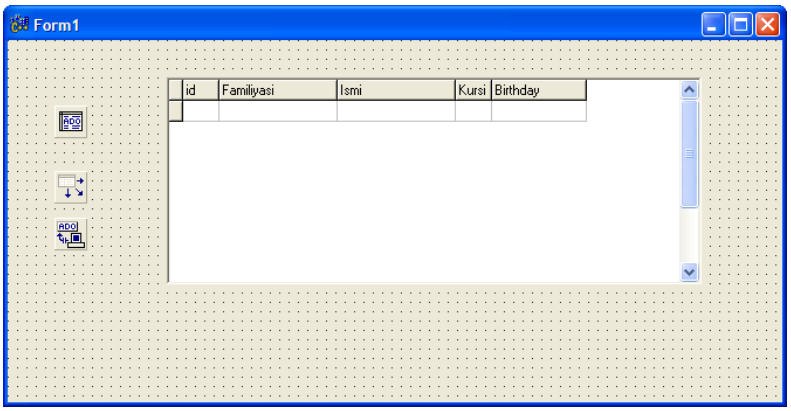
	2. Juft miqdordagi elementlarga ega bo‘lgan butun sonlar fayli berilgan. Shu faylning birinchi yarmi o‘chirib tashlansin. 3. Belgili fayl berilgan. Undagi elementlar kodlari o‘sish tartibida joylashgan holda tartiblansin.
30.	1. Juft miqdordagi elementlarga ega bo‘lgan butun sonlar fayli berilgan. Shu faylning ikkinchi yarmi o‘chirib tashlansin. 2. 50 ta elementdan ortiq bo‘lgan elementlardan iborat butun sonlar fayli berilgan. Shu fayl elementlarini boshidan boshlab 50ta elementgacha o‘chirilsin. 3. Hech bo‘lmaganda bitta probel belgisi mavjud bo‘lgan belgili fayl berilgan. Shu fayldagi oxirgi kelgan probeldan oldin joylashgan barcha elementlar o‘chirilsin. (Shu probelni ham hisobga olgan holda)

Ma’lumotlar bazasini tashkil etish hamda uni Builderda bog‘lash

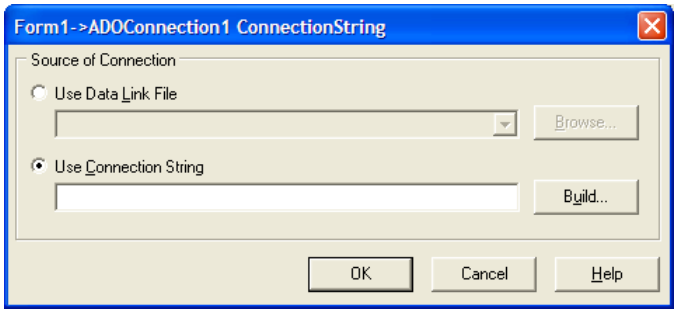
Ma’lumotlar bazasi bilan ishlovchi dasturlar judayam ko‘p. Ularni ishlatilish sohasi bo‘yicha bir nechta guruhlarga bo‘lish mumkin. Bu yerda biz uchun qulay bo‘lgan Access 2000 da Talaba.mdb fayl yaratamiz va undagi Talaba jadvalini Builder muhitida ulashni ko‘rib chiqamiz.

Avvalambor, Talaba.mdb faylini yaratib unda jadval yaratamiz va bu ma’lumotlar bazasini Builderdagi loyihamiz yaratilgan papkaga saqlaymiz.

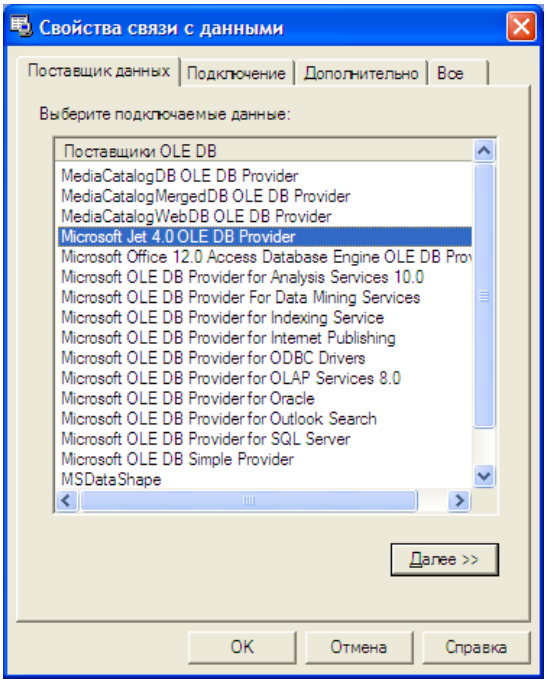
1. Builderda Access bilan ulanish uchun Standart panelidagi ADO panelidan ADOConnection1 va ADOTable1 komponentasini, DataAccess panalidan DataSource1 ni, DataControl panelidan DBGrid1 komponantasini tanlaymiz va ularni formaga tashlaymiz.



2. ADOConnection1 ni ustiga sichqonchani ikki marta bosish orqali uning ConnectionString xususiyati ochiladi.

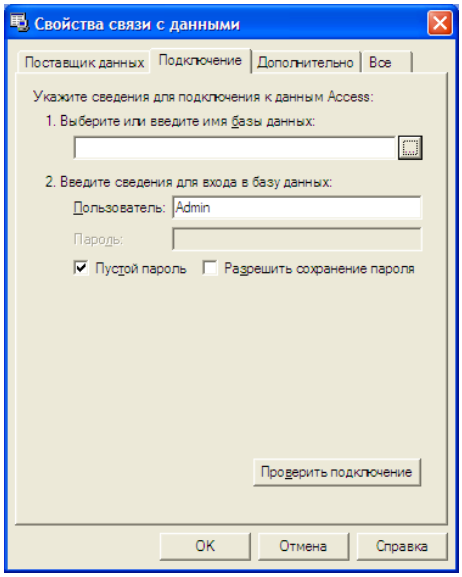


3. Undan Build tugmasini bosamiz.



4. Microsoft.Jet.OLEDB.4.0 Provider ni tanlaymiz.

5. So‘ngra (…) tugmasi orqali Accesda yaratilgan fayl ko‘rsatiladi hamda uning ulanganligi “Проверить подключение” tugmasi bilan tekshiriladi. Agarda “Проверка подключение выполнена” natijasi chiqsa ulangan sanaladi.



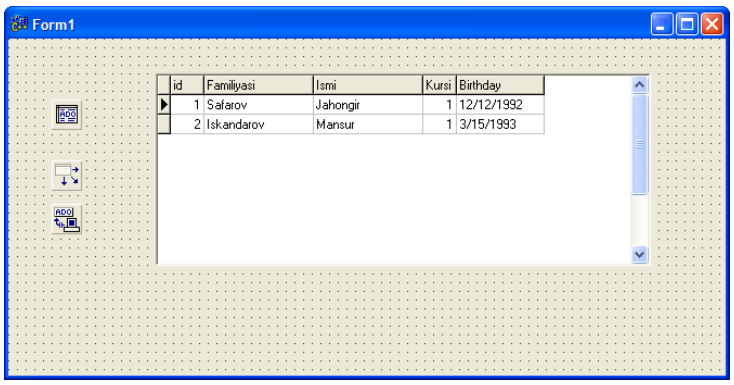
6. Endi ma’lumotlar bazasidagi Talaba jadvalini ulash lozim. Buning uchun ADOTable1 ning Connection xususiyatiga ADOConnection1 tanlanadi.

7. So‘ngra TableName xususiyatidan kerakli jadval (talaba) tanlanadi. Agarda UserName va Password so‘ralsa OK deyish lozim.

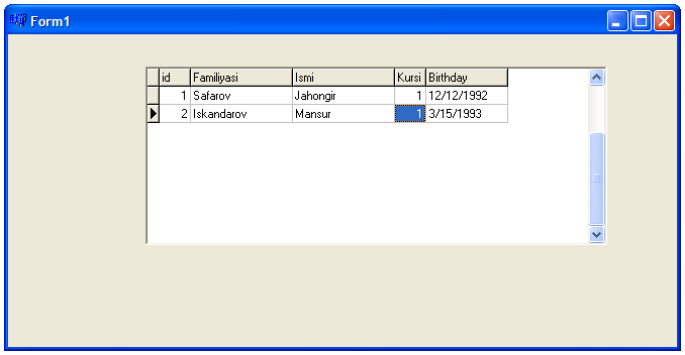
8. Albatta ADOTable1ni Active xususiyatini True qilish lozim.

9. Endi DataSource1 ni DataSet xususiyatidan ADOTable1ni tanlaymiz.

10. Eng oxirida DBGrid1 ni DataSource xusisiyatiga DataSource1 ni tanlaymiz hamda barcha bajarilgan amallarni saqlaymiz.



11.F9 tugmasi yordamida loyihani ishga tushiramiz.



3.1-jadvalda berilgan masalalarni Accesda ma’lumotlar bazasini tuzib Builder bilan bog‘lang. Jadvalga kamida 20 ta satr kiriting hamda har bir ustun bo‘yicha saralash amallarini Builderda bajaring (Masalan, har bir ustun uchun alohida tugmacha qo‘yib saralashni amalga oshirish mumkin.).

Nazorat savollari

1. Fayl deb nimaga aytiladi ?
2. Fayllar nechaga bo‘linadi?
3. Fayllar nima uchun ochiladi?
4. Fayl qanday yopiladi?
5. Fayllar ustida qanday amallarni bajarish mumkin?
6. Fayl bilan ishlovchi funksiyalar uchun qaysi kutubxonadan foydalaniladi?
7. Binar fayl nima?
8. Fayllar necha xil holda bo‘lishi mumkin?

2. MA’LUMOTLAR TUZILMASI VA ALGORITMLAR UCHUN AMALIY MASHG‘ULOTLAR

2.1. 4-amaliy mashg‘ulot. Vektorlar, massivlar, yozuvlar va jadvallar

Vektor (vector) – bu S++ standart kutubxonasidan olingan shablon yoki namuna, erkin holda murojaatni amalga oshiruvchi dinamik massivlarda foydalaniladi. Vektorlar ma’lumotlar tuzilmasidan bo‘lib, ro‘yxat va DEK kabi konteyner xususiyatiga ega.

Vector bosh faylda joylashdi, ya’ni #include<vector>.standart komponetalar kabi, std nomlar tarkibida bo‘ladi. Vektorni asosiy xusuiyati massiv elementiga erkin murojaatni amalga oshirish bilan birga vektor o‘z o‘lchamini avtomatik tarzda o‘zgartira oladi (elementlarni kiritish yoki o‘chirish orqali).

Vektorni barcha elementlari bitta toyifaga (int yoki float yoki char) tegishli bo‘lishi kerak. Vektorlarni dastur tarkibida shakllantirish quyidagicha:

```
vector<int> myVector;      // elementlari toyifasi int bo‘lgan bo‘sh vektor
vector<float> myVector(10) // 10 ta elementli, toyifasi float bo‘lgan vektor
vector<char> myVector(5, ' ') // 5 ta bo‘sh joydan iborat vektor
```

misol:

#include <iostream>
#include <vector>
#include <conio.h>
using namespace std;
int main()
{ vector <int> v; v.push_back(8); v.push_back(9);
vector<vector<int> > A; A.push_back(v); A.push_back(v);
A.push_back(v);
for (int i=0; i<3; i++) { for (int j=0; j<2; j++)
cout<< A[i][j] << " "; cout <<endl;}
getch();
return 0; }
	Natija:
	8 9
	8 9
	8 9

Massivlar quyidagi turlarga bo‘linadi:

1) bir o‘lchovli massivlar;

2) ikki o‘lchovli massivlar;
3) qatorli yoki belgi massivlar;
4) dinamik massivlar.

Bir o‘lchovli massivlarga misollar:

1- misol. Massivni elementlarini yig‘indisini toping. Misolni echilishi quyidagicha:

```
#include<iostream>
using std::cout;
#include<conio.h>
main()
{
  float s,j;
  int i;
  s=0;
  float a[5]={2,3,4,-1,-5};
  for (i=0; i<5; ++i)
  s = s + a[i];
  cout<<"\n s = "<<s;
   getch();
  return 0;
}
```

Natija:

S = 3

2- misol. Bir o‘lchamli n ta bir xil toyifali elementdan iborat massiv berilgan. Quyidagilarni bajaring:

a) manfiy elementlarning yig‘indisini;

b) maksimal va minimal qiymatli elementlar orasida joylashgan elementlardan yangi massiv tuzing;

v) massiv elementlarini o‘sish tartibida joylashtiring.

Misolni dasturi:

```
#include<iostream>
using namespace std;
int main()
{
  int x,y,n,h,a[1000],min,max;
  cin >> n;
  for(int i=0;i<n;i++)
  cin >> a[i];
  int s=0;
for(int i=0;i<n;i++){ if(a[i]<0) s+=a[i];}
  if(s!=0){ cout << "Manfiy elementlar yigindisi:";
cout << endl; cout << s; cout << endl;  }
    else
   {  cout << "Manfiy elementlari yoq;";cout << endl;  }
 min=a[0];x=0;
 for(int i=0;i<n;i++) {
   if(a[i]<min)
     x=i; }
  max=a[0];y=0;
 for(int i=0;i<n;i++){
  if(a[i]>max)
  y=i;
  }
```

```
cout << "maksimal va minimal elementlar orasidagi sonlar:";
 if(y>x){
 for(int i=x+1;i<y;i++){
cout << endl;
cout << a[i]; } }
else
{
for(int i=y+1;i<x;i++){
cout << endl;
cout << a[i];}
 }
cout << endl;
  min=a[0];int m=1;
  for(int j=0;j<n;j++)
  { for(int i=m;i<n;i++){
   if(min>=a[i]){
   h=min;
   min=a[i];a[i]=h;
   }    }
   a[j]=min;
   min=a[j+1];
   m++;cout <<a[j];
  }
  return 0;
  }
```

3-misol. Bir o‘lchovli **n** ta elementdan iborat massiv berilgan. Quyidagini hisoblang:

a) minimal element raqamini aniqlang;

b) birinchi va ikkinchi raqami manfiy elementlar orasidagi sonlar yig‘indisini;

v) massivni shunday almashtiringki, dastlabki elementlari moduli 1 dan kichik bo‘lsin, keyin qolganlarini joylashtiring.

Misolni echish:

```
#include<iostream>
#include<cmath>
using namespace std;
int main(){
  float a[1000], min;int n,h;
  cin >> n;
  for(int i=0;i<n;i++)
  cin >>a[i];
min=a[0];

 for(int i=0;i<n;i++){
   if(min>=a[i])
   h=i;
   }
   cout <<"eng kicik element numeri:" << h <<endl;
   float s=0;
   for(int i=0;i<n;i++){
   if(a[i]<0){i++;
   while(a[i]>0){
     s+=a[i];
     i++;}  }  }
     cout <<"manfiy sonlar orasidagi sonlar yigindisi"<<s<<endl;
```

```
  for(int i=0;i<n;i++){
    if(fabs(a[i]<1))
    cout<<a[i]<<" ";}
    for(int i=0;i<n;i++){
    if(fabs(a[i]!=1))
    cout<<a[i]<<" ";
    }
 return 0;
 }
```

№4 -topshiriq

1 – variant. A(5) massiv berilgan. Juft elementlarining miqdori va yig‘indisini toping.

2 – variant. A(6) massiv berilgan. Massivning eng kichik elementini va uning tartib raqamini toping.

3 – variant. A(4) massiv berilgan. Masssivning eng katta elementini va tartib raqamini toping.

4 – variant. A(8) massiv berilgan. Massivning eng kichik elementini toping.

5 – variant. A(6) massiv berilgan. Massivni yig‘indisini, juft elementlar miqdorini, eng kichik va eng katta elementlar o‘rnini toping.

6 – variant. A(21) massiv berilgan. Massivning eng katta elementini toping.

7 – variant. A(17) massiv berilgan. Uning minimal elementini toping.

8 – variant. A(9) massiv berilgan. Massivning elementlarini kamayish tartibida joylashtiring.

9 – variant. A(12) massiv berilgan. Massivning elementlarini o‘sish tartibida joylashtiring.

10 – variant. A(11) massiv berilgan. Massivning manfiy elementlarini miqdorini aniqlang.

11 – variant. A(8) massiv berilgan. Massivning musbat elementlarini miqdorini aniqlang.

12 – variant. A(8) massiv berilgan. Massivning musbat toq elementlarini miqdorini aniqlang.

13 – variant. A(16) massiv berilgan. Massivning musbat juft elementlarini miqdorini aniqlang.

14 – variant. A(15) massiv berilgan. Massivning musbat juft elementlarini yig‘indisini aniqlang.

15 – variant. A(13) massiv berilgan. Massivning musbat toq elementlarini yig‘indisini aniqlang.

16 – variant. A(22) massiv berilgan. Massivning musbat elementlarini ko‘paytmasini aniqlang.

17 – variant. A(6) massiv berilgan. Massivning 4 ga teng elementlarining miqdorini aniqlang.

18 – variant. A(5) va A(10) massiv berilgan. Massivning bir xil elementlarini yig‘indisini aniqlang.

19 – variant. A(5) va A(10) massiv berilgan. Massivning bir xil bo‘lmagan elementlarini yig‘indisini aniqlang.

20 – variant. A(6) massiv berilgan. Massivning 4 dan kichik elementlari miqdorini aniqlang.

21 – variant. A(9) massiv berilgan. Massivning manfiy elementlari miqdorini aniqlang.

22 – variant. A(22) massiv berilgan. Massivning musbat elementlarini miqdori va yig‘indisini aniqlang.

23–variant. A(30) massiv berilgan. Massivning musbat elementlarini orasidan juft hamda eng kichik elementni aniqlang.

24–variant. A(30) massiv berilgan. Massivning manfiy elementlarini orasidan toq hamda eng katta elementni aniqlang.

25–variant. A(30) massiv berilgan. Massivning elementlarini orasidan eng kattasini topish va qolgan elementlarni “nol”ga tenglash dasturini tuzing.

26–variant. Beshta familiyadan iborat massiv tuzing va uning elementlarini oxiridan boshlab ustun shaklida bosmaga chiqaring.

27–variant. Yigirmata familiyadan iborat massiv tuzing va klaviaturadan kiritilayotgan harfdan boshlanuvchi familiyani ekranga chiqaruvchi dasturni tuzing.

Bir o‘lchovli massivlar va ko‘rsatkichlar

№5-topshiriq

1–variant. Sonli elementlardan tashkil topgan bir hadli massivda quyidagi amallar hisoblansin:

1) massivning manfiy elementlar yig‘indisi;
2) massivning maksimal va minimal elementlari orasidagi ko‘paytmasi.
   Massiv o‘sish tartibida joylashtirilsin.

2–variant. Sonli elementlardan tashkil topgan bir hadli massivda quyidagi amallar hisoblansin:

1) massivning musbat elementlar yig‘indisi;

2) massivning maksimal va minimal elementlarining modullari orasidagi ko‘paytmasi.

Massiv kamayish tartibida joylashtirilsin.

3–variant. Sonli elementlardan tashkil topgan bir hadli massivda quyidagi amallar hisoblansin:

1) juft o‘rindagi massiv elementlarining ko‘paytmasi;
2) birinchi va oxirgi nol qiymatlari orasidagi massiv elementlarining yig‘indisi.

Massiv shunday joylashtirilsinki, avval massivning musbat elementlari, so‘ngra manfiy elementlari joylashtirilsin (0 ga teng bo‘lgan massiv elementlari musbat deb hisoblansin).

4–variant. Sonli elementlardan tashkil topgan bir hadli massivda quyidagi amallar hisoblansin:

1) toq o‘rindagi massiv elementlarining yig‘indisi;
2) birinchi va oxirgi manfiy qiymatlari orasidagi massiv elementlarining yig‘indisi.

Massivdan moduli 1 dan oshmagan elementlarini o‘chirib, uni qisqartiring. Bo‘shagan massiv elementlarini 0 bilan to‘ldiring.

5–variant. Sonli elementlardan tashkil topgan bir hadli massivda quyidagi amallar hisoblansin:

1) massivning maksimal elementi topilsin;
2) massivning oxirgi musbat elementigacha bo‘lgan elementlarning yig‘indisi.

Massivdan moddullari [a, b] intervalida joylashgan elementlarni o‘chirib, uni qisqartiring. Bo‘shagan massiv elementlarini oxirini 0 bilan to‘ldiring.

6–variant. Sonli elementlardan tashkil topgan bir hadli massivda quyidagi amallar hisoblansin:

1) massivning minimal elementi topilsin;
2) massivning birinchi va oxirgi musbat elementlari orasidagi summasi.

Massiv shunday joylashtirilsinki, avval uning 0 ga teng elementlari, so‘ngra qolgan barcha elementlari joylashsin.

7– variant. Sonli elementlardan tashkil topgan bir hadli massivda quyidagi amallar hisoblansin:

1) massivning maksimal elementining tartib raqami topilsin;
2) massivning birinchi va ikkinchi 0 ga teng bo‘lgan elementlari orasida joylashgan elementlarning ko‘paytmasi topilsin.

Massiv shunday joylashtirilsinki, uning birinchi yarmida massivning toq elementlari, qolgan yarmida juft elementlari joylashsin.

8–variant. Sonli elementlardan tashkil topgan bir hadli massivda quyidagi amallar hisoblansin:

1) massivning minimal elementining tartib raqami topilsin;
2) massivning birinchi va ikkinchi manfiy qiymatlari orasidagi elementlarning summasi topilsin.

Massiv shunday joylashtirilsinki, avval massiv elementlarining moduli 1 dan oshmaganlari, so‘ngra qolgan barcha elementlar joylashsin.

9– variant. Sonli elementlardan tashkil topgan bir hadli massivda quyidagi amallar hisoblansin:

1) massivning modul bo‘yicha maksimal elementi topilsin;
2) massivning birinchi va ikkinchi musbat qiymatlari orasidagi elementlarning summasi topilsin.

Massiv shunday joylashtirilsinki, massivning 0 ga teng bo‘lgan barcha elementlari massivning oxirida joylashsin.

10–variant. Sonli elementlardan tashkil topgan bir hadli massivda quyidagi amallar hisoblansin:

1) massivning modul bo‘yicha minimal elementi topilsin;

2) massivning 0 ga teng bo‘lgan birinchi elementidan so‘ng, qolgan barcha elementlarining moduldagi summasi.

Massiv shunday joylashtirilsinki, uning birinchi yarmida massivning juft elementlari, qolgan yarmida toq elementlari joylashsin.

11–variant. Sonli elementlardan tashkil topgan bir hadli massivda quyidagi amallar hisoblansin:

1) massivning modul bo‘yicha minimal elementining tartib raqami topilsin;
2) massivning birinchi manfiy elementidan so‘ng qolgan barcha elementlarning moduldagi summasi hisoblansin.

Massivni [a, b] interval oralig‘ida joylashgan barcha elementlarini o‘chirib, uni qisqartiring. Bo‘shagan massiv elementlarini oxirini 0 bilan to‘ldiring.

12–variant. Sonli elementlardan tashkil topgan bir hadli massivda quyidagi amallar hisoblansin:

1) massivning modul bo‘yicha maksimal elementining tartib raqami topilsin;
2) massivning birinchi musbat elementidan so‘ng qolgan barcha elementlarning moduldagi summasi hisoblansin.

Massiv shunday joylashtirilsinki, avval massivning [a, b] interval oralig‘ida joylashgan elementlarining butun qismi, so‘ngra qolgan barcha elementlar joylashsin.

13 – variant. Sonli elementlardan tashkil topgan bir hadli massivda quyidagi amallar hisoblansin:

1) A dan V gacha bo‘lgan diapazon oralig‘ida joylashgan massiv elementlarining miqdori topilsin;
2) massivning maksimal elementidan so‘ng joylashgan barcha elementlarning summasi topilsin.

Massivning moduldagi elementlarini kamayish tartibida joylashtiring.

14–variant. Sonli elementlardan tashkil topgan bir hadli massivda quyidagi amallar hisoblansin:

1) 0 ga teng bo‘lgan massiv elementlarining miqdori topilsin;
2) massivning minimal elementidan so‘ng joylashgan barcha elementlarning summasi topilsin.

Massivning moduldagi elementlarini o‘sish tartibida joylashtiring.

15–variant. Sonli elementlardan tashkil topgan bir hadli massivda quyidagi amallar hisoblansin:

1) S dan katta bo‘lgan massiv elementlarining miqdori topilsin;
2) massivning modul bo‘yicha maksimal elementidan so‘ng joylashgan barcha elementlarining ko‘paytmasi hisoblansin.

Massiv shunday joylashtirilsinki, avval massivning manfiy elementlari, so‘ngra musbat elementlari joylashtirilsin (0 ga teng bo‘lgan massiv elementlari musbat deb hisoblansin).

16–variant. Sonli elementlardan tashkil topgan bir hadli massivda quyidagi amallar hisoblansin:

1) massivning manfiy bo‘lgan elementlarining umumiy miqdori topilsin;
2) massivning modul bo‘yicha minimal elementidan so‘ng joylashgan barcha elementlarining summasi hisoblansin.

Massivning barcha manfiy elementlarini shu elementlarning kvadratlariga o‘zgartirib, massivni o‘sish tartibida joylashtiring.

17–variant. Sonli elementlardan tashkil topgan bir hadli massivda quyidagi amallar hisoblansin:

1) massivning musbat bo‘lgan elementlarining umumiy miqdori topilsin;
2) massivning oxirgi 0 ga teng bo‘lgan elementidan so‘ng joylashgan barcha elementlarning summasi hisoblansin.

Massiv shunday joylashtirilsinki, avval massivning butun qismi 1 dan oshmagan elementlari, so‘ngra qolgan barcha elementlari joylashsin.

18–variant. Sonli elementlardan tashkil topgan bir hadli massivda quyidagi amallar hisoblansin:

1) S dan kichik bo‘lgan massiv elementlarining miqdori topilsin;
2) massivning oxirgi manfiy elementidan so‘ng joylashgan barcha elementlarini butun qismining summasi hisoblansin.

Massiv shunday joylashtirilsinki, avval massivning maksimal elementidan 20% ga farq qiladigan elementlari, so‘ngra qolgan barcha elementlari joylashsin.

19–variant. Sonli elementlardan tashkil topgan bir hadli massivda quyidagi amallar hisoblansin:

1) massivning manfiy elementlarini ko‘paytmasi topilsin;
2) massivning maksimal elementigacha bo‘lgan musbat elementlarining summasi hisoblansin.

Massivni teskari tartibda joylashtiring.

20–variant. Sonli elementlardan tashkil topgan bir hadli massivda quyidagi amallar hisoblansin:

1) massivning musbat elementlarini ko‘paytmasi topilsin;
2) massivning minimal elementigacha bo‘lgan elementlarining summasi hisoblansin.

Massivning adohida juft elementlarini, alohida toq elementlarini o‘sish tartibida joylashtiring.

Ikki o‘lchovli massivlar

№6 - topshiriq

1–variant. Berilgan butun sonli to‘g‘ri burchakli matritsadan quyidagilar aniqlansin:

1) tarkibida bironta 0 ga teng elementi bo‘lmagan satrlarning umumiy miqdori aniqlansin;
2) berilgan sonlardan bir martadan ko‘p qatnashganlari orasidan maksimali topilsin.

2–variant. Butun sonli to‘g‘ri burchakli matritsa berilgan. Tarkibida bironta ham 0 ga teng elementi bo‘lmagan ustunlarning umumiy miqdori aniqlansin.

3–variant. Berilgan butun sonli to‘g‘ri burchakli matritsadan quyidagilar aniqlansin:

1) kamida bitta 0 ga teng elementi bo‘lgan ustunlarning umumiy miqdori aniqlansin;
2) eng ko‘p bir xil elementlardan tashkil torgan satrning tartib raqami aniqlansin;

4–variant. Berilgan butun sonli kvadrat matritsadan quyidagilar aniqlansin:

1) manfiy elementlari mavjud bo‘lmagan satrlarning elementlaridan ko‘paytma olinsin;
2) matritsaning asosiy parallel dioganallari orasidagi elementlar summasining maksimumi topilsin.

5–variant. Berilgan butun sonli kvadrat matritsadan quyidagilar aniqlansin:

1) manfiy elementlari mavjud bo‘lmagan ustunlar elementlarining ko‘paytma olinsin;
2) matritsaning ikkilamchi parallel diagonallari orasidagi moduldagi elementlar yig‘indisining minimumi topilsin.

6–variant. Berilgan butun sonli to‘g‘ri burchakli matritsadan quyidagilar aniqlansin:

1) kamida bitta manfiy qiymatli elementga ega qatorlarning yig‘indisi hisoblansin;
2) ustun va qatorlarning soni teng matritsa bo‘lganida.

Eslatma: A matritsa

7–variant. Berilgan (8×8) o‘lchamli matritsadan shunday k lar topilsinki, matritsaning k-r qatori uning k-m ustuni bilan mos tushsin.

Kamida bitta manfiy qiymatli elementga ega qatorlarning summasi hisoblansin.

8–variant. Matritsani 0 bilan to‘lgan qatorlarini va ustunlarini o‘chirib, u ixchamlashtirilsin.

Kamida bitta musbat elementga ega bo‘lgan matritsa qatorlarining birinchisi aniqlansin.

9–variant. Butun sonli to‘g‘ri burchakli matritsaning qatorlaridagi bir xil qatnashgan elementlarning umumiy miqdoriga qarab, matritsaning qatorlari o‘sish tartibida joylashtirilsin.

Bironta ham manfiy elementlari mavjud bo‘lmagan matritsa ustunlarining birinchisi aniqlansin.

10–variant. Berilgan butun sonli to‘g‘ri burchakli matritsadan quyidagilar aniqlansin:

1) kamida bitta 0 qiymatiga ega qatorlarning umumiy miqdori topilsin;
2) eng uzun bir xil elementlardan tashkil topgan ustunning tartib raqami aniqlansin.

11–variant. Berilgan butun sonli kvadrat matritsadan quyidagilar aniqlansin:

1) manfiy elementlari bo‘lmagan qatorlardagi elementlarning umumiy summasi aniqlansin;
2) matritsani asosiy parallel diogonallari orasidagi elementlar yig‘indisining minimumi topilsin.

12–variant. Berilgan butun sonli to‘g‘ri burchakli matritsadan quyidagilar aniqlansin:

1) kamida bitta elementi 0 ga teng bo‘lgan qatorlarda joylashgan manfiy elementlarning umumiy soni topilsin;
2) ustun va qatorlarning soni tayanch nuqtallarga ega matritsa bo‘lganida.

Eslatma: A matritsa

13–variant. Natural n va m hamda haqiqiy sonli A(1:n, 1:m) massiv berilgan bo‘lsin. SHu massivning har bir satridagi eng katta elementlar ichidan eng kichigini toping.

14–variant. n butun son va haqiqiy sonli B(1:n, 1:n) massiv berilgan. Uning bosh va qarama-qarshi diagonallaridagi elementlar yig‘indisini hisoblang.

15–variant. Natural n va m hamda haqiqiy sonli A(1:n, 1:m) massiv berilgan bo‘lsin. SHu massivning har birinchi elementlari musbat bo‘lgan ustundagi elementlar yig‘indisi hisoblansin.

16–variant. Natural n va m hamda haqiqiy sonli A(1:n, 1:m) massiv berilgan bo‘lsin. SHu massivning har birinchi elementlari musbat bo‘lgan satrdagi elementlar ko‘paytmasini hisoblansin.

17–variant. Butun sonli A(1:20, 1:10) massivda necha xil elementlar uchrashini aniqlang.

18–variant. Butun sonli A(1:20, 1:10) massivda bir xil elementlar mavjud yoki mavjud emasligini aniqlang.

19–variant. n butun son va haqiqiy sonli B(1:n, 1:n) massiv berilgan. Quyidagi ifodani hisoblang: $x_1y_1 + \cdots + x_ny_n$. Bu erda x_i-B ning i-satridagi eng katta element, y_j esa B ning j- ustinidagi eng kichik element.

20–variant. Natural n va m hamda haqiqiy sonli A(1:n, 1:m) massiv berilgan. SHu massivning eng katta va eng kichik elementlari joylashgan satrlari o‘rinlarini almashtiring.

Qatorli massivlar va fayllar

№7-topshiriq

1–variant. Belgilar qatori berilgan. Eng uzun so‘zni teskari tartibda joylashtiring.

2–variant. Matndagi S1-qatordagi “Burgut” so‘zi bilan “Oq qush” o‘rinlari almashtirilsin.

3–variant. Matnli faylni o‘qib, undan uchta gapni ajratib, uni teskari tartibda chiqarib beruvchi dastur tuzilsin.

4–variant. Matnli faylni o‘qib, undan faqat klaviatura orqali kiritilgan gaplarni ekranga chiqarib beruvchi dastur tuzilsin.

5–variant. Matnli faylni o‘qib, undan faqat ikki xonali sonlar qatnashgan qatorlarni ekranga chiqarib beruvchi dastur tuzilsin.

6–variant. Fayldan lotin harflariga oid bo‘lgan matnlarni o‘qib, undan unli harflar bilan boshlanadigan so‘zlarni ekranga chiqarib beruvchi dastur tuzilsin.

7–variant. Fayldan matnni o‘qib, undan har ikkita qo‘shni so‘zlarni joyini almashtirib beruvchi dastur kodini yozing.

8–variant. Matnli faylni o‘qib, undan faqat vergul ishtirok etmagan gaplarni chiqarib beruvchi dastur tuzilsin.

9–variant. Fayldan matnni o‘qib, undan faqat to‘rtta harfdan oshmagan so‘zlarning qanchaligini aniqlab beruvchi dastur kodini yozing.

10–variant. Fayldan matnni o‘qib, undan faqatssitatalarni, ya’ni qo‘shtirnoq ichidagi yozuvlarni ekranga chiqarib beruvchi dastur tuzilsin.

11–variant. Fayldan matnni o‘qib, undan faqat berilgan so‘zlar miqdoridan iborat bo‘lgan gaplarni ekranga chiqarib beruvchi dastur tuzilsin.

12–variant. Fayldan lotin harflariga oid bo‘lgan matnlarni o‘qib, undan unli harflar bilan boshlanadigan va shu harflar bilan yakunlanadigan so‘zlarni ekranga chiqarib beruvchi dastur tuzilsin.

13–variant. Fayldan matnni o‘qib, undan faqat ikki xonali sonlar qatnashmagan qatorlarni ekranga chiqarib beruvchi dastur tuzilsin.

14–variant. Fayldan matnni o‘qib, undan faqat tire bilan boshlanuvchi yoki probel simvollari bilan boshlanuvchi gaplarni ekranga chiqarib beruvchi dastur tuzilsin.

15–variant. Fayldan lotin harflariga oid bo‘lgan matnlarni o‘qib, matnda uchragan har bir so‘zning birinchi harfi unli harflar bilan boshlanganlarini o‘rniga uning yozuv shakliga almashtirilib, uni ekranga chiqarib beruvchi dastur yozilsin.

16–variant. Matnli faylni o‘qib, unda uchragan 0 dan 9 gacha bo‘lgan sonlarning o‘rniga mos ravishda “nol”, “bir”, ... “to‘qqiz” so‘zlari bilan almashtirib, har bir yangi gapni yangi qatordan boshlovchi dastur tuzing.

17–variant. Matnli faylni o‘qib, unda qatnashgan eng uzun so‘zni aniqlab, uni necha marta qatnashganligini aniqlab beruvchi dastur kodini yozing.

18–variant. Matnli faylni o‘qib, avval unda qatnashgan so‘roqli gaplarni, so‘ngra esa undovli gaplarni ekranga chiqarib beruvchi dastur tuzilsin.

19–variant. Matnli faylni o‘qib, har bir gapdan so‘ng shu gapning ichida klaviaturadan kiritilgan so‘z necha marta qatnashganligini hisoblab, uni ekranga chiqarib beruvchi dastur kodi yozilsin.

20–variant. Matnli faylni o‘qib, uning hamma gaplarini teskari tartibda joylashtirib, uni ekranga chiqarib beruvchi dastur tuzilsin.

21–variant. Matnli faylni o‘qib, undan avval bir harfdan tashkil topgan so‘zlarni, so‘ngra qolgan barcha so‘zlarni ekranga chiqarib beruvchi dastur tuzilsin.

22–variant. Matnli faylni o‘qib, undan maksimal punktuatsiyalarga ega bo‘lgan gapni ekranga chiqarib beruvchi dastur tuzilsin.

Yozuv va jadvallar

1- misol. Quyidagi maydonlardan tashkil topgan, STUDENT nomi bilan tuzilmani hosil qiling:

- Familiya;
- Initsiallari;
- Fakulteti;
- Guruh raqami;
- O‘zlashtirish darajasi;
- Manzili.

Navbatdagi amallarni bajaruvchi dasturni tuzing:

- 10 ta STUDENT toyifasidagi tuzilmani klaviatura orqali massiv ma’lumotlarini kiriting;
- Talaba haqidagi ma’lumotlarni kiriting;
- Agar bunday talabalar bo‘lmasa, u holda ekranga mos xabar matni chiqarilsin.

Misolni echish:

```
#include <iostream>
#include<conio.h>
 using namespace std;

struct talaba
  {
   char fam[15];
   char ismi[15];
   char fakultet[10];
   float guruhi;
   char manzili[20];
  }uquvchi[100];
```

```
main()
{
 int n;
 cout<<"Talabalar sonini kiriting = ";
 cin>>n;
 for (int i=0; i<n; i++)
   {
    cout<<"Talaba familiyasini kiriting = ";
    cin>>uquvchi[i].fam;
     cout<<"Talaba ismini kiriting = ";
     cin>>uquvchi[i].ismi;
       cout<<"Fakultet nomini kiriting = ";
       cin>>uquvchi[i].fakultet;
         cout<<"Guruh raqamini kiriting = ";
         cin>>uquvchi[i].guruhi;
       cout<<"Talaba manzilini kiriting = ";
       cin>>uquvchi[i].manzili;
       cout<<"\n ---------------------------------- \n";
   }
     getch();
     return 0;
}
```

Misol:

```
#include<iostream>
#include<conio.h>
using namespace std;
int main()
```

```
{
struct employee{
        char name[64];
        long employee_id;
        float salary;
        char phone[10];
        int office_number ;
        } worker;
                strcpy(worker.name, "Tolibov Bekzod");
        worker.employee_id = 12345;
        worker.salary = 250000.00;
        worker.office_number = 102;
                 strcpy(worker.phone, "7065478");
cout<<"Xizmatchi = "<< worker.name << endl;
cout<<"Telefon = "<< worker.phone << endl;
cout<<"Maosh = "<< worker.salary << endl;
cout<<"Office = "<< worker.office_number << endl;
getch();
return 0; }
```

3-misol. Struktura va funksiya.

```
#include<iostream>
#include<conio.h>
using namespace std;
struct employee{
        char name[64];
        long employee_id;
        float salary;
```

```
        char phone[10];
        int office_number ;
        };
void show_employee(employee worker)
        {
         cout<<"Xizmatchi = "<< worker.name << endl;
         cout<<"Telefon = "<< worker.phone << endl;
         cout<<"Maosh = "<< worker.salary << endl;
         cout<<"Office = "<< worker.office_number << endl;
        }
int main()
{
        employee worker;
        strcpy(worker.name, "Tolibov Bekzod");
        worker.employee_id = 12345;
        worker.salary = 250000.00;
        worker.office_number = 102;
        strcpy(worker.phone, "7065478");
        show_employee(worker);
  getch();
return 0;
}
```

4-misol. N-ta ishchi xaqidagi ma’lumotni kiritish:

```
#include<iostream>
#include<conio.h>
#include<string.h>
```

```
using namespace std;
struct employee{
            char name[64];
            long employee_id;
            float salary;
            char phone[10];
            int office_number ;
            };
int show_employee(employee worker)
            {
             cout<<"Xizmatchi = "<< worker.name << endl;
             cout<<"Xizmatchi = "<< worker.employee_id << endl;
             cout<<"Telefon = "<< worker.phone << endl;
             cout<<"Maosh = "<< worker.salary << endl;
             cout<<"Office = "<< worker.office_number << endl;
             //return show_employee;
            }
int main()
{ employee worker;
   int n;
       cout<<"\n Xizmatchi sonini kiriting = ";
       cin>>n;
          for(int i=0; i<n; ++i){
            cout<<"\n Xizmatchi FIO kiriting = ";
            cin>>worker.name;
            cout<<"\n Xizmatchi raqamini kiriting = ";
            cin>>worker.employee_id;
            cout<<"\n Xizmatchi maoshini kiriting = ";
```

```
        cin>>worker.salary;
        cout<<"\n Xizmatchi office raqamini kiriting = ";
        cin>>worker.office_number;
        cout<<"\n Xizmatchi tel raqamini kiriting = ";
        cin>>worker.phone;
        cout<<"\n ---------------------- \n";}
        cout<<"\n ======================== \n";
        show_employee(worker);
getch();
return 0;
}
```

5-misol. Ko‘rsatkichlardan foydalanib funksiyani ishlab chiqish:

```
#include<iostream>
#include<conio.h>
#include<string.h>
using namespace std;
struct employee{
        char name[64];
        long employee_id;
        float salary;
        char phone[10];
        int office_number ; };
void name(employee *worker)
        {       cout<<"\n Xizmatchi Ismini keriting = ";
                cin>> worker->name;     }
void phone(employee *worker)
```

```
        {       cout<<"\n Telefon = ";
                cin>> worker->phone;        }
void salary(employee *worker)
        {       cout<<"\n Maosh = ";
                cin>> worker->salary;       }
void office_number(employee * worker)
        {       cout<<"\n Office raqami = ";
                cin>> worker->office_number;    }
void get_employee_id(employee *worker)
        {       cout<<"\n Xizmatchi raqamini kiriting = ";
                cin>> worker -> employee_id;    }
int main()
{ employee worker;
   int n;
      cout<<"\n Xizmatchi sonini kiriting = ";
      cin>>n;
         for(int i=0; i<n; ++i)
      {    name(&worker);
           get_employee_id(&worker);
           salary(&worker);
           office_number(&worker);
           phone(&worker);
           cout<<"\n ---------------------- \n";}
           cout<<"\n ======================== \n";
getch();
return 0;
}
```

№ 8-topshiriq

1–variant. Quyidagi maydonlardan tashkil topgan, STUDENT nomi bilan tuzilmani hosil qiling:

- Familiya va initsiallari;
- Guruh raqami;
- O‘zlashtirish darajasi (massiv 5ta elementdan iborat).

Navbatdagi amallarni bajaruvchi dasturni tuzing:

- 10 ta STUDENT toyifasidagi tuzilmani klaviatura orqali massiv ma’lumotlarini kiriting;
- Ro‘yxatda guruh raqami o‘sish tartibida bo‘lishi shart;
- O‘rtacha bali 4.0 dan yuqori bo‘lgan talabalar ro‘yxati ekranga chiqarilsin.
- Agar bunday o‘zlashtirishga ega talabalar bo‘lmasa, u holda ekranga mos xabar matnini chiqarilsin.

2–variant. Quyidagi maydonlardan tashkil topgan, STUDENT nomi bilan tuzilmani hosil qiling:

- Familiya va initsiallari;
- Guruh raqami;
- O‘zlashtirish darajasi (massiv 5ta elementdan iborat).

Navbatdagi amallarni bajaruvchi dasturni tuzing:

- 10 ta STUDENT toyifasidagi tuzilmani klaviatura orqali massiv ma’lumotlarini kiriting; Ro‘yxat talabalarni o‘zlashtirish baliga nisbatan o‘sish tartibida joylashtirilsin;
- 4 va 5 bahoga o‘zlashtirgan talabalar ro‘yxati ekranga chiqarilsin;
- Agar bunday o‘zlashtirishga ega talabalar bo‘lmasa, u holda ekranga mos xabar matni chiqarilsin.

3–variant. Quyidagi maydonlardan tashkil topgan, STUDENT nomi bilan tuzilmani hosil qiling:

- Familiya va initsiallari;
- Guruh raqami;
- O‘zlashtirish darajasi (massiv 5ta elementdan iborat).

Navbatdagi amallarni bajaruvchi dasturni tuzing:

- 10 ta STUDENT toyifasidagi tuzilmani klaviatura orqali massiv ma’lumotlarini kiriting; Ro‘yxat talabalarni alfavit bo‘yicha tartiblansin;
- 2 bahosi bor barcha talabalarni familiyasi va guruh raqami bo‘yicha ro‘yxatini ekranga chiqarilsin;
- Agar bunday o‘zlashtirishga ega talabalar bo‘lmasa, u holda ekranga mos xabar matnini chiqarilsin.

4–variant. Quyidagi maydonlardan tashkil topgan, AEROFLOT nomi bilan tuzilmani shakllantiring:

- Belgilangan reys punkti nomini;
- Reys raqami;
- Samolyot turi.

Navbatdagi amallarni bajaruvchi dasturni tuzing:

- 7 ta AEROFLOT toyifadan tashkil topgan tuzilmani klaviatura orqali massiv ma’lumotlarini kiriting; Ro‘yxat reys raqami asosida o‘sish tartibida bo‘lsin;
- Reys raqami, samolyot turi, belgilangan uchish yo‘li, berilgan nom bilan mos holda ekranga chiqarilsin;
- Agar bunday reys mavjud bo‘lmasa, u holda ekranga mos xabar matni chiqarilsin.

5–variant. Quyidagi maydonlardan tashkil topgan, AEROFLOT nomi bilan tuzilmani shaklantiring:

- Belgilangan reys punkti nomini;
- Reys raqami;
- Samolyot turi.

Navbatdagi amallarni bajaruvchi dasturni tuzing:

- 7 ta AEROFLOT toyifadan tashkil topgan tuzilmani klaviatura orqali massiv ma’lumotlarini kiriting; ro‘yxat belgilangan punkt nomi asosida alfavit tartibida bo‘lsin;
- Punkit nomi, reys raqami, xizmat ko‘rsatuvchi samolyot turi bilan mos holda ekranga chiqarilsin;
- Agar bunday reys mavjud bo‘lmasa, u holda ekranga mos xabar matni chiqarilsin.

6–variant. Quyidagi maydonlardan tashkil topgan, WORKER nomi bilan tuzilmani shaklantirilsin:

- Ishchining familiyasi va initsiali;
- Lavozimi;
- Ishga kirgan vaqti yoki yili.

Navbatdagi amallarni bajaruvchi dasturni tuzing:

- 10 ta WORKER toyifadan tashkil topgan tuzilmani klaviatura orqali massiv ma’lumotlarini kiriting; ro‘yxat alfavit asosida tartiblansin;
- Ishchi familiyasi, uning ish stajini dastur klaviatura orqali kiritilgan ma’lumotlar asosida hisoblab, ekranga chiqarsin;
- Agar bunday ishchi bo‘lmasa, u holda ekranga mos xabar matni chiqarilsin.

7–variant. Quyidagi maydonlardan tashkil topgan, TRAIN nomi bilan tuzilmani shaklantirilsin:

- Belgilangan punkt nomi;
- Poezd raqami;
- Jo‘nash vaqti.

Navbatdagi amallarni bajaruvchi dasturni tuzing:

- 7 ta TRAIN toyifadan tashkil topgan tuzilmani klaviatura orqali massiv ma’lumotlarini kiriting; belgilangan punkt ro‘yxat alfavit asosida tartiblansin;

- Klaviatura orqali kiritilgan vaqt asosida poezdlar haqidagi ma’lumotni ekranga chiqarsin;
- Agar bunday poezd bo‘lmasa, u holda ekranga mos xabar matni chiqarilsin.

8–variant. Quyidagi maydonlardan tashkil topgan, TRAIN nomi bilan tuzilmani shaklantirilsin:

- Belgilangan punkt nomi;
- Poezd raqami;
- Jo‘nash vaqti.

Navbatdagi amallarni bajaruvchi dasturni tuzing:

- 6 ta TRAIN toyifadan tashkil topgan tuzilmani klaviatura orqali massiv ma’lumotlarini kiriting; Belgilangan jo‘nash vaqtlari bo‘yicha poezdlar ro‘yxati tartiblansin;
- Klaviatura orqali kiritilgan yo‘nalish punktlari asosida poezdlar haqidagi ma’lumot ekranga chiqarsin;
- Agar bunday yo‘nalishda poezd bo‘lmasa, u holda ekranga mos xabar matni chiqarilsin.

9–variant. Quyidagi maydonlardan tashkil topgan, TRAIN nomi bilan tuzilmani shaklantirilsin:

- Belgilangan punkt nomi;
- Poezd raqami;
- Jo‘nash vaqti.

Navbatdagi amallarni bajaruvchi dasturni tuzing:

- 8 ta TRAIN toyifadan tashkil topgan tuzilmani klaviatura orqali massiv ma’lumotlarini kiriting; poezd raqamlari bo‘yicha ro‘yxat tartiblansin;
- Klaviatura orqali kiritilgan poezd raqamlari haqidagi ma’lumot ekranga chiqarsin;

- Agar bunday raqamdagi poezd bo‘lmasa, u holda ekranga mos xabar matni chiqarilsin.

10–variant. Quyidagi maydonlardan tashkil topgan, MARSH nomi bilan tuzilmani shaklantirilsin:

- Yo‘nalishni boshlanish punktining nomi;
- Yo‘nalishni yakunlanish punktining nomi;
- Yo‘nalish raqami.

Navbatdagi amallarni bajaruvchi dasturni tuzing:

- 5 ta MARSH toyifadan tashkil topgan tuzilmani klaviatura orqali massiv ma’lumotlarini kiriting; yo‘nalishlar raqamlari bo‘yicha ro‘yxat tartiblansin;
- Klaviatura orqali kiritilgan yo‘nalishlar raqamlari haqidagi ma’lumot ekranga chiqarsin;
- Agar bunday yo‘nalish bo‘lmasa, u holda ekranga mos xabar matni chiqarilsin.

11–variant. Quyidagi maydonlardan tashkil topgan, MARSH nomi bilan tuzilmani shakllantirilsin:

- Yo‘nalishni boshlanish punktning nomi;
- Yo‘nalishni yakunlanish punktning nomi;
- Yo‘nalish raqami.

Navbatdagi amallarni bajaruvchi dasturni tuzing:

- 9 ta MARSH toyifadan tashkil topgan tuzilmani klaviatura orqali massiv ma’lumotlarini kiriting; yo‘nalishlar raqamlari bo‘yicha ro‘yxat tartiblansin;
- Klaviatura orqali kiritilgan boshlang‘ich yoki yakuniy punkt yo‘nalishlar bo‘yicha raqamlari haqidagi ma’lumot ekranga chiqarsin;
- Agar bunday yo‘nalish bo‘lmasa, u holda ekranga mos xabar matni chiqarilsin.

12–variant. Quyidagi maydonlardan tashkil topgan, NOTE nomi bilan tuzilmani shakllantirilsin:

- Familiya, ismi;

- Telefon raqami;
- Tug‘ilgan sanasi (massiv uchta raqamdan tashkil topgan).

Navbatdagi amallarni bajaruvchi dasturni tuzing:

- 10 ta NOTE toyifasidan tashkil topgan tuzilmani klaviatura orqali massiv ma’lumotlari kiritilsin; ro‘yxat abonentlarning tug‘ilgan sanasi asosida tartiblansin;
- Klaviatura orqali kiritilgan telefon raqamlari asosida abonentlar haqidagi ma’lumot ekranga chiqarsin;
- Agar bunday raqamdagi abonentlar bo‘lmasa, u holda ekranga mos xabar matni chiqarilsin.

13–variant. Quyidagi maydonlardan tashkil topgan, NOTE nomi bilan tuzilmani shakllantirilsin:

- Familiya, ismi;
- Telefon rakami;
- Tug‘ilgan sanasi (massiv uchta raqamdan tashkil topgan).

Navbatdagi amallarni bajaruvchi dasturni tuzing:

- 8 ta NOTE toyifasidan tashkil topgan tuzilmani klaviatura orqali massiv ma’lumotlari kiritilsin; ro‘yxat alfavit asosida tartiblansin;
- Klaviatura orqali kiritilgan abonentlarning tug‘ilgan oy sanasi bo‘yicha ma’lumot ekranga chiqarsin;
- Agar bunday abonent bo‘lmasa, u holda ekranga mos xabar matni chiqarilsin.

14–variant. Quyidagi maydonlardan tashkil topgan, NOTE nomi bilan tuzilmani shaklantirilsin:

- Familiya, ismi;
- Telefon raqami;
- Tug‘ilgan sanasi (massiv uchta raqamdan tashkil topgan).

Navbatdagi amallarni bajaruvchi dasturni tuzing:

- 9 ta NOTE toyifasidan tashkil topgan tuzilmani klaviatura orqali massiv ma’lumotlari kiritilsin; ro‘yxat abonentlarning telefon raqamlarini dastlabki uchtasi asosida tartiblansin;
- Klaviatura orqali kiritilgan abonent haqidagi ma’lumotlarni ekranga chiqarsin;
- Agar bunday abonent bo‘lmasa, u holda ekranga mos xabar matni chiqarilsin.

15–variant. Quyidagi maydonlardan tashkil topgan, BELGI nomi bilan tuzilmani shakllantirilsin:

- Familiya, ismi;
- Burj belgisi;
- Tug‘ilgan sanasi (massiv uchta raqamdan tashkil topgan).

Navbatdagi amallarni bajaruvchi dasturni tuzing:

- 8 ta BELGI toyifasidan tashkil topgan tuzilmani klaviatura orqali massiv ma’lumotlari kiritilsin; ro‘yxat tug‘ilgan sana asosida tartiblansin;
- Klaviatura orqali kiritilgan familiya egasi haqidagi ma’lumot ekranga chiqarsin;
- Agar bunday familiya bo‘lmasa, u holda ekranga mos xabar matni chiqarilsin.

16–variant. Quyidagi maydonlardan tashkil topgan, BELGI nomi bilan tuzilmani shakllantirilsin:

- Familiya, ismi;
- Burj belgisi;
- Tug‘ilgan sanasi (massiv uchta raqamdan tashkil topgan).

Navbatdagi amallarni bajaruvchi dasturni tuzing:

- 8 ta BELGI toyifasidan tashkil topgan tuzilmani klaviatura orqali massiv ma’lumotlari kiritilsin; ro‘yxat tug‘ilgan sana asosida tartiblansin;
- Klaviatura orqali kiritilgan familiya egasining tug‘ilgan burji haqidagi ma’lumot ekranga chiqarsin;
- Agar bunday familiya bo‘lmasa, u holda ekranga mos xabar matni chiqarilsin.

17–variant. Quyidagi maydonlardan tashkil topgan, BELGI nomi bilan tuzilmani shakllantirilsin:

- Familiya, ismi;
- Burj belgisi;
- Tug‘ilgan sanasi (massiv uchta raqamdan tashkil topgan).

Navbatdagi amallarni bajaruvchi dasturni tuzing:

- 7 ta BELGI toyifasidan tashkil topgan tuzilmani klaviatura orqali massiv ma’lumotlari kiritilsin; ro‘yxat Burj belgisi asosida tartiblansin;
- Klaviatura orqali kiritilgan familiya egasi haqidagi ma’lumot tug‘ilgan oyi bo‘yicha ekranga chiqarsin;
- Agar bunday familiya bo‘lmasa, u holda ekranga mos xabar matni chiqarilsin.

18–variant. Quyidagi maydonlardan tashkil topgan, PRICE nomi bilan tuzilmani shakllantirilsin:

- Molning nomi;
- Bunday turdagi mollar sotuvchi do‘kon nomi;
- So‘mdagi molning narxi.

Navbatdagi amallarni bajaruvchi dasturni tuzing:

- 8 ta PRICE toyifasidan tashkil topgan tuzilmani klaviatura orqali massiv ma’lumotlari kiritilsin; ro‘yxatdagi mollar nomi bo‘yicha alfavit asosida tartiblansin;
- Klaviatura orqali nomi kiritilgan mol haqidagi to‘liq ma’lumot ekranga chiqarsin;
- Agar bunday nomdagi mol bo‘lmasa, u holda ekranga mos xabar matni chiqarilsin.

19–variant. Quyidagi maydonlardan tashkil topgan, PRICE nomi bilan tuzilmani shakllantirilsin:

- Molning nomi;

- Bunday turdagi mollar sotuvchi do‘kon nomi;
- So‘mdagi molning narxi.

Navbatdagi amallarni bajaruvchi dasturni tuzing:

- 10 ta PRICE toyifasidan tashkil topgan tuzilmani klaviatura orqali massiv ma’lumotlari kiritilsin; ro‘yxat do‘konlar nomi bo‘yicha alfavit asosida tartiblansin;
- Klaviatura orqali nomi kiritilgan mol sotiladigan do‘konlar haqidagi ma’lumot ekranga chiqarsin;
- Agar bunday nomdagi mol sotiladigan do‘kon bo‘lmasa, u holda ekranga mos xabar matni chiqarilsin.

20–variant. Quyidagi maydonlardan tashkil topgan, ORDER nomi bilan tuzilmani shaklantirilsin:

- To‘lovning hisob raqami;
- Mijozning to‘lov raqami;
- So‘mda o‘tkazilgan pul miqdori.

Navbatdagi amallarni bajaruvchi dasturni tuzing:

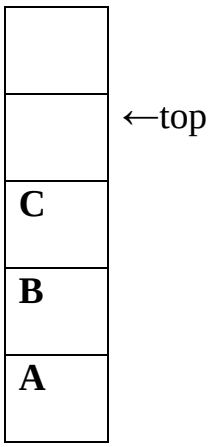
- 10 ta ORDER toyifasidan tashkil topgan tuzilmani klaviatura orqali massiv ma’lumotlari kiritilsin; ro‘yxat to‘lovchilarning familiyasi bo‘yicha alfavit asosida tartiblansin;
- Klaviatura orqali kiritilgan pul miqdorini to‘lagan mijozlar haqidagi ma’lumot ekranga chiqarsin;
- Agar bunday miqdorda pul to‘lagan mijoz bo‘lmasa, u holda ekranga mos xabar matni chiqarilsin.

2.2. 5-amaliy mashg‘ulot. YArim statik ma’lumotlar tuzilmasi steklar, navbatlar, deklar ustida amallar bajarish

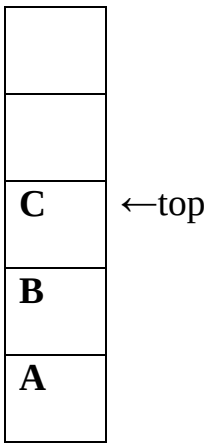
Ma’lumotlar tuzilmasi guruhlashgan ma’lumotlarni to‘plamini o‘zida ask etiradi. Bunga sodda misol sifatida massivlarni keltirish mumkin bo‘ladi. Massivlarni barcha elementlari bir xil toyifada bo‘ladi. Ayrim hollarda shunday masalalarga duch kelinadiki, ularda toyifalar u yoki bu tartibda bo‘ladi. Bunday ma’lumotlarga steklar, navbatlar, deklar misol bo‘ladi.

Stek deb shunday ma’lumotlar tuzilmasiga aytiladiki, bunda eng oxiri kelgan element birinchi o‘chiriladi. Stek odatda vertikal massiv sifatida oxirgi elementini ko‘rsatuvchi ko‘rsatkichi bilan tasvirlanadi va bu ko‘rsatkich (top) stek uchi deb yuritiladi. YAngi element stek uchiga joylashtiriladi va u birinchi bo‘lib o‘chiriladi. Stek ko‘pincha LIFO deb yuritiladi (Last in, First out – birinchi kelgan oxiri chiqadi). Agar stekda bitta element bo‘lib, u ham o‘chirilsa, ushbu stekga boshka element joylashtirib bo‘lmaydi. Bunday stek bo‘sh stek deb yuritiladi.

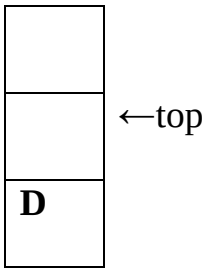
Masalan, stekda A, V va S elementlarga ega bo‘lib, bulardan A birinchi element, S oxirgi element bo‘lsa, u grafik shaklda quyidagicha ifodalanadi:

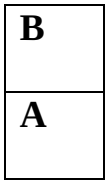


a-rasm. Stekni tashkil etuvchi elementlar



b-rasm. Stekdan o‘chirilgan elment





v-rasm. Stekga qo‘shilgan element

Stek ustida element kiritish, stek bo‘shligini tekshirish va stekdan elementni o‘chirish amallari bajariladi hamda quyidagi funksiyalar yordamida amalga oshiriladi:

- stekga element kritish funksiyasi:
  void Push(LIFO, top, number)
- navbat bo‘shligini tekshirish funksiyasi:
  void Empty(top)
- navbatdan elementni o‘chirish yoki chiqarish funksiyasi:
  void Pop(top)

Navbatlar

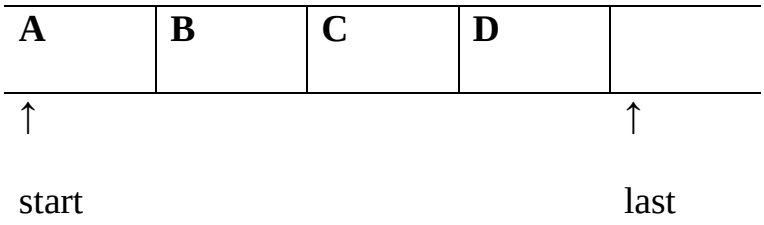
Navbat deb shunday ma’lumotlar tuzilmasiga aytiladiki, bunda belgilangan oraliqqa elementlarni kiritish va xizmat ko‘rsatilgan elementni chiqarish yoki o‘chirish bajariladi. Agar navbatda hech qanday element bo‘lmasa, navbat bo‘sh deyiladi. Bunday holda Start va last ko‘rsatkichlar navbatni faqat bir joyni ko‘rsatadi.

Navbatni amalga oshirishning turli usullari mavjud. Ulardan eng soddasi quyidagicha: navbat amalga oshirilayotgan massivda ikkita ko‘rsatkich bo‘ladi. Birinchisi navbat boshini ko‘rsatsa, ikkinchisi esa mos holda navbat oxirini ko‘rsatadi.

Navbat boshi – bu massivdan o‘chiriladigan elementning raqamidir.

Navbat oxiri – bu navbatga kiritiladigan massiv elementidir.

Navbatni ishlash tamoyili birinchi kelgan elementga birinchi bo‘lib xizmat ko‘rsatiladi: FIFO (First In, First Out-birinchi kelgan birinchi ketadi).



g-rasm. Navbatni birinchi elementi A va oxirgi elementi D.

Navbat ustida element kiritish, navbat bo‘shligini tekshirish va navbatdan elementni o‘chirish amallari bajariladi hamda quyidagi funksiyalar yordamida amalga oshirish mumkin:

- navbatga element kiritish funksiyasi:
  void Add(int last, int number)
- navbat bo‘shligini tekshirish funksiyasi:
  void Empty(int start, int last)
- navbatdan elementni o‘chirish yoki chiqarish funksiyasi:
  void Remove(int strat, int last)

Stekga misol:

1-misol. Agar stekka kiruvchi son yonma-yon turgan elementlari teng bo‘lsa ularni o‘chirsin va "yes" yozuvi, aks holda "no" yozuvini ekranga chiqsin.

Stek elementlari: 2211558866

```
#include<iostream>
#include<conio.h>
using namespace std;
char lifo[100],ch[100];int i,top;
void push(char number);
int main(){
  cin >>ch;
  top=0;
  for(i=0;i<strlen(ch);i++){
    push(ch[i]);
    if(lifo[top-1]==lifo[top-2]&&top>1)
    top-=2;
    }
    if(top==0)cout <<"yes";else cout<<"no";
    getch();
  return 0;
  }
  void push(char number){
    lifo[top]=number;
    top++;
    }
```

№9 - topshiriq

1–variant. Balandligi h=30 sm bo‘lgan qutiga, o‘lchami h=3 sm bo‘lgan yog‘och taxtadan (yog‘och taxta qutini yuzasi bilan bir xil o‘lchamda) nechta kiritilganda quti to‘lishini ko‘rsatuvchi dasturni tuzing.

2–variant. 10 elementli bir o‘lchamli massiv berilgan. Ushbu massiv stek xususiyatiga ega bo‘lib, unga 11-element kiritilganda stek to‘lib ketganini ma’lum qiluvchi dasturni tuzing.

3–variant. Quyida keltirilgan vazifani bajaruvchi dasturni tuzing. Stek o‘lchami 5 ga teng. Unga 3 ta element kiritilganda top ko‘rsatkich qiymati nechaga teng bo‘ladi.

4–variant. Stekda 7 ta element mavjud. Mazkur stekni elementlardan bo‘shatish dasturini tuzing.

5–variant. Stekda 10 ta element mavjud. Elementlarni bitta-bitta tanlovchi dasturni tuzing.

6–variant. Avtomobilni maksimum tezligi 220 km/s. Mazkur avtomobil tezligi 60 km/s, 80 km/s, 100 km/s tezliklardan ortib borganda, mazkur tezlik qanday axoli punktlarida va qanday yo‘llar uchun mo‘ljallanganligi haqidagi xabarni ekranga chiqaruvchi dasturni tuzing. Bunda har bir tezlik ko‘rsatkichi stekga element sifatida kiritilsin.

7–variant. Qo‘shish (+) arifmetik amali asosida stekka element kiritish dasturini tuzing. Elementlarini kiritish jarayoni quyidagicha bo‘lsin: birinchi va ikkinchi elementlar stekka kiritilgandan so‘ng, uchinchi elementni dastur birinchi va ikkinchi elementlar yig‘indisidan hosil bo‘lsin. To‘rtinchi element esa ikkinchi va uchinchi elementlar yig‘indisidan hosil bo‘lsin. SHu tariqa stekka 7 ta element kiritish dasturini tuzing.

Stek o‘lchami	Kiritilayotgan raqam va amal	Stek tarkibidagi elementlar
1	3	3
2	1	3, 1
3	+	3, 1, 4,
4	+	3, 1, 4, 5
5	+	3, 1, 4, 5, 9
6	+	3, 1, 4, 5, 9, 14
7	+	3, 1, 4, 5, 9, 14, 23

8–variant. Ayrish (-) arifmetik amali yordamida stekka element kiritish dasturini tuzing. YUqoridagi variantdagi kabi birinchi va ikkinchi elementlarni foydalanuvchi tomonidan kiritiladi. Qolgan elementni esa dastur o‘zi qo‘shni elementlarni juftligi ayirmasidan hosil bo‘lgan qiymatlarni kiritib borish natijasida

stekni to‘ldirsin. Bunda ham stek elementlari 7 ta bo‘lguncha davom etsin. Bu jarayon quyidagi jadvalda keltirilgan:

Stek o‘lchami	Kiritilayotgan raqam va amal	Stek tarkibidagi elementlar
1	3	3
2	1	3, 1
3	-	3, 1, 2,
4	-	3, 1, 2, 1
5	-	3, 1, 2, 1, 1
6	-	3, 1, 2, 1, 1, 0
7	-	3, 1, 2, 1, 1, 0, 1

9–variant. Algebraik ifodadagi raqamlar ustida bajarilgan arifmetik amallar va raqamlar o‘rtasidagi mos munosabatlarni o‘rnatish orqali (har bir amal natijasida hosil bo‘lgan qiymat stekni bitta elementi bo‘lsin) stekka element kiritish dasturini tuzing. Bu jarayon quyidagi jadvalda keltirilgan:

Kiritilayotgan raqam yoki belgi	Stek tarkibidagi elementlar
4	4
6	4, 6
3	4, 6, 3
-	4, 3
*	12
8	12, 8
6	12, 8, 6
-	12, 2

2	12, 2, 2
/	12, 1
+	13

a) (1+2)-3*(4+6)+(7-5)+45/9. bunda har bir amal bajarish natijasida hosil bo‘lgan son stekka bitta element bo‘lib kiradi.

b) a=2, b= 3, c=4, d=5 raqamlar berilgan quyidagi ifodani hisoblash jarayonidagi har bir amal bajarilganda hosil bo‘lgan qiymatni stekka bitta element qilib kiritish dasturini tuzing:

-a*b +c – d/a +c*d.

10–variant. N ta akrobat navbatda ko‘rgazmali chiqishida minora hosil qilishdi. Ushbu minorani har bir bosqichidagi akrobatlarni sonli miqdori stekka bittadan element bo‘lib kirsin. Birinchi bosqich stekni birinchi elementi, ikkinchi bosqich ikkinchi va h.k., bo‘lsin.

11–variant. Elementlar kiritilib, navbat shakllantirilsin. 15 ta elementdan iborat navbatni tashkil qiluvchi dasturni tuzing.

12–variant. Elementlar kiritilib, navbat shakllantirilsin. Navbatni birinchi va oxirgi elementlari o‘rni almashtiruvchi dasturni tuzing.

13–variant. 6 ta elementga mo‘ljallangan navbatga elementlar kiritish dasturini tuzing.

14–variant. Navbatni o‘rtasidani element o‘chirilsin. Agar navbat elementlari soni toq bo‘lsa bitta element, aks holda ikkita element navbatdan o‘chirilsin.

15–variant. Elementlar kiritilib, navbat shakllantirilsin. Navbatni juft o‘rinda turgan elementlarini o‘chiruvchi dastur tuzilsin.

16–variant. Elementlar kiritilib, navbat shakllantirilsin. Navbat o‘rtasiga '+' belgini joylashtiruvchi dastur tuzilsin.

17–variant. Elementlar kiritilib, navbat shakllantirilsin. Eng kichik element topilib va undan keyin “0”ni joylashtiruvchi dastur tuzilsin.

18–variant. 7 ta elementga mo‘ljallangan navbat berilgan. Elementlar kiritilib, navbat shakllantirilsin. Navbatning eng katta element topilib, undan keyin “***” belgini joylashtirilsin hamda navbatdagi o‘rnatilgan o‘lcham saqlanishini ta’minlovchi dastur tuzilsin.

19–variant. 10 ta elementga mo‘ljallangan navbat berilgan. Elementlar kiritilib, navbat to‘ldirilsin va undagi eng kichik elementni topib o‘chiruvchi dastur tuzilsin.

20–variant. Elementlar kiritilib, navbat shakllantirilsin. Ushbu navbatni o‘lchami 9 ga teng. Navbatni birinchi elementga teng barcha elementlar o‘chirilsin.

21–variant. Elementlar kiritilib, navbat shakllantirilsin. Navbatning oxirgi elementga teng barcha elementlarni o‘chiruvchi dastur tuzilsin.

22–variant. Elementlar kiritilib, navbat shakllantirilsin. Navbatning o‘lchami 9 ga teng va eng katta elementga teng barcha elementlarini o‘chiruvchi dastur tuzilsin.

2.3. 6-amaliy mashg‘ulot. Dinamik va chiziqsiz ma’lumotlar tuzilmasi

Bog‘langan ro‘yxatlar. Daraxtsimon ma’lumotlar tuzilmasi

1-misol. Ro‘yxat ustida quyidagi amallarni bajaruvchi protsedura yoki funksiyalarni o‘z ichiga olgan dasturni tuzing:

- ro‘yxatga element qo‘shish;
- ro‘yxat elementlarini ekranga chiqarish;
- ro‘yxatdan element qidirish;
- ro‘yxatdan element o‘chirish;

dasturi quyida keltirilgan:

```
#include <iostream.h>
#include <conio.h>
#include <alloc.h>
#include <stdlib.h>
struct TNode {
  int value;
  TNode* pnext;
  TNode(int val): pnext(0), value(val) {}
};
// Ro‘yxatga element qo‘shish
void add2list(TNode **pphead, int val) {
  TNode **pp = pphead, *pnew;
  while(*pp) {
  if(val < (*pp)->value)
          break;
  else
          pp = &((*pp)->pnext);
  }
  pnew = new TNode(val);
  pnew->pnext = *pp;
  *pp = pnew;
}
// Ro‘yxat elementlarini ekranga chiqarish
void print(TNode *phead) {
  TNode* p = phead;
  while(p) {
   cout <<""<< p->value<<"|" <<"--> ";
   p = p->pnext;
  }
  cout << endl;
}
// Ro‘yxatda elementni qidirish, C++
TNode* Find(TNode *phead, int x)
{
TNode *p=phead;
 while(p)
 if (p->value==x) return p;
          else p = p->pnext;
          return 0;
}

// Ro‘yxat elementini o‘chirish
void deleteList(TNode *phead) {
```

```
  if(phead) {
   deleteList(phead->pnext);
   if(phead)
         delete phead;
  }
 }

 // Asosiy qism
 int main() {int mas[1000], N, key; TNode* T;
  TNode *phead = 0;
  //srand(time(0));
  cout<<"Royhat uzunligini kirit"<<endl;
  cin>>N;
  cout<<"Elementlarni kirit!"<<endl;
  for(int j=0; j<N; j++)
  cin>>mas[j];
  for(int i = 0; i < N; ++i)
   add2list(&phead,mas[i]);
   cout<<"Qidiruv elementni kiriting!"<<endl;
   cin>>key;
   // rand() % 100);
  cout << "Elements of the list:" << endl;
  print(phead);
  T=Find(phead,key);
  if (T==0) cout <<"Bunday element yoq"<<endl;
  else cout <<"Bunday element bor"<<" "<<T->value<<" "<<T<<endl;
  deleteList(phead);
  getch();
 }
```

2-misol. Daraxt ustida amallar:

- Daraxtning ildizini hosil qilish;

- Daraxtga element qo‘shish;

- Daraxtdan element o‘chirish;

- Daraxtda element mavjudligini tekshirish;

```
#include<iostream.h>
```

```
#include<string.h>
#include<conio.h>
#pragma hdrstop
typedef int Tinfo;
typedef struct Item* Pitem;
struct Item {
   Tinfo Key;
   Pitem Left;
   Pitem Right;    };
   class TTree{
      private :
      Pitem Root;
      public :
      TTree();
      void Add(Tinfo);
      void Del(Tinfo);
      void View();
      void Exist(Tinfo Key);
      ~TTree();        };
//---------------------------------
TTree::TTree(){     Root=0;     };
   //---------------------------
   static void IniTree(Pitem& P,Tinfo X)
   {      P=new Item;
          P->Key=X;
          P->Left=0;
          P->Right=0;
      };
```

```
static void Iendleft(Pitem& P,Tinfo X)
{
  Pitem R;
  R=new Item;
  R->Key=X;
  R->Left=0;
  R->Right=0;
  P->Left=R;
    }
    static void InRight(Pitem P,Tinfo X)
    {
    Pitem R;
    R=new Item;
    R->Key=X;
    R->Left=0;
    R->Right=0;
    P->Right=P;
        }
   static void Tree_Add(Pitem P,Tinfo X)
   {
     int OK;
     OK=false;
     while(!OK){
         if(X>P->Key)
         if(P->Right!=0)
         P=P->Right;
         else {
             InRight (P,X);
```

```
      OK=true;
      }
      else if(P->Left!=0)P=P->Left;
      else{Iendleft(P,X);
      OK=true;}
  }
}
void TTree::Add(Tinfo Key)
{
 if(Root==0)
 IniTree(Root,Key);
 else Tree_Add(Root,Key);
    }
static void delete_(Pitem& P,Tinfo);

static void Del(Pitem& R,Pitem& Q)
{
 if(R->Right!=0)
 Del(R->Right,Q);
 else
 {
   Q->Key=R->Key;
   Q=R;
   R=R->Left;
     }
    }
static void delete_(Pitem& P,Tinfo X)
{
```

```
   Pitem Q;
   if(P!=0)
   if(X<P->Key)
   delete_(P->Left,X);
   else
   if(X>P->Key)
   delete_(P->Right,X);
   else{
      Q=P;
      if(Q->Right==0)
      P=Q->Left;
      else
      if(Q->Left==0)
      P=P->Right;
      else
      Del(Q->Left,Q);
      delete Q;
      }
     else
     cout<<"Bunday element daraxtda yuq..."<<endl;
}
    void TTree::Del(Tinfo Key)
    {
     delete_(Root,Key);
        }

    static void PrintTree(Pitem R,Tinfo L)
    {
```

```
int i;
if(R!=0){
  PrintTree(R->Right,L+3);
  for(i=1;i<=L;i++)
  cout<<"  ";
  cout<<R->Key<<endl;
  PrintTree(R->Left,L+3);
    }
  }
  void TTree::View()
  {
  PrintTree(Root,1);
     }
     //----------------------------
     static void Search(Pitem& P,Tinfo X )
     {
      if(P==0){
        cout<<"bunday element yuq"<<endl;
         }else
         if(X>P->Key)
         Search(P->Right,X);
         else if(X<P->Key)
         Search(P->Left,X );
         else
         cout<<"Bunday element bor..."<<endl;
         }
     void TTree::Exist(Tinfo Key)
     {
```

```
 Search(Root,Key);
  }
 static void Node_Dispose(Pitem P)
 {
  if(P!=0){
  if(P->Left!=0)
  Node_Dispose(P->Left);
  if(P->Right!=0)
  Node_Dispose(P->Right);
  delete P;
     }      }
    TTree::~TTree()
    {
      Node_Dispose(Root);
       }
//-------------------------------------------------------
          void inputKey(char S[128],Tinfo& Key)
           {  cout<<S<<endl;    cin>>Key; }
       TTree *Tree=new TTree;
       int N;    Tinfo Key;
       int main(int argc,const char* argv[] )
       {     do {
       cout<<"1-daraxtga element qoshish..."<<endl;
       cout<<"2-element uchirish..."<<endl;
       cout<<"3-daraxt tuguni..."<<endl;
       cout<<"4-tugunni tekshirish..."<<endl;
       cout<<"5-chiqish..."<<endl;
       cin>>N;
```

```
switch(N){case 1:{inputKey("qoshiluvchi element qiymatini kiriting...",Key); Tree->Add(Key); };
                case 2:{inputKey("uchiriladigan element qiymatini kiriting...",Key); Tree->Del(Key); } break;
                case 3:Tree->View();break;
                case 4:{
 inputKey("element kiriting,borligini siz bilishni istagan ",Key);Tree->Exist(Key);
                        } break; } } while(!(N==5)); getch(); return EXIT_SUCCESS; }
```

Natija:

```
Bunday element daraxtda yuq...
1-daraxtga element qoshish...
2-element uchirish...
3-daraxt tuguni...
4-tugunni tekshirish...
5-chiqish...
4
element kiriting,borligini siz bilishni istagan
7
Bunday element bor...
1-daraxtga element qoshish...
2-element uchirish...
3-daraxt tuguni...
4-tugunni tekshirish...
5-chiqish...
4
element kiriting,borligini siz bilishni istagan
6
bunday element yuq
1-daraxtga element qoshish...
2-element uchirish...
3-daraxt tuguni...
4-tugunni tekshirish...
5-chiqish...
```

№10-topshiriq

1–variant. Avtobus parkidagi avtobus haqidagi dinamik ma’lumotlardan tashkil topgan dastur tuzing.

Har bir avtobus haqidagi ma’lumotlar quyidagilardan iborat:

- avtobus nomeri (raqami);
- haydovchi ismi sharifi;
- yo‘nalish nomeri;

Dastur quyidagilarni ta’minlab berishi kerak:

- parkdagi barcha avtobuslar haqidagi dastlabki ma’lumotlarni ro‘yxat ko‘rinishida taqdim etish;
- har bir avtobus parkdan chiqayotganda uning nomeri kiritiladi va u haqidagi ma’lumotlar parkdagi avtobuslar haqidagi ma’lumotlar ro‘yxatidan chiqarilib, yo‘nalishda bo‘lgan avtobuslar haqidagi ma’lumotlar ro‘yxatiga qo‘shib qo‘yish;
- har bir avtobus yo‘nalishdan parkka kirganda uning nomeri kiritiladi va u haqidagi ma’lumotlar yo‘nalishda bo‘lgan avtobuslar haqidagi ma’lumotlar ro‘yxatidan parkdagi avtobuslar haqidagi ma’lumotlar ro‘yxatiga qo‘shib qo‘yish;
- so‘rov bo‘yicha parkda yoki yo‘nalishda bo‘lgan avtobus haqidagi ma’lumotlarni chiqarish.

2–variant. Kutubxonada mavjud kitobning ayni vaqtdagi holati haqidagi ma’lumotlar bilan ishlovchi dastur yarating.

Har bir kitob quyidagi ma’lumotlardan tashkil topadi:

- UDK nomeri;
- Muallif ismi sharifi;
- Nomi;
- Nashr etilgan sanasi;
- Kitobning kutubxonadagi miqdori;

Dastur quyidagilarni amalga oshirishi kerak:

- kutubxonadagi barcha kitoblar haqidagi ma’lumotlarni ikkilik daraxt ko‘rinishida tasvirlash;
- kutubxonaga kelgan yangi kitob haqidagi ma’lumotlarni ro‘yxatga qo‘shish;
- kutubxonadan chiqarib tashlangan kitob haqidagi ma’lumotlarni ro‘yxatdan chiqarib tashlash;

- so‘rov bo‘yicha kutubxonada mavjud kitoblar haqidagi ma’lumotlarni chiqarilgan yili bo‘yicha tartiblab, bosmaga chiqarish.

3–variant. Ayni vaqtdagi buyurtma berilgan aviabiletlar haqidagi ma’lumotlar bilan ishlovchi dastur tuzing.

Har bir buyurtma qilingan ma’lumotlar quyidagilardan tashkil topadi:

- belgilangan punkt (bo‘lim);
- reys nomeri;
- familiya ismi sharifi;
- uchish ko‘zlangan sana;

Dastur quyidagilarni amalga oshirishi kerak:

- barcha buyurtmalarni ro‘yxat tarzida saqlash;
- ro‘yxatga yangi buyurtma qo‘shish;
- buyurtmani o‘chirish;
- belgilangan reys nomeri va uchish vaqti bo‘yicha buyurtmalarni chiqarish;
- barcha buyurtmalarni chiqarish.

4–variant. Ayni vaqtdagi buyurtma berilgan aviyabiletlar haqidagi ma’lumotlar bilan ishlovchi dastur tuzing.

Har bir buyurtma qilingan ma’lumotlar quyidagilagdan tashkil topadi:

- belgilangan punkt (bo‘lim);
- reys nomeri;
- familiya ismi sharifi;
- uchish ko‘zlangan sana;

Dastur quyidagilarni amalga oshirishi kerak:

- barcha buyurtmalarni ikkilik daraxt tarzida saqlash;
- ro‘yxatga yangi buyurtma qo‘shish va o‘chirish;
- belgilangan reys nomeri va uchish vaqti bo‘yicha ketma-ket o‘chirilish tartibida bosmaga chiqarish;

- barcha buyurtmalarni chiqarish.

5–variant. Kutubxonada mavjud kitobning ayni vaqtdagi holati haqidagi ma’lumotlar bilan ishlovchi dastur yarating.

Har bir kitob quyidagi ma’lumotlardan tashkil topadi:

- UDK nomeri;
- Muallif ismi sharifi;
- Nomi;
- Nashr etilgan sanasi;
- Kitobning kutubxonadagi miqdori;

Dastur quyidagilarni amalga oshirishi kerak:

- kutubxonadagi barcha kitoblar haqidagi ma’lumotlarni ro‘yxat ko‘rinishida tasvirlash;
- kutubxonaga kelgan yangi kitob haqidagi ma’lumotlarni ro‘yxatga qo‘shish;
- kitob kitobxonga berilganda uning UDK nomeri kiritiladi, dastur kutubxonada mavjud kitoblar sonini bittaga kamaytiradi yoki so‘ralgan kitob kutubxonada mavjud emasligi yoki u qo‘lda ekanligi haqida ma’lumot chiqarish;
- kutubxonaga qaytgan kitobning UDK nomeri kiritiladi va dastur kitoblar ro‘yxatini bittaga orttirish;
- so‘rov bo‘yicha kutubxonada mavjud kitoblar haqidagi ma’lumotlarni chiqarish.

6–variant. Avtobus parkidagi avtobus haqidagi dinamik ma’lumotlardan tashkil topgan dastur tuzing.

Har bir avtobus haqidagi ma’lumotlar quyidagilardan iborat:

- avtobus nomeri (raqami);
- haydovchi ismi sharifi;
- yo‘nalish nomeri;
- avtobusni parkda yoki yo‘nalishda ekanligini ifodalovchi belgi.

Dastur quyidagilarni ta’minlab berishi kerak:

- parkdagi barcha avtobuslar haqidagi dastlabki ma’lumotlarni ro‘yxat ko‘rinishida taqdim etish;
- har bir avtobus parkdan chiqayotganda uning nomeri kiritiladi va dastur shu avtobus haqidagi ma’lumotlarga “avtobus yo‘nalishda” belgisi qo‘yiladi;
- har bir avtobus parkka kirganda uning nomeri kiritiladi va dastur shu avtobus haqidagi ma’lumotlarga “avtobus parkda” belgisi qo‘yiladi;
- so‘rov bo‘yicha avtobus parkda yoki yo‘nalishda ekanligi haqidagi ma’lumotlarni chiqarish.

7–variant. Libirentdan qidirish usuliga mo‘ljallangan dastur yaratish. Labirent kvadratlardan iborat bo‘lgan matritsa ko‘rinishda ifodalangan. Har bir kvadrat ochiq yoki yopiq bo‘lishi mumkin. YOpiq kvadratga kirish taqiqlangan. Agar kvadrat ochiq bo‘lsa unga kirish mumkin, faqat burchakdan emas. Har bir kvadrat matritsadagi koordinatasi bilan ifodalanadi.

Dastur labirent orqali belgilangan joydan kirish kerak. O‘tish yo‘li qidirilganidan so‘ng, topilgan o‘tish yo‘li kvadratlarning koordinatalari ko‘rinishida chiqariladi. Yo‘llarni saqlash uchun stekdan foydalaning.

8–variant. Avtomobil to‘xtash joyida bitta mashinalar turish chizig‘idan iborat va u yagona kirishga va yagona chiqishga egadir. Avtomashina egalari ixtiyoriy vaqtda va joydan avtomashinalarini olib kirish hamda chiqib ketish imkoniyati yo‘q. Endi avtomashina egasi o‘z mashinasini to‘xtash joyidan olib kirish yoki chiqib ketishi uchun belgilangan tartibda amalga oshiradi.

SHunday dastur tuzish kerakki, dastur avtomashinalar to‘xtash joyiga mashinani olib kirish va chiqish jarayonini modellashtirsin. Dasturni buyruq kiritish satriga ixtiyoriy avtomashina raqamini kiritilgandan uni to‘xtash joyiga qachon kirishi yoki qachon chiqib ketishi haqida ma’lumot bersin. Ushbu jarayon navbat shaklida tashkil qilinsin.

9–variant. Egiluvchan magnit diskka yozishni modellashtiruvchi dastur yarating. Diskdagi umumiy hajm 360 Kbayt. Fayllar 18 baytdan 32 Kbaytgacha

bo‘lgan oraliqdagi ixtiyoriy uzunlikka ega. Fayllar ishlatilish jarayonida diskka yozilishi yoki o‘chirilishi mumkin.

Dastlab fayllar bir tekisda, ketma-ket yoziladi. Diskdan fayl o‘chirilgach, o‘rnida bo‘shliq hosil bo‘ladi va yangi yozilayotgan fayl shu bo‘shliqqa sig‘sa, u shu erga yoziladi. Aks holda yangi fayl diskdagi eng oxirida yozilgan fayldan keyingi o‘ringa yoziladi.

Agar fayl hajmi eng katta bo‘sh joydan ortib ketsa, halokat xabari beriladi. Faylni yozish yoki o‘chirish jarayonida buyruqlar satrida fayl nomi, uning uzunligi (baytlarda) va yozish yoki o‘chirish kerakligi haqidagi belgini kiritish talab etiladi. Dastur so‘rov asosida diskdagi bo‘sh joy haqida ma’lumot berishi kerak.

Ko‘rsatma: diskdagi bo‘sh va band xotira joylari ro‘yxatini tuzib chiqish kerak bo‘ladi.

10–variant. Fayl tizimida fayllar kataloglari chiziqli ro‘yxat ko‘rinishida tasvirlangan.

Har bir fayl uchun katalogda quyidagi ma’lumotlar mavjud:

- fayl nomi;
- yaratilgan vaqti;
- faylga murojaatlar soni.

  Dastur quyidagilarni amalga oshirishi kerak:

- fayl katalogining dastlabki formasi;
- fayl katalogini chiqarish;
- yaratilgan vaqti ko‘rsatilgan vaqtdan kichik bo‘lgan fayllarni o‘chirish;
- fayllarni ko‘proq murojaat qilinganligi bo‘yicha tanlash.

  Dastur menyu yordamida ishlashi va kiritish jarayonidagi xatoliklarni nazorat qilishi kerak.

11–variant. Bir nechta dasturlar uchun yordamchi matnlar chiziqli ro‘yxat ko‘rinishida tashkil etilgan. Har bir yordamchi matn komponentasi termin (so‘z) va shu terminni tushuntiruvchi matndan tashkil topgan. Bir terminga bog‘langan matn satrlari soni birdan beshtagacha bo‘lishi mumkin.

Dastur quyidagilarni amalga oshirishi kerak:

- yordamchi matnning dastlabki tuzilishi;
- yordamchi matnni chiqarish;
- kiritilgan termin bo‘yicha tushuntiruvchi matnni chiqarish.

Dastur menyu yordamida muloqot qilish va kiritish vaqtidagi xatoliklarni nazorat qilishi kerak.

12–variant. Kvartira ayirboshlash byurosida kartoteka chiziqli ro‘yxat tarzida tashkil etilgan. Har bir kvartira haqidagi ma’lumotlarga quyidagilar kiradi:

- xonalar soni;
- qavat;
- maydoni;
- manzili;

Dastur quyidagilarni amalga oshirishi kerak:

- katalogning dastlabki formirovkasi;
- ayirboshlash uchun ariza qabul qilish;
- kartotekalar ichidan keraklisini topish: xonalar soni va qavati mos kelgan va maydoni orasidagi farq 10% bo‘lgan kvartiralarni chiqarish va ro‘yxatdan o‘chirish. Aks holda arizani ro‘yxatga qo‘shib qo‘yish;
- barcha ro‘yxatlarni chiqarish.

Dastur menyu yordamida muloqot qilish va kiritish vaqtidagi xatoliklarni nazorat qilishi kerak.

13–variant. Inglizcha-ruscha lug‘at ikkilik daraxt ko‘rinishida tashkil etilgan. Har bir komponenta inglizcha so‘zdan va shu so‘zga mos keluvchi

ruscha so‘zlardan shu bilan birga shu komponentaga qilingan murojaatlar sonidan tashkil topgan. Dastavval daraxt ingliz alfaviti tarzida tartiblangan. Qo‘llanilish vaqtida har bir komponentaga qilingan murojaat uning schetchigini bittaga oshiradi. Dastur quyidagilarni amalga oshirishi kerak:

- aniq bir murojaatlar soni bilan keltirilgan lug‘atning dastlabki holatini ta’minlash;
- quyidagi algoritm asosida lug‘atni quyidagi tarzda tashkil qilsin:
  a) eski lug‘atdan schetchik eng yuqori qo‘yilgan komponentani qidirish;
  b) topilgan komponenta yangi lug‘atga keltiriladi va eskisi o‘chiriladi;
  v) toki chiquvchi lug‘at vujudga kelmaguncha dastlabki a) bo‘limga o‘tish.
- chiquvchi va yangi lug‘atni chiqarish.

Dastur menyu yordamida muloqot qilish va kiritish vaqtidagi xatoliklarni nazorat qilishi kerak.

14–variant. Aholining so‘rov anketasi ikki xil so‘rovlar guruhidan iborat.

Birinchi guruh respondent haqidagi ma’lumotlardan tashkil topgan:

- yoshi;
- jinsi;
- ma’lumoti (boshlang‘ich, o‘rta, yuqori).

Ikkinchi guruh anketaning maxsus savollaridan tashkil topgan, bu savollarga faqat “HA” yoki “YO‘Q” deb javob berish kerak bo‘ladi.

Dastur quyidagilarni amalga oshirishi kerak:

- anketalarning dastlabki shaklidan chiziqli ro‘yxat tashkil etish;
- anketalar tahlili asosida quyidagi savollarga javob topilishi kerak: a) 40 yoshdan katta bo‘lgan oliy ma’lumotli erkaklardan nechtasi anketa savollariga “HA” deb javob berdi; b) 30 yoshdan kichik o‘rta ma’lumotli bo‘lgan ayollardan nechtasi anketa savollariga “YO‘Q” deb javob berdi; v) 25 yoshdan kichik boshlang‘ich

ma’lumotga ega bo‘lgan erkaklardan nechtasi anketa savollariga “HA” deb javob berdi;

- barcha anketa va savollarga javoblarni chiqarish.

Dastur menyu yordamida muloqot qilish va kiritish vaqtidagi xatoliklarni nazorat qilishi kerak.

15–variant. Kutubxonada mavjud kitobning ayni vaqtdagi holati haqidagi ma’lumotlar bilan ishlovchi dastur yarating. Har bir kitob quyidagi ma’lumotlardan tashkil topadi:

- UDK nomeri;
- Muallif ismi sharifi;
- Nomi;
- Nashr etilgan sanasi;
- Kitobning kutubxonadagi miqdori;

Dastur quyidagilarni amalga oshirishi kerak:

- kutubxonadagi barcha kitoblar haqidagi ma’lumotlarni ro‘yxat ko‘rinishida tasvirlash;
- kutubxonaga kelgan yangi kitob haqidagi ma’lumotlarni ro‘yxatga qo‘shish;
- chiqarib tashlangan kitob haqidagi ma’lumotlarni o‘chirib tashlash;
- so‘rov bo‘yicha kutubxonada mavjud kitoblar haqidagi ma’lumotlarni nashr etilgan sanasi bo‘yicha tartiblab chiqarish.

16–variant. Xalqaro telefon stansiyasida abonent kartotekasi telefon nomer va shu nomer egasi haqidagi ma’lumotlar bilan chiziqli ro‘yxat tarzida tashkil etilgan. Dastur quyidagilarni amalga oshirishi kerak bo‘ladi:

- kartotekani dastlabki formasini chiziqli ro‘yxat tarzida tashkil etish;
- barcha kartotekalarni chiqarishni amalga oshirish;
- telefon nomeri va suhbat vaqtini kiritish;
- telefon muloqoti uchun to‘lovni amalga oshirish uchun chiqarish.

Dastur menyu yordamida muloqot qilish va kiritish vaqtidagi xatoliklarni nazorat qilishi kerak.

17–variant. Xalqaro telefon stansiyasida abonent kartotekasi telefon nomer va shu nomer egasi haqidagi ma’lumotlar bilan ikkilik daraxt ko‘rinishida tashkil etilgan. Dastur quyidagilarni amalga oshirishi kerak bo‘ladi:

- kartotekani dastlabki formasini ikkilik daraxt ko‘rinishida tashkil etish;
- barcha kartotekalarni chiqarishni amalga oshirish;
- telefon nomeri va suhbat vaqtini kiritish;
- telefon muloqoti uchun to‘lovni amalga oshirish uchun chiqarish.

Dastur menyu yordamida muloqot qilish va kiritish vaqtidagi xatoliklarni nazorat qilishi kerak.

18–variant. Temir yo‘l vokzalining avtomatlashtirilgan tizimi poezdlar jo‘nashi haqidagi ma’lumotlardan tashkil topgan. Har bir poezd quyidagi ma’lumotlarga ega:

- poezd nomeri;
- ko‘zlangan bekat;
- jo‘nash vaqti.

Axborot tizimida axborotlar chiziqli ro‘yxat tarzida tashkil etilgan.

Dastur quyidagilarni amalga oshirishi kerak bo‘ladi:

- axborot tizimiga dastlabki ma’lumotlarni kiritish va ularni chiziqli ro‘yxat tarzida tashkil etish;
- barcha ro‘yxatni chiqarish;
- poezd nomeri kiritish va shu poezdga tegishli barcha ma’lumotlarni chiqarish;
- ko‘zlangan bekat nomini kiritish va shu bekatga kelishi kerak bo‘lgan poezd haqidagi ma’lumotlarni chiqarish.

Dastur menyu yordamida muloqot qilish va kiritish vaqtidagi xatoliklarni nazorat qilishi kerak.

19–variant. Temir yo‘l vokzalining avtomatlashtirilgan tizimi poezdlar jo‘nashi haqidagi ma’lumotlardan tashkil topgan. Har bir poezd quyidagi ma’lumotlarga ega:

- poezd nomeri;
- ko‘zlangan bekat;
- jo‘nash vaqti.

Axborot tizimida axborotlar ikkilik daraxt tarzida tashkil etilgan.

Dastur quyidagilarni amalga oshirishi kerak bo‘ladi:

- axborot tizimiga dastlabki ma’lumotlarni kiritish va ularni binar daraxt tarzida tashkil etish;
- barcha ro‘yxatni chiqarish;
- poezd nomeri kiritish va shu poezdga tegishli barcha ma’lumotlarni chiqarish;
- ko‘zlangan bekat nomini kiritish va shu bekatga kelishi kerak bo‘lgan poezd haqidagi ma’lumotlarni chiqarish.

Dastur menyu yordamida muloqot qilish va kiritish vaqtidagi xatoliklarni nazorat qilishi kerak.

20–variant. Quyidagi ishlarni bajaruvchi protsedura yoki funksiyani yozing:

- E parametrga bo‘sh bo‘lmagan T daraxtning eng chapda joylashgan bargining yozuvi o‘zlashtirilsin;
- E yozuvni T daraxtga kiritilish sonini aniqlash.

21–variant. Daraxt tugunlari haqiqiy sonlar bo‘lsin. Quyidagi ishlarni bajaruvchi protsedura yoki funksiyani yozing:

- daraxt barcha tugunlarini o‘rta arifmetik qiymatini hisoblash;
- qiymati yuqoridagi protsedura (funksiya)da hosil bo‘lgan songa teng elementni daraxtga qo‘shish.

*22–variant.*Daraxt tugunlari yozuvi – haqiqiy sonlar bo‘lsin. YOzuvi manfiy bo‘lgan daraxt tugunlarini o‘chiruvchi protsedura tuzing.

23–*variant.* Daraxt tugunlari yozuvi – haqiqiy sonlar bo‘lsin. Quyidagi ishlarni bajaruvchi protsedura yoki funksiyani yozing:

- bo‘sh bo‘lmagan darxtda yozuv qiymati maksimal yoki minimal bo‘lgan tugunlarni topish;
- daraxtning barcha barglari yozuvini chop etuvchi.

24–*variant.* Quyidagi ishlarni bajaruvchi protsedura yoki funksiyani yozing:

- daraxt tuguni yozuvi E bo‘lgan element T daraxtga tegishli yoki tegishli emasligini aniqlash;
- agar bunday yozuv topilmasa, uni daraxtga qo‘shish.

25–*variant.* Elementlari butun sonlardan iborat daraxt shakllantiring. Quyidagi funksiyalarni o‘z ichiga olsin:

- hosil bo‘lgan daraxt balandligini aniqlasin;

- terminaldagi elementlarni o‘sish tartibida tartiblagan holda bosmaga chiqaring.

2.4. 7-amaliy mashg‘ulot. Ma’lumotlarni saralash

Quyida tartibsiz elementlardan iborat massiv berilgan, ushbu massivning elementlarini bir nechta saralash usullari orqali amalga oshirilgan dasturlar kodi keltirilgan:

Indeksli saralash usuli:	**Natija:**

```
#include <iostream>
#include <conio.h>
using namespace std;
void insertSort(int a[], int N)
{  int x, i, j;
   for ( i=0; i < N; i++)
{ x=a[i];
        for ( j=i-1; j>=0 && a[j] > x; j--)
              a[j+1] = a[j]; a[j+1] = x; } }
int main ()
{ int i, n, mas[1000];
cout<<"\n Massiv uzunligini kiriting!\n";
cin>>n; cout<<"Massiv elementlarini
kiriting!\n";
       for (i=0; i<n; i++)
cin>>mas[i]; insertSort(mas,n);
cout<<"\n Saralangan massivni
chiqaring\n";         for(i=0; i<n; i++)
      cout<<mas[i];
  getch();
return 0;   }
```

```
 Masiv uzunligini kiriting = 10
Massiv elementlarini kiriting :
11
25
65
1
85
0
47
96
35
21
Saralangan massiv !
0
1
11
21
25
35
47
65
85
96
```

Pufaksimon saralash usuli:

natija:

```
 #include <iostream.h>
 #include <conio.h>
 void bubbleSort(int a[], long N)
 {
  long i, j;
  int x;
  for( i=0; i < N; i++)
  {   for( j = N-1; j > i; j-- )
   {     if ( a[j-1] > a[j] )    {
    x=a[j-1]; a[j-1]=a[j]; a[j]=x;    }  }  }
 }

 int main ()
 {
 int i, n, mas[1000];
 cout<<"\n Massiv uzunligini kiriting =
 ";
 cin>>n;
 cout<<"Massiv elementlarini kiriting :
 "<<endl;
 for (i=0; i<n; i++)
```

```
 Masiv uzunligini kiriting = 8
Massiv elementlarini kiriting :
45
0
32
56
98
8
47
3
Saralangan massiv
0
3
8
32
45
47
56
98
```

```
cin>>mas[i];
bubbleSort(mas,n);
cout<<"Saralangan massiv"<<endl;
for(i=0; i<n; i++)
cout<<mas[i]<<endl;
getch();
return 0;
}
```

Tanlash yoki ajratilgan oraliq asosida saralash usuli:

```
#include <iostream.h>
#include <conio.h>
void selectSort(int a[], int N)
{
 int i, j, k;
 int x;
 for( i=0; i < N; i++)
 {
  k=i; x=a[i];
  for( j=i+1; j < N; j++)
   if ( a[j] < x )
   {     k=j; x=a[j];    }
   a[k] = a[i]; a[i] = x;
 } }
main ()
{
int i, n, mas[1000];
cout<<"\n Massiv uzunligini kiriting = ";
cin>>n;
cout<<"Massiv elementlarini kiriting :"<<endl;
for (i=0; i<n; i++)
cin>>mas[i];
selectSort(mas,n);
cout<<"Saralangan massiv"<<endl;
for(i=0; i<n; i++)
cout<<mas[i]<<endl;
getch();
return 0;
}
```

Natija:

SHeyker saralash usuli:

```
#include <iostream.h>
#include <conio.h>

void shakerSort(int a[], int N)
{
 int j, k = N-1;
 int lb=1, ub = N-1; // Massivni saralanmagan qismi chegarasi
 int x;
 do {
        // Quyidan yuqoriga o'tish
  for( j=ub; j>0; j-- ) {
   if ( a[j-1] > a[j] ) {
        x=a[j-1]; a[j-1]=a[j]; a[j]=x;
        k=j;
   }   }
  lb = k+1;
  // Yuqoridan quyiga o'tish
  for (j=1; j<=ub; j++) {
   if ( a[j-1] > a[j] ) {
        x=a[j-1]; a[j-1]=a[j]; a[j]=x;
        k=j;
   }   }
  ub = k-1;
 } while ( lb < ub );
}
main ()
{
int i, n, mas[1000];
cout<<endl<<"Massiv uzunligini kiriting = ";
cin>>n;
cout<<"Massiv elementlarini kiriting!"<<endl;
```

```
for (i=0; i<n; i++)
cin>>mas[i];
shakerSort(mas,n);
cout<<"Saralangan massiv"<<endl;
for(i=0; i<n; i++)
cout<<mas[i]<<endl;
getch();
return 0;
}
```

Natija:

Saralashning takomillashgan usullari

Quiksort – tez saralash usuli

CHarlьzom Xoarom 1960 y.

Bu usulning **g‘oyasi** to‘g‘ridan–to‘g‘ri almashtirish algoritmidagi saralashga tegishli bo‘lib, uning asosini kalitlarni tanlangan tayanch kalitga nisbatan ikkiga ajratish tashkil qiladi.

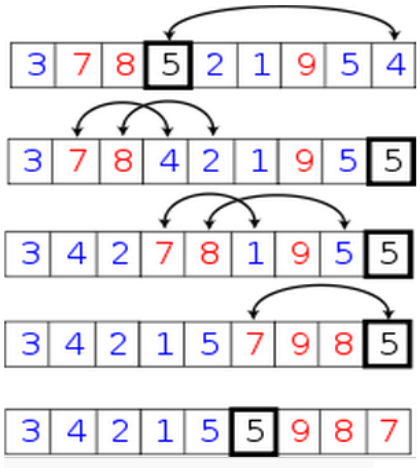
Mazkur algoritmning ishlash tamoyili quyidagicha:

1-qadam. massivdan tayanch elementni tanlash.

2-qadam. Massivni belgilangan tayanch kalitga nisbatan ikkiga ajratish, ya'ni, tayanch kalitga nisbatan kichik yoki teng elementlar chap qismga, katta yoki teng elementlar o'ng qism massivga joylashtiradi.

3-qadam. Massivda ikkita qism massivlar hosil qilindi.

4-qadam. Ikkita qism massivlar orasida qiyosiy saralash va qism massivlarning tarkibi elementlari orasida saralash rekursiv holda amalga oshiriladi.



5 dan chap tomonda kalitlari kichik, o'ng tomonda esa kalitlari 5 dan katta bo'lgan elementlar joylashadi (yuqoridagi chizma).

№12-topshiriq

1-variant. Ta’mirlash ustaxonasida bir nechta (N ta) mashina bor. Ular to‘g‘risida quyidagi ma’lumotlarga egamiz: raqami, markasi, egasini ismi, oxirgi marta ta’mirlanganligi sanasi (kuni, oyi, yili), ta’mirdan chiqishi lozim bo‘lgan sana (kun, oy, yil).

2-variant. To‘g‘ridan-to‘g‘ri qo‘shish usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqish (variantga mos ravishda): Mashina egalarini ismlari bo‘yicha alifbo tartibida joylashtirilsin va mos ravishda ularning mashinalari haqidagi ma’lumotlar chiqarilsin.

3-variant. Pufaksimon saralash usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqing: Avtomobillarni ta’mirlanish tartibi ishlab chiqilsin. Bu erda ta’mir tugashi sanasi qaysi avtomobil uchun ertaroq bo‘lsa shunga birinchi navabatda xizmat ko‘rsatiladi.

4-variant. Pufaksimon saralash usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqing: "Jiguli" markasidagi mashinalarni oldingi ta’mir sanasi bo‘yicha kamayish tartibida joylashtirilsin.

5-variant. Pufaksimon saralash usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqing: Oldingi ta’mir qilinganlar soni 2 ga teng bo‘lgan mashinalar raqamlari bo‘yicha kamayish tartibida joylashtirilsin.

6-variant. Tanlash yoki ajratilgan oraliq asosida saralash usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqing: Oldin ta’mir qilinmagan mashinalarni ta’mirdan chiqish sanasi bo‘yicha o‘sish tartibida joylashtiring.

7-variant. Indeksli saralash usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqing: "Mersedes" markali mashina egalarini alifbo bo‘yicha teskari tartibda joylashtiring.

8-variant. SHeyker saralash usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqing: Boshqalaridan oldinroq ta’mirlanadigan mashinalarni ularni markasi bo‘yicha alifbo tartibida joylashtiring (ta’mir tugatilishi sanasi 31.12.2007 dan erta).

9-variant. Indeksli saralash usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqing: "Jiguli" markasidagi mashinalarni raqamlari bo‘yicha o‘sish tartibida joylashtiring.

10-variant. To‘g‘ridan-to‘g‘ri qo‘shish usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqish: o‘tgan yildan beri ta’mirlanmagan mashinalarni ularning egalari ismlari bo‘yicha alifbo tartibida joylashtiring.

11-variant. To‘g‘ridan-to‘g‘ri qo‘shish usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqish: keyingi oyda ta’mirlanishi lozim bo‘lgan mashinalarni oxirgi marta ta’mirlanganlik sanasi bo‘yicha o‘sish tartibida keltiring.

12-variant. Indeksli saralash usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqing: Oldingi ta’mir qilinganlar soni 3 martadan ko‘p bo‘lgan mashinalarni egalari ismlarini alifbo bo‘yicha teskari tartibda joylashtiring.

13-variant. Pufaksimon saralash usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqing: "Mersedes" markasidagi mashinalarni raqamlari bo‘yicha kamayish tartibida joylashtiring.

14-variant. To‘g‘ri tanlov usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqilsin. N ta talabadan iborat guruh tuzilsin. Quyidagi ma’lumotlar berilgan: familiya, ism, tug‘ilgan yili, fanlar bo‘yicha bahosi: MS va A, oliy matematika, fizika, dasturlash, topshirgan sessiya umumiy bali saralash amalga oshirilsin.

15-variant. SHeyker saralash usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqilsin. N ta talabadan iborat guruh tuzilsin. Quyidagi ma’lumotlar berilgan: familiya, ism, tug‘ilgan yili, fanlar bo‘yicha bahosi: talabalar familiyalarini alifbo tartibida saralash amalga oshirilsin.

16-variant. SHeyker saralash usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqilsin. N ta talabadan iborat guruh tuzilsin. Quyidagi ma’lumotlar berilgan: familiya, ism, tug‘ilgan yili, fanlar bo‘yicha bahosi: Talabalar familiyalarini alifbo bo‘yicha teskari tartibda saralash amalga oshirilsin.

17-variant. To‘g‘ridan-to‘g‘ri qo‘shish saralash usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqilsin. N ta talabadan iborat guruh tuzilsin. Quyidagi ma’lumotlar berilgan: familiya, ism, tug‘ilgan yili, fanlar bo‘yicha bahosi: Talabalarni yoshi bo‘yicha o‘sish tartibida saralash amalga oshirilsin.

18-variant. To‘g‘ridan-to‘g‘ri qo‘shish saralash usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqilsin. N ta talabadan iborat guruh tuzilsin. Quyidagi ma’lumotlar berilgan: familiya, ism, tug‘ilgan yili, fanlar bo‘yicha bahosi: Talabalarni yoshi bo‘yicha kamayish tartibda.

19-variant. To‘g‘ridan-to‘g‘ri qo‘shish saralash usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqilsin. N ta talabadan iborat guruh tuzilsin. Quyidagi ma’lumotlar berilgan: familiya, ism, tug‘ilgan yili, fanlar bo‘yicha bahosi: Talabalarni umumiy bali bo‘yicha o‘sish tartibda saralansin.

20-variant. To‘g‘ridan-to‘g‘ri qo‘shish saralash usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqilsin. N ta talabadan iborat guruh tuzilsin. Quyidagi ma’lumotlar berilgan: familiya, ism, tug‘ilgan yili, fanlar bo‘yicha bahosi: Talabalarni umumiy bali bo‘yicha o‘sish tartibda saralansin.

21-variant. To‘g‘ridan-to‘g‘ri qo‘shish saralash usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqilsin. N ta talabadan iborat guruh tuzilsin. Quyidagi ma’lumotlar berilgan: familiya, ism, tug‘ilgan yili, fanlar bo‘yicha bahosi: Talabalarni birinchi imtihoni natijasi bo‘yicha o‘sish tartibda.

22-variant. To‘g‘ridan-to‘g‘ri qo‘shish saralash usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqilsin. N ta talabadan iborat guruh tuzilsin. Quyidagi ma’lumotlar berilgan: familiya, ism, tug‘ilgan yili, fanlar bo‘yicha bahosi: Talabalarni ikkinchi imtihoni natijasi bo‘yicha kamayish tartibda.

23-variant. To‘g‘ridan-to‘g‘ri qo‘shish saralash usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqilsin. N ta talabadan iborat guruh tuzilsin. Quyidagi ma’lumotlar berilgan: familiya, ism, tug‘ilgan yili, fanlar bo‘yicha bahosi: Talabalarni uchinchi imtihoni natijasi bo‘yicha o‘sish tartibda.

24-variant. To‘g‘ridan-to‘g‘ri qo‘shish saralash usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqilsin. N ta talabadan iborat guruh

tuzilsin. Quyidagi ma’lumotlar berilgan: familiya, ism, tug‘ilgan yili, fanlar bo‘yicha bahosi: Talabalarni to‘rtinchi imtihoni natijasi bo‘yicha kamayish tartibda.

25-variant. To‘g‘ridan-to‘g‘ri qo‘shish saralash usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqilsin. N ta talabadan iborat guruh tuzilsin. Quyidagi ma’lumotlar berilgan: familiya, ism, tug‘ilgan yili, fanlar bo‘yicha bahosi talabalarni birinchi va ikkinchi imtihoni natijalarini o‘sish tartibida saralansin.

26-variant. To‘g‘ridan-to‘g‘ri qo‘shish saralash usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqilsin. N ta talabadan iborat guruh tuzilsin. Quyidagi ma’lumotlar berilgan: familiya, ism, tug‘ilgan yili, fanlar bo‘yicha bahosi talabalarni birinchi va ikkinchi imtihoni natijalarini kamayish tartibida saralansin.

27-variant. Pufaksimon saralash usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqilsin: A massivni eng katta (eng kichik) elementini ekranga chiqarish dasturini tuzing.

28-variant. SHeyker saralash usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqilsin. A massiv elementlari qiymatlarini kamayish tartibida saralash dasturini tuzing.

29-variant. Indeksli saralash usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqing: A massivda elementlar berilgan. Mazkur massiv elementlaridan shunday V massiv shakllantiruvchi shunday dastur tuzingki, V massiv elementlari kamayish tartibida saralangan bo‘lsin.

30-variant. To‘g‘ridan-to‘g‘ri qo‘shish saralash usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqilsin. Elementlari o‘sish tartibida joylashgan A sonli massiv va a soni berilgan. a ni A massivga shunday qo‘shingki, tartiblanganlik buzilmasin.

31-variant. Pufaksimon saralash usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqilsin: elementlari o‘sish tartibida joylashgan A massivni, elementlari kamayish tartibida joylashgan massiv ko‘rinishda tezkor dastur tuzing.

32-variant. SHeyker saralash usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqilsin. Manfiy va musbat sonlardan tashkil topgan A massiv berilgan. Barcha manfiy sonlarni chiqarib, musbatlarini o‘sish tartibda joylashtiruvchi dastur tuzing.

33-variant. Indeksli saralash usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqing: Berilgan A massivdan ketma-ket sonlar olib, ulardan o‘sish tartibida shakllantirilgan V massiv hosil qiluvchi dastur tuzing.

34-variant. Pufaksimon saralash usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqilsin: Mualliflar ro‘yxati A massiv shaklida berilgan. Mualliflarni alifbo tartibida shakllantirish va shakllangan ro‘yxatni ekranga chiqarish dasturini tuzing.

35-variant. SHeyker saralash usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqilsin. Telefon stansiyasida n ta mijoz bor. Quyidagi shaklda ro‘yxat hosil qiluvchi dastur tuzing: telefon raqami, mijoz familiyasi (telefon raqamlari o‘sish tartibda joylashadi).

36-variant. Indeksli saralash usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqing: A massivni uzunliklari har xil bo‘lgan n ta so‘z tashkil qiladi. So‘zlarni uzunliklari bo‘yicha o‘sish tartibida joylashtiruvchi dastur tuzing.

38-variant. To‘g‘ridan-to‘g‘ri qo‘shish saralash usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqilsin. A massiv elementlari modullarini o‘sish va kamayish tartibida saralovchi dastur tuzing.

39-variant. SHeyker saralash usulidan foydalanib, saralashni amalga oshirish dasturini ishlab chiqilsin. Saralangan A sonli massivda har bir juft elementlar orasiga

chap elementdan katta o‘ng elementdan kichik son qo‘ying va hosil bo‘lgan massivni ekranga chiqaring.

2.5. 8-amaliy mashg‘ulot. Qidiruv usullari

1-misol. Binar qidiruv usulidan foydalanib massiv elementlari orasida belgilangan elementni qidirish dasturi quyida keltirilgan:

1-usul.

```
#include <iostream.h>
#include <conio.h>
int Binsearch(int a[], int N, int key, int *t)
{
int l=0, r=N-1, mid=(l+r)/2;
        while (l<=r)
{ *t+=1;
if (a[mid]==key) return mid;
if (a[mid]>key) r=mid-1;
        else l=mid+1;
        mid=(l+r)/2;
}
a[N]=key;
return N;
}
main ()
{
int i, N, mas[1000], key, P, t=0;
cout<<endl<<"Masiv uzunligini kiriting!"<<endl;
cin>>N;
cout<<"Massiv elementlarini kiriting!"<<endl;
for (i=0; i<N; i++)
cin>>mas[i];
cout<<"Qidiruv elementini kiriting!"<<endl;
cin>>key;
P=Binsearch(mas,N,key, &t);
if (P==N) cout<<"Bunday elementni massivga qo'shis lozim"<<" "<<P+1<<"
"<<t<<endl;
else cout<<"Bunday element bor"<<" "<<P+1<<" "<<t<<endl;
getch();
return 0;
}
```

2-usul.

```
#include <iostream.h>
#include <conio.h>
int InSeqsearch(int realArray[], int N, int kind[2][1000],int m,int key, int *t)
{
  int i=0,
    low = 0,
    hi = 0;
  while ((i<m) && (kind[0][i]<=key))
         {
          i++;
          (*t)++;
         }
 (*t)++;
  if (i==0)
   low=0;
  else
   low=kind[1][i-1];
  if (i==m)
   hi=N;
  else
   hi=kind[1][i]-1;

  for (int j=low; j<=hi; j++)
  {
        (*t)++;
        if( key==realArray[j] )
        {
          return j;
        }
  }
  return -1;
}
main ()
{
int i = 0 ,
  N = 0,
  mas[1000] = {0},
  kind[2][1000] = {0},
  key = 0,
  P = 0,
```

```
  index = 0,
  kindIndex = 0,
  t = 0;
cout<<endl<<"Massiv uzunligini kiriting!"<<endl;
cin>>N;
cout<<"Massiv elementlarini kiriting!"<<endl;
for (i=0; i<N; i++)
  cin>>mas[i];
cout<<"Qidiruv elementini kiriting!"<<endl;
cin>>key;
cout<<"Boshlangich qadamni kiriting! "<<endl;
cin>>P;
i = P-1;
while(i<N)
{
 kind[0][kindIndex] = mas[i];
 kind[1][kindIndex++] = i;
 i += P;
}
  index = InSeqsearch(mas,N,kind,kindIndex,key, &t);
  if (index == -1)
   cout<<"Bunday element massivda yuq "<< index <<" "<<t<<endl;
  else
   cout<<"Bunday element bor"<<" "<<index+1<<" "<<t<<endl;
  getch();
  return 0;
}
```

3-usul.

```
#include <iostream.h>
#include <conio.h>
#include <alloc.h>
#include <stdlib.h>
  struct TNode {
  int value;
  TNode* pnext;
 };
          //Ro'yhatga element qo'shish
void add2list(TNode **pphead, int val) {
  TNode **pp = pphead, *pnew;
  pnew = new TNode;
  pnew->value=val;
```

```
 pnew->pnext = *pp;
 *pp = pnew;
}
                //Ro'yhat elementlarini ekranga chiqarish
void print(TNode *phead)
        {
  TNode* p = phead;
  while(p) {
  cout <<" "<< p->value<<"-> ";
  p = p->pnext;
 }
 cout << endl;
}
              // Ro'yhatda element qidirish, C++
TNode* Find(TNode *phead, int x)
{
        TNode *p=phead;
         while(p)
        if (p->value==x) return p;
        else p = p->pnext;
        return 0;

}
                //Ro'yhat elementini o'chirish
void deleteList(TNode *phead) {
 if(phead) {
  deleteList(phead->pnext);
  if(phead)
        delete phead;
 }
}
                //Asosiy qism
int main() {int mas[1000], N, key; TNode* T;
 clrscr();
 TNode *phead = 0;
 cout<<"Ro'yhat uzunligini kirit"<<endl;
 cin>>N;
 cout<<"Elementlarni kirit!"<<endl;
 for(int j=0; j<N; j++)
 cin>>mas[j];
 for(int i = 0; i < N; ++i)
  add2list(&phead,mas[i]);
  cout<<"Qidiruv elementni kiriting!"<<endl;
```

```
 cin>>key;
 cout << "Elements of the list:" << endl;
print(phead);
T=Find(phead,key);
if (T==0) cout <<"Bunday element yoq"<<endl;
else cout <<"Bunday element bor"<<" "<<T->value<<" "<<T<<endl;
//deleteList(phead);
getch();
return 0;
}
```

4-usul.

```
#include <iostream.h>
#include <conio.h>
int search(int a[], int N, int key, int *t)
{
int i=0;
        while (i!=N)
{*t+=1;
if (a[i]==key) return i;
        else i++;
}
//a[N]=key;
return -1;
}
main ()
{
int i, N, mas[1000], key, P, t=0;
cout<<endl<<"Masiv uzunligini kiriting!"<<endl;
cin>>N;
cout<<"Massiv elementlarini kiriting!"<<endl;
for (i=0; i<N; i++)
cin>>mas[i];
cout<<"Qidiruv elementini kiriting!"<<endl;
cin>>key;
P=search(mas,N,key,&t);
if (P==-1) cout<<"Bunday elementni massivga qo'shis lozim"<<" "<<P+1<<"
"<<t<<endl;
else cout<<"Bunday element bor"<<" "<<P+1<<" "<<t<<endl;
getch();
return 0;
}
```

№11-topshiriq

1–*variant.* Ketma-ket qidiruv usulidan foydalanib, A massiv eng kichik elementini toping.

2–variant. A massiv berilgan, A massivdagi elementlar orasidan 30 dan katta elementlarni toping.

3–*variant.* Ketma-ket qidiruv usulidan foydalanib, A massiv elementlari orasidan 3 (3,6,9, ...) ga karralilarini ekranga chiqaring.

4–*variant.* Ketma-ket qidiruv usulidan foydalanib moduli 20 dan katta va 50 dan kichik elementlarni toping.

5–*variant.* Ketma-ket qidiruv usulidan foydalanib, A massiv elementlari orasidan 4 (4,8,...) ga karralilarini ekranga chiqaring.

6–*variant.* Ketma-ket qidiruv usulidan foydalanib 50 dan katta sonlarni ekranga chiqaring.

7–*variant.* Ketma-ket qidiruv usulidan foydalanib, A massivda elementni va taqqoslashlar sonini toping.

8–*variant.* Binar qidiruvdan foydalanib elementlarni tasodifiy ravishda toping.

9–*variant.* Mashina raqamlari ro‘yxati berilgan: 345, 368, 876, 945, 564, 387, 230. Binar qidiruvdan foydalanib berilgan raqamli mashina qaysi joyda turganini toping.

10–*variant.* Ro‘yxatda har ikkinchi elementni qidiring va taqqoslashalr sonini aniqlang.

11–*variant.* Binar qidiruvdan foydalanib berilgan kalitli elementni toping. Butun sonlar massivi berilgan. Boshiga qo‘yish va transpozitsiya usuliga protsedura tuzilsin. Berilgan masalani eching va masalada har ikkala usulda topilgan elementni ro‘yxat boshiga joylashtiring.

12–*variant.* Berilgan massiv elementlari ichidan eng katta elementni topilsin.

13–*variant.* Berilgan massiv elementlari ichidan eng kichik elementni topilsin.

14–*variant.* 11 ga butun bo‘linuvchi eng katta sonni toping (agar bunday sonlar ko‘p bo‘lsa, u holda ularning eng kattasini toping; agar bunday son mavjud bo‘lmasa – shunga mos ma’lumot chiqaring).

15–*variant.* 11 ga butun bo‘linuvchi eng katta sonni toping (agar bunday sonlar ko‘p bo‘lsa, u holda ularning eng kichigini toping; agar bunday son mavjud bo‘lmasa – shunga mos ma’lumot chiqaring).

16–*variant.* Qo‘shni elementlari ayirmasi 72 dan kichik bo‘lgan elementni toping. Agar bunday elementlar ko‘p bo‘lsa, u holda ularning eng kattasini toping; agar bunday element mavjud bo‘lmasa – shunga mos ma’lumot chiqaring.

17–*variant.* Qo‘shni elementlari bo‘linmasi juft son bo‘lgan elementni toping. Agar bunday elementlar ko‘p bo‘lsa, u holda ularning eng kattasi yoki eng kichigini toping; agar bunday element mavjud bo‘lmasa – shunga mos ma’lumot chiqaring.

18–*variant.* Qo‘shni elementlar ayirmasi juft bo‘lgan elementni toping. Agar bunday elementlar ko‘p bo‘lsa, u holda ularning eng kattasi yoki eng kichigini toping; agar bunday element mavjud bo‘lmasa – shunga mos ma’lumot chiqaring.

19–*variant.* Kerakli elementgacha bo‘lgan elementlarning o‘rta arifmetigi 12 ga teng bo‘lgan element topilsin. Agar bunday element mavjud bo‘lmasa – shunga mos ma’lumot chiqaring.

20–*variant.* 10 bo‘linuvchi maksimal elementni toping. Agar bunday element mavjud bo‘lmasa – shunga mos ma’lumot chiqaring.

21–*variant.* Qo‘shni elementlari ayirmasi juft va 3 ga bo‘linadigan elementni toping. Agar bunday element mavjud bo‘lmasa – shunga mos ma’lumot chiqaring.

22–*variant.* Kerakli elementdan keyingi elementlarning o‘rtacha kvadratik qiymati 10 dan kichik bo‘lgan elementni toping. Agar bunday elementlar ko‘p bo‘lsa, u holda ularning eng kattasini toping; agar bunday element mavjud bo‘lmasa – shunga mos ma’lumot chiqaring.

23–*variant.* Har bir x element uchun tg(x) qiymatini aniqlang va eng katta qiymatga ega bo‘lgan elementni 1-chi o‘ringa qo‘ying.

3. MURAKKAB TOPSHIRIQLAR

3.1. 9-amaliy mashg‘ulot

1-variant. SHaxmat doskasidagi ikkita katak koordinatalari berilgan. Minimum nechta yurishda o‘tishlar sonini ekranga chiqadi.

2-variant. Labirint bo‘yicha o‘tishni aniqlovchi dasturni tuzing.

Labirint kvadratlardan iborata matritsa ko‘rinishida berilgan. Har bir kvadrat yo ochiq yoki yopiq. YOpiq kvadratlarga o‘tish taqiqlangan. Agar kvadrat ochiq bo‘lsa, unga tomonlardan o‘tish mumkin, burchak bo‘yicha mumkin emas. Har bir kvadrat matritsadagi koordinatalari bilan aniqlanadi. Dasturda berilgan o‘tishlardan harakatlanib, labirintdan chiqish yo‘li topilsin. O‘tish topilgandan so‘ng, yo‘lni kvadrat koordinatalari ko‘rinishida chiqarsin. Yo‘lni saqlash uchun stekdan foydalaning.

3-variant. $\mathbf{K^n}$ ($1 \le K \le 9$ butun, $\boldsymbol{n}$ natural son) sonini birinchi va oxirgi raqamini aniqlang.

4-variant. Ikkita to‘g‘ri to‘rtburchak o‘z uchlarini koordinatalari bilan berilgan bo‘lsin. To‘rtburchaklarning biri ikkinchisi ichida joylasha oladimi?

5-variant. Otaning K ta o‘g‘li va $2n$ ta sigiri bor. $n=p*K$. Birinchi sigir 1 litr, ikkinchisi 2 litr va x.k., $2n$ - chisi $2n$ litr sut beradi. Ota sigirlarni o‘g‘illariga shunday

taqsimlab berish kerakki, har bir o‘g‘il teng miqdordagi sigirga va sutga ega bo‘lsin. Har bir o‘g‘il qanday raqamli sigirga ega bo‘lishini aniqlang?

6-variant. 0 va 1 lardan tashkil topgan A[N,M] massiv berilgan bo‘lsin. Bu jadval labirintni ifodalagan bo‘lib, yo‘llarni nollar, to‘siqlarni esa birlar bildiradi. Kompyuter labirintga kirib – chiqish yo‘llarini aniqlasin. U o‘zini yo‘lini sakkiz raqami bilan bildirsin, ya’ni yurish yo‘llaridagi nollarni sakkiz bilan almashtirsin.

7-variant. p sanoq tizimidagi ixtiyoriy butun sonni *q*-sanoq tizimiga o‘tkazsin. (*p*,*q*≤16).

8-variant. Ma’lumki, shaxmat taxtasining ixtiyoriy katagidagi ot bilan shaxmat taxtasini to‘la aylanib chiqishi mumkin. Bunda bitta katakka faqat bir marta yurilsin. Otning yo‘lini aniqlang.

9-variant. k musbat butun son va *k* ta $a_1, a_2, \dots, a_k$ butun sonlar berilgan. *S[m,n]=a(m) + a(m+2) +...+a(n-1)+a(n) (1≤m≤n≤k)*

yig‘indining mumkin bo‘lgan eng katta qiymatini toping.

10-variant. SHaharda sariyog‘ni xaridorga bir bo‘lakdan sotiladi, biroq yana sotib olishni xohlagan odam navbatga turib, bir necha martadan sariyog‘ harid qilishi mumkin. Do‘konga *N* bo‘lak sariyog‘ olib kelib, sotuv boshlangan. Oldiniga xaridorlar yo‘q edi, keyin esa har t_1 vaqt oralab, bittadan kela kela boshladilar. Xaridorlar sariyog‘ xarid qilgandan so‘ng, xaridor navbatning oxiridan yangidan turib oladi. Har bir xaridorga xizmat ko‘rsatish uchun t_2 vaqt sarf bo‘ladi.Agar bir vaqtni o‘zida ikki kishi navbatga turmoqchi bo‘lsa, harid qilishga ulgurgani keyin, yangi kelib qo‘shilmoqchi bo‘lgan kishi oldin turadi. Eng oxirgi bo‘lak sariyog‘ni sotib olgan haridorni tartib raqamini aniqlang. Do‘kondan nechta haridor sariyog‘ bilan qaytgan. Qaysi haridorlarga eng ko‘p bo‘laklar nasib etganini aniqlang.

11-variant. Mos ravishda oy, kun va yilni ifodalovchi *m,d, y*-natural sonlar berilgan. Berilgan sanaga mos keluvchi hafta kunini aniqlab beruvchi dastur tuzilsin.

12-variant. *a* va *b* natural sonlar berilgan. Ularning har biri ko‘pi bilan 60 tagacha raqamdan tashkil topgan. SHu sonlarning ko‘paytmasi hisoblansin va bosib chiqarilsin.

13-variant. *m* va *n* butun sonlar berilgan (ikkalasi ham nolga teng emas). *m/n* ifodaning qiymatini o‘nli kasr ko‘rinishida quyidagicha aniqlang:

m/n=c. $c_1 c_2 \dots c_n \quad (q_1, q_2, \dots, q_m)$, bu erda, *c* - sonning butun qismi, c_i -davrdan oldingi raqamlar ($1 \le i < p$), q_j - davrdagi raqamlar ($1 \le j < t$). Davrdagi raqamlar miqdori *t* alohida qayd etilsin.

14-variant. Bir domino (o‘yini) to‘plamidan uning g‘ishtcha shaklidagi etti donasi berilgan. Bilamizki, g‘ishtchalar o‘rtasidagi chiziq ularni ikki qismga ajratadi, har bir qismida bittadan oltitagacha doira shaklidagi chuqurchalari bo‘ladi, yoki tekis holda bo‘lishi ham mumkin. (Biz kompyuter uchun birdan oltigacha raqamlar yozilgan, yoki hech qanday raqam yozilmagan holni ko‘rishimiz mumkin.) Berilgan 7 dona domino g‘ishtchalaridan mumkin bo‘lganicha zanjirlar tuzing, zanjirda ikki dona g‘ishtcha bir-biri bilan ulanishi uchun ularni ulanayotgan qismlaridagi chuqurchalar miqdori teng bo‘lishi zarur.

15-variant. Istalgan matnni berilgan namuna bo‘yicha shifrlang.

*16-variant.*Tekislikda *n* ta to‘g‘ri to‘rtburchaklarning har biri istalgan diagonali uchlarining koordinatalari bilan aniqlanadi. To‘g‘ri to‘rtburchakning tomonlari koordinata o‘qlariga parallel joylashgan. Barcha to‘g‘ri to‘rtburchaklar uchun umumiy bo‘lgan sohaning yuzasi topilsin.

17-variant. O‘zining raqamlari kublarining yig‘indisiga teng bo‘lgan barcha uch xonali sonlarni toping. Masalan: 123 uchun 13+23+33=36 ; 153 uchun 13+53+33=125.

18-variant. Binoda *N* ta pod’ezd, *M* ta qavat va har bir zina maydonida *K* ta xonadon joylashgan. *A* xonadon raqam berilgan *P* pod’ezd raqami va *F* qavatning

raqamini aniqlang. Masalan: N=8, M=5, K=4, bo‘lsa, A=57 nomerli xonadon P=3 pod’ezdda va F=5 qavatda joylashgan.

19-variant. Yo‘l chiptalari 000000 dan 999999 gacha bo‘lgan olti xonali raqamlar yordamida nomerlangan. Hammasi bo‘lib qancha “baxtli” bilet mavjud? (Dastlabki uchta raqamlari yig‘indisi hamda oxirgi uchta raqam yig‘indisi bir xil. Masalan: 143080 — “baxtli”.) (Masalani hammasi bo‘lib, 3000 dan ortiq bo‘lmagan amal yordamida hal qiling.)

20-variant. A[1..20] butun sonli massiv hamda *m* butun son berilgan bo‘lsin. SHunday uchta natural *i, j* va *k* sonlarni topingki, *A[i]+ A[j]+A[k]=m* bo‘lsin. Agar bunday sonlar bo‘lmasa, bu haqda axborot berilsin.

21-variant. Umumiy ajdodlar. YAgona ota-onadan tarqalgan va faqat erkak jinsidagi avlodlar ko‘rsatilgan bitta oilaning genealogik daraxt sxemasi berilgan. Bu sxemadagi chiziqlar otani barcha o‘g‘illari bilan birlashtiradi. Ana shu genealogik daraxt sxemasini saqlash uchun samarali ma’lumotlar strukturasini yarating. Sxemadagi ikki odam uchun eng yaqin ajdodni aniqlang.

22-variant. N (N>1) butun son berilgan bo‘lsin. [1, N] kesmada *f* funksiya berilgan bo‘lib, uning grafigi uchlari (*1; f(1)*), (*2; f(2)*), …, (*N; f(N)*) bo‘lgan siniq chiziqdan iborat. (*f(i)* sonlari haqiqiy son hamda ustma-ust tushishi ham mumkin). Manfiy bo‘lmagan va butun *k* soni berilgan. 0Y o‘qining qaysi nuqtasidan 0X o‘qiga parallel hamda *f* funksiyaning grafigini roppa-rosa *k* marta kesib o‘tuvchi to‘g‘ri chiziqni o‘tkazish mumkin. Masalan: *k*=2 va rasmdagi grafik uchun javob: 0, 1, 2, (2, 2.5).

23-variant. Nurning qaytishi. Ikki shisha plastina ustma-ust qo‘yilgan. YUqoridagi plastinaga 999º gacha burchak ostida nur tushayapti. Agar nurning *n* marta qaytishi mumkin bo‘lsa, plastinalar ichidagi barcha nur qaytishlarini tasvirlang. Rasmda *n* = 3.

24-variant. a1, a2, : , an va *b* haqiqiy sonlar berilgan bo‘lsin.

a) Quyidagi tenglamaning barcha echimlarini toping:

| x - a1 | + | x - a2 | + ... + | x - an | = b.

b) Qanday holatda tenglama qancha echimga ega bo‘ladi?

25-variant. Haqiqiy sonlar o‘qida boshlang‘ich va oxirgi nuqtalari bilan *n* ta kesma berilgan bo‘lsin. Bu kesmalarni o‘qqa perpendikulyar bo‘lgan kamida nechta qoziqlarga o‘tkazish mumkin. Qoziqlarning holati va miqdorini aniqlash dasturini yozing. Qoziqlar miqdorini kamaytirish mumkin emasligini isbot qiling.

26-variant. Barcha natural sonlar yonma-yon yozilgan: 123...910111213... . M-chi o‘rinda qaysi raqam yozilgan? Masalan: 1 va 10-chi o‘rinda 1, 15-chida - 2, 100-chida - 5, 1000-chida esa -3 turibdi.

27-variant. p/q ko‘rinishidagi qisqarmaydigan kasrlar sonini aniqlang. Bu erda *p* va *q* natural son hamda $p+q \leq 100$.

28-variant. N ta qaroqchi xazina topib olishdi. Birinchi qaroqchi bitta tanga hamda qolgan tangalarning *n* dan birini oldi. Boshqa qaroqchilar ham xuddi shunday yo‘l tutishdi. O‘rtada qolgan tangalarni esa hammalari teng bo‘lib oldilar. Ana shunday bo‘lishga mos keladigan tangalarning eng kam soni *k* ni aniqlang.

Masalan: $n=2$ uchun $k=11$. SHunda 1-qaroqchi 1+5=6 ta tanga, 2-chisi esa 1+2=3

ta tanga oladi. O‘rtada qolgan 2 ta tangani teng bo‘lib olishadi.

29-variant. S va *T* satrlar berilgan bo‘lsin. Ular bo‘sh joy belgilarini hisobga olmaganda ustma-ust tushadimi? Bu satrlarni o‘zgartirish yoki yordamchi satr kiritish mumkin emas. Masalan “*ab b ca*” va “abb c a” satrlar uchun “ha”, “ab c” va “ac b” satrlar uchun “Yo’q”.

*30-variant.*Tekislikda ko‘pburchakning(x_i, y_j) koordinatalari $(i,j=1,\ldots,n)$ va *M(x, y)* nuqta berilgan. *M* nuqta berilgan ko‘pburchak ichida yotishi yoki yotmasligini aniqlang.

31-variant. k natural va *x0* haqiqiy sonlar hamda *n* - darajali ko‘phad o‘zining koeffitsientlari bilan berilgan bo‘lsin:

$$p(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0$$

SHu ko‘phadning $y=x^0$ nuqtada olingan *k* - tartibli hosilasini hisoblang.

*32-variant.*Bernulli sonlari quydagi rekkurent formula bilan topiladi:

$$B_0 + C_{k+1} B_1 + C_{k+1}^2 B_2 + \cdots + C_{k+1}^n B_n = 0 \,,$$

$$k = 1,2, \ldots .; \; B_0 = 1; \; C_n^k = \frac{n!}{(n-1)! * n!} \,.$$

M ta Bernulli soni topilsin.

Ro‘yxat, navbat va stek

Ishning maqsadi

1. Ro‘yxat bilan tanishish.
2. Navbat bilan tanishish.
3. Stek bilan tanishish.
4. Dinamik ma’lumotlat strukturasi bilan ishlash.

Topshiriq

1. Har bir talaba jurnaldagi tartib raqami bo‘yicha vazifalarning dasturini tuzishi lozim.
2. Hisobot shaklida oldin vazifa, uni bajarishda foydalanilgan funksiyalar, dastur kodi va natijalarni keltirib o‘tishi lozim.
3. 6-laboratoriya beriladigan barcha vazifalarni bitta hisobot shaklida topshiring.

Ro‘yxat

Ko‘p misollarda tarkibi, hajmi o‘zgaruvchan bo‘lgan murakkab konstruksiyalardan foydalanishga to‘g‘ri keladi. Bunday o‘zgaruvchan ma’lumotlar dinamik informasion strukturalar deb ataladi.

Eng asosiy dinamik informasion struktura bu **ro‘yxatdir**. Ro‘yxat uchun 3 ta oddiy amal aniqlangan.

- Navbatga yangi element joylashtirish
- Navbatdan element o‘chirish.
- Navbatni bo‘sh yoki bo‘sh emasligini aniqlash.

Ro‘yxatning asosiy xossasi shundan iboratki, ixtiyoriy elementini o‘chirish va ixtiyoriy elementidan keyin yangi element qo‘shish mumkin. Bu amallar boshqa elementlarga ta’sir o‘tkazmaydi.

Bir bog‘lamli ro‘yxatning har bir bo‘g‘inida ma’lumot va keyingi element adresi joylashgan. Agar bu ko‘rsatkich nol qiymatga ega bo‘lsa ro‘yxat oxirigacha o‘qib bo‘lingan bo‘ladi. Ro‘yxatni ko‘rib chiqishni boshlash uchun birinchi elementining adresini bilish yetarli.

Bir bog‘lamli ro‘yxat elementlari quyidagi strukturali tur orqali ta’riflangan obyektlardan iborat bo‘ladi

```
struct strukturali_tur_nomi
{
struktura elementlari;
struct strukturali_tur_nomi*ko‘rsatkich;
};
```

Bu ro‘yxat ma’lumot saqlovchi maydonlar, hamda keyingi element adresini saqlovchi ko‘rsatkichdan iborat.

Quyida ko‘riladigan bir bog‘lamli ro‘yxat har bir tuguni butun son va keyingi element adresi saqlanadigan ko‘rsatkichdan iborat. Eng birinchi elementi boshlang‘ich marker bo‘lib ma’lumot saqlash uchun emas balki ro‘yxat boshiga o‘tish uchun xizmat qiladi. Bu element hech qachon o‘chirilmaydi.

Ro‘yxat elementi quyidagi strukturali tur obyekti

```
struct slist_node
{
int info;
struct slist_node* next;
};
```

Ro‘yxat bilan ishlash uchun asosiy funksiyalar sarlavhalari:

void delete(struct slist_node*p) – berilgan tugundan keyingi elementni o‘chirish

void insert(struct slist_node*p, int nn) - berilgan elementdan keyin element qo‘shish

int empty(struct slist_node*beg) – ro‘yxat bo‘shligini tekshirish

Bu funksiya asosida quyidagi funksiyalar kiritilgan

struct slist_node* creat_slist(int size) – berilgan sondagi tugundan iborat ro‘yxatni yaratish. Tugun axborot qismi klaviatura orqali kiritilgan sonlar bilan to‘ldiriladi.

void free_slist(struct slist_node* beg) – ro‘yxatni o‘chirish

void print_slist(struct slist_node* beg) - ro‘yxat elementlarini ekranga chiqarish

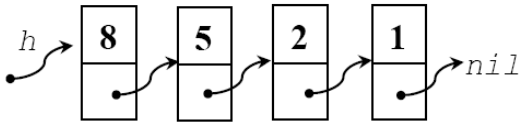
Stek

Stek deb shunday strukturaga aytiladiki, stekka kelib tushgan oxirgi elementga birinchi bo‘lib xizmat ko‘rsatiladi va stekdan chiqariladi. Mazkur ko‘rinishdagi xizmat ko‘rsatishni LIFO (Last input-First output, ya’ni oxirgi kelgan – birinchi ketadi) nomlash qabul qilingan. Stek bir tomondan ochiq bo‘ladi. Stekka bolalar piramidasi va kitoblar qatlamini misol keltirish mumkin.

6.1.-rasm. Stekka misollar

Steklar asosan arifmetik ifodali masalalarni tahlil qilishda, perebor (ajratish) li masalalarda hamda graflardagi algoritmlarda ishlatiladi.

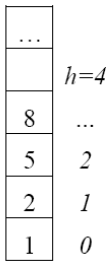
Stekni amalga oshirish uchun chiziqli ro‘yxatdan foydalaniladi, ya’ni ro‘yxatda boshida ko‘rsatkichda saqlanadi:



6.2.-rasm. Stekni chiziqli ro‘yxat orqali ifodalash

Qo‘shish va o‘chirish amallari ro‘yxat boshidan amalga oshiriladi. Bo‘sh stekda ko‘rsatkich boshi (0) ga teng.

Misol, agar stek 4 ta sondan tashkil topgan bo‘lsin. Ular o‘z navbatida 0, 1, 2 va 3 bilan raqamlangan. h = 4 ga teng, ya’ni stekda 4 ta son bor va keyingi qo‘shilayotgan sonni o‘rni stek massivida 4 bo‘ladi. Buni quyidagicha tasvirlash mumkin:

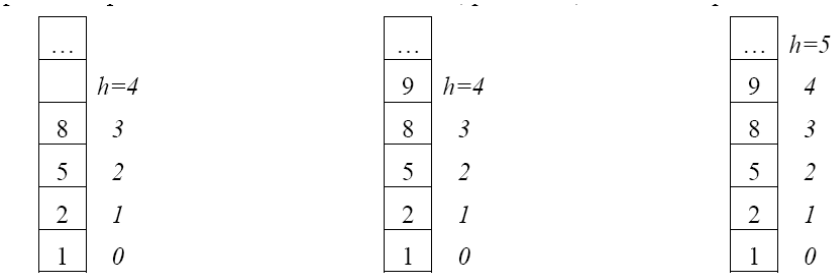


6.3-rasm. Stekni massiv orqali tasvirlash

Stek boshiga element qo‘shish uchun qiymatni yozamiz va h ko‘rsatkichni oshiramiz:

```
a[h ++] = k;
```

Stekga qiymati k=9 sonini qo‘shish jarayonini quyidagicha grafik ko‘rinishda tasvirlash mumkin:



6.4-rasm. Massivga element qo‘shish

Stek boshidan elementni chiqarish uchun teskari amaldan foydalanish lozim:

```
k = a[ -- h];
```

Bo‘sh stekning boshidagi ko‘rsatkichi h = 0 ga teng. Massivga element qo‘shish va o‘chirish davomida stek boshi massiv bo‘ylab ko‘chib turadi.

Universal stekning har bir tuguni axborot qismi void turidagi ko‘rsatkichdan iborat strukturadir:

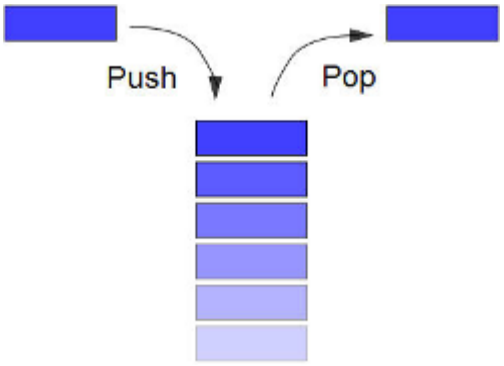
```
struct slist_node
{
 void* info;
 struct slist_node* pred;
};
```

Stek tugunining ro‘yxat tugunidan farqi shundaki, o‘zidan oldingi tugun adresini saqlovchi ko‘rsatkich ishlatilgan.

Stek o‘zi alohida struktura sifatida kiritilgan

```
struct stack
{
struct slist_node* end;
int size;
int width;
};
```

Bu yerda end oxirgi tugunga ko‘rsatkich, width ma’lumot hajmi, size navbatdagi elementlar soni.



Asosiy funksiyalar:

void pop(struct stack*p) – stek oxiridagi elementni o‘chirish.

void push(struct stack*p, void* val) –stek oxiriga element qo‘shish. Bu yerda val kiritilayotgan ma’lumotga ko‘rsatkich.

char* top(struct stack p) – stek oxiridagi tugun axborot qismiga ko‘rsatkich qaytarish.

int empty(struct stack p) – stek bo‘shligini tekshirish.

int size (struct stack p) – stek elementlari soni.

Bundan tashqari stekni inisiallash uchun quyidagi sarlavhali funksiya kiritilgan

void ini_stack (struct stack* p,int n) - Bu yerda n kiritilayotgan ma’lumotlar hajmi.

```
#include<stdio.h>
#include<stdlib.h>
#include<mem.h>
#include<conio.h>
struct slist_node
{
void* info;
struct slist_node* pred;
};

struct stack
{
struct slist_node* end;
int size;
int width;
};

void pop(struct stack*p)
{
struct slist_node* p1 = p->end;
if (p->size>0)
{
p->end = p1->pred;
free(p1);
p->size--;
}
}

void push(struct stack*p, void* val)
{
struct slist_node* p1 = (struct
slist_node*)malloc(sizeof(struct slist_node));
p1->info = malloc(p->width);
memcpy(p1->info,val,p->width);
if(p->size == 0)
{
p->end = p1;
}
else
```

```
{
p1->pred = p->end;
p->end = p1;
}
p->size++;
}

char* top(struct stack p)
{

return p.end->info;
}

int size(const struct stack* p)
{
return p->size;
}

int empty(struct stack p)
{
if (p.size>0) return 0;else return 1;
}
void ini_stack(struct stack* p,int n)
{
p->end = NULL;
p->size = 0;
p->width = n;
}
int main()
{
int i,m;
int*pi;
struct stack ps;
ini_stack(&ps,sizeof(int));
for(i = 0;i<5;i++) push(&ps,&i);
while(!empty(ps))
{
pi = (int*)top(ps);
printf("%d ",*pi);
pop(&ps);
}
getch();
return 0;
}
```

Namuna. Agar stekka kiruvchi yonma-yon turgan elementlari teng bo‘lsa ularni o‘chiring va “Yes”, aks holda “No” chiqaring.

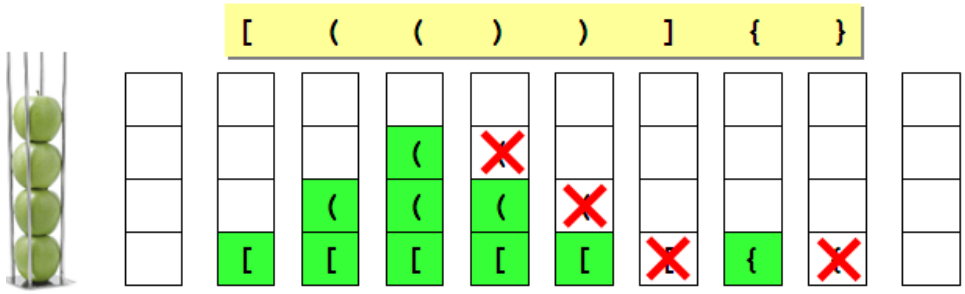
Kiruvchi	**Chiquvchi**
1122559977	Yes
1123558874	No

```
#include <iostream.h>
#include <conio.h>
char lifo[100], ch[100]; int i, top;
void push(char number){
     lifo[top] = number;
     top++;
     }
int main()
{
    cin>>ch;
    top = 0;
    for (i=0; i<strlen(ch); i++)
     {
       push(ch[i]);
       if (lifo[top-1] == lifo[top-2] && top>1)
        top -=2;
     }
    if (top==0) cout<<"Yes"; else cout<<"No";
  getch();
  return 0;
}
```

Namuna. Masala: [], {} va () uchta tiplardan tashkil topgan satr berilgan. Qavslar to‘g‘ri qo‘yilganligini tekshiring (boshqa simvollarga qaramagan holda). Masalan:

[()]{}][[({})]

Yechim: qavslar ichma ichligini hisoblagich. Agarda hisoblagich 0 bo‘lsa, qavs to‘g‘ri qo‘yilgan bo‘ladi.



Algoritm:

1) Boshida stek bo‘sh;

2) Sikl boshida satr barcha simvollarini tartib bo‘yicha ko‘rib chiqamiz;

3) Agar navbatdagi simvol ochiladigan qavs bo‘lsa, uni stek boshiga o‘tkazamiz;

4) Agar simvol yopiladigan qavs bo‘lsa, stek boshini tekshiramiz: u yerda berilgan qavsga mos ochiluvchi qavs bo‘lishi lozim (agarda bunday bo‘lmasa, u holda xatolik);

5) Agar stek oxiri bo‘sh bo‘lmasa, u holda ifoda noto‘g‘ri.

Dastur kodi

```
#include <iostream.h>
#include <conio.h>
#include <stdio.h>
struct Stack {
   char data[100]; // 100 ta simvolli stek
   int  size;      // elementlar soni
   };
int Push ( Stack &S, char x )
{
if ( S.size == 100 ) return 0;
S.data[S.size] = x;
S.size ++;
return 1;
}
char Pop ( Stack &S )
{
if ( S.size == 0 ) return char(255);
S.size --;
return S.data[S.size];
}
```

```
int isEmpty ( Stack &S )
{
if ( S.size == 0 )
     return 1;
else return 0;
}

int main()
{
char br1[3] = { '(', '[', '{' };
  char br2[3] = { ')', ']', '}' };
  char s[80], upper;
  int i, k, error = 0;
  Stack S;
  S.size = 0;
  printf("Qavsli ifodani kiriting > ");
  gets ( s );
for ( i = 0; i < strlen(s); i++ )
  {
  for ( k = 0; k < 3; k++ )
    {
    if ( s[i] == br1[k] )// agar ochilivchi qavs
bo‘lsa
      {
      Push ( S, s[i] ); // stekka kiritish
      break;
      }
    if ( s[i] == br2[k] )// agar yopilivchi qavs
bo‘lsa
      {
      upper = Pop ( S );  // yoqori elementni olish
      if ( upper != br1[k] ) error = 1;
      break;
      }
    }
  if ( error ) break;
  }
  if ( ! error && (S.size == 0) )
       printf("\nError\n");
  else printf("\nExcelent\n");
    getch();
    return 0;
}
```

Navbat

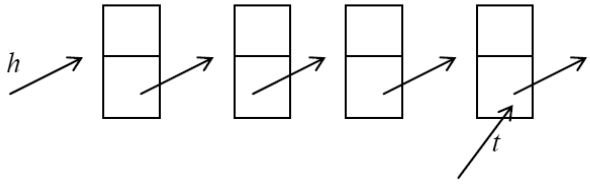
Navbat deb shunday strukturaga aytiladiki, navbatga kelib tushgan birinchi elementga birinchi bo‘lib xizmat ko‘rsatiladi va navbatdan chiqariladi. Navbatga biz kundalik hayotda magazinda, o‘qishda, ishda har doim duch kelamiz. Buni albatta dasturlashda ham ishlatish mumkin. Masalan, siz printerga bir vaqtning o‘zida uchta hujjatni chop etish bo‘yicha buyruq berdingiz. Ular o‘z navbatida navbatida hosil qiladi va eng birinchi navbat (masalan, yuzdan bir sekun oldin) olgani birinchi chop etiladi.

Navbat boshi – massivdan o‘chiriladigan elementning raqami.

Navbat oxiri – navbatga kiritiladigan massiv elementi.

Mazkur ko‘rinishdagi xizmat ko‘rsatishni navbat, ya’ni **FIFO** (*First input-First output*, ya’ni birinchi kelgan – birinchi ketadi) nomlash qabul qilingan. Navbat har ikkala tomondan ochiq bo‘ladi.

Navbatni amalga oshirish uchun ikkita ko‘rsatkichdan foydalaniladi: h boshi va t oxiri.



Qo‘shish oxiridan va o‘chirish ro‘yxat boshidan amalga oshiriladi. Agarda navbat bo‘sh bo‘lsa, h va t ning qiymati nol ga teng bo‘ladi (ya’ni navbat bo‘sh bo‘ladi). Shuning uchun element qo‘shishda ro‘yxat o‘zgarmaydi.

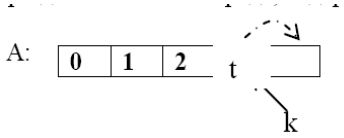
Navbatlar asosan arifmetik ifodali masalalarni tahlil qilishda, prebor (ajratish)li masalalarda hamda graflardagi algoritmlarda ishlatiladi.

Navbatni eng oddiy ko‘rinishda chegaralangan massiv A[0..N-1] ko‘rinishida tasvirlash mumkin.

Navbatni oxiridan yangi element qo‘shish uchun h va t zarur. Buning uchun qiymatni yozamiz va t ni oshiramiz:

```
A[t] = k;
t = (t + 1) mod N;
```

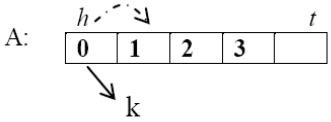
0, 1 va 2 dan tashkil topgan navbatga yangi element qo‘shganda k = 3 ga teng bo‘ladi:



Navbat boshidan o‘chirish:

```
k = A[h];
h = (h + 1) mod N;
```

Masalan, yuqoridagi navbatdan 0 elementini o‘chirgandan so‘ng quyidagini olamiz:



Universal navbat har bir tuguni axborot qismi void turidagi ko‘rsatkichdan iborat strukturadir:

```
struct slist_node
{
void* info;
struct slist_node* next;
};
```

Navbat o‘zi alohida struktura sifatida kiritilgan.

```
struct que
{
struct slist_node* beg;
struct slist_node* end;
int size;
```

```
int width;
};
```

Bu yerda beg birinchi tugunga ko‘rsatkich, end oxirgi tugunga ko‘rsatkich, width ma’lumot hajmi, size navbatdagi elementlar soni.

Asosiy funksiyalar:

void pop(struct que*p) – navbat oxiridagi elementni o‘chirish.

void push(struct que*p, void* val) –navbat boshiga element qo‘shish. Bu yerda val kiritilayotgan ma’lumotga ko‘rsatkich.

char* top(struct que p) – navbat boshidagi tugun axborot qismiga ko‘rsatkich qaytarish.

int empty(struct que p) – navbat bo‘shligini tekshirish.

int size (struct que p) – navbat elementlari soni.

```
#include<stdio.h>
#include<stdlib.h>
#include<mem.h>
#include<conio.h>
struct slist_node
{
void* info;
struct slist_node* next;
};
struct que
{struct slist_node* beg;
struct slist_node* end;
int size;
int width;
};
void pop(struct que*p)
{
struct slist_node* p1 = p->beg;
if (p->size>0)
{
p->beg = p1->next;
free(p1);
p->size--;
}
}
void push(struct que*p, void* val)
```

```
    {
    struct slist_node* p1 = (struct
slist_node*)malloc(sizeof(struct slist_node));
    p1->info = malloc(p->width);
    memcpy(p1->info,val,p->width);
    if(p->size == 0)
    {
    p->beg = p1;
    p->end = p1;
    }
    else
    {
    p->end->next = p1;
    p->end = p1;
    }
    p->size++;
    }
    char* top(struct que p)
    {
    return p.beg->info;
    }
    int size(const struct que* p)
    {
    return p->size;
    }
    int empty(struct que p)
    {
    if (p.size>0) return 0;else return 1;
    }
    void ini_que(struct que* p,int n)
    {
    p->beg = NULL;
    p->end = NULL;
    p->size = 0;
    p->width = n;
    }
    int main()
    {
    int i,m;
    int*pi;
    struct que ps;
    ini_que(&ps,sizeof(int));
    for(i = 0;i<5;i++) push(&ps,&i);
    while(!empty(ps))
```

```
{
pi = (int*)top(ps);
printf("%d ",*pi);
pop(&ps);
}
getch();
return 0;
}
```

6.1-variant. Stekka va navbatga oid variantlar

1. Uchta tipli (oddiy, kvadrat va figurali) qavslardan tashkil topgan ifoda berilgan. Shu berilgan ifodadagi qavslarni to‘g‘ri yoki noto‘g‘ri qo‘yilganligini aniqlovchi dastur tuzing. Bunda albatta ochilgan qavs yopilgan bo‘lishi lozim. Bunda ularni o‘rinlariga ham e’tibor berish lozim. Qavslar to‘g‘ri bo‘lsa “To‘g‘ri”, aks holda “Noto‘g‘ri” so‘zini chiqaring

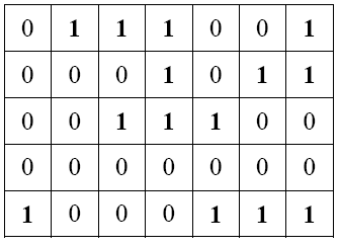
Kiruvchi	**Chiquvchi**
({}	Noto‘g‘ri
[{}([])]	To‘g‘ri
[(])	Noto‘g‘ri
()]}	Noto‘g‘ri

2. Postfiks formada (teskari polyak yozuvi) ifoda berilgan. Oldin son keyin esa amallar belgisi berilgan. Uning qiymatini topuvchi dastur tuzing.

Kiruvchi	**Chiquvchi**	**Изох**
62+	8	6+2=8
56-	-1	5-6=1
735*4+-	-12	7-(3*5+4)=-12

3. Matrisa shaklidagi dengizni tasvirlovchi xarita berilgan (0 – dengiz, 1-quruqlik). Orol sifatida faqat birlar bilan belgilangan sohani olish lozim. Berilgan xaritadan orollar soni va eng katta orol maydonini toping.

Kiruvchi	**Chiquvchi**	**Изох**

5 7 0 1 1 1 0 0 1 0 0 0 1 0 1 1 0 0 1 1 1 0 0 0 0 0 0 0 0 0 1 0 0 0 1 1 1	4 7	

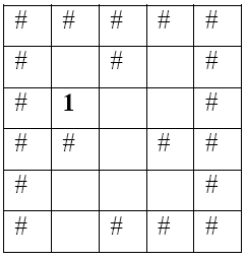
4. N ta doira berilgan. Bu doiralardan ustma-ust qo‘ygan holda piramida yig‘ilmoqda. Yig‘ishda ularning markazlari bitta nuqtada turishi lozim. Demak, n ta doira va ularning radiuslari berilgan. Agar ularni ustma-ust qo‘yganda tepadan qaraganda qaysi radiusli doiralarni ko‘rish mumkin.

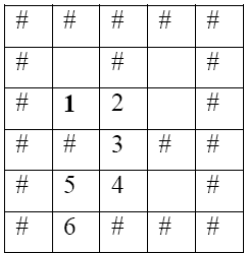
Kiruvchi	**Chiquvchi**
6 6 2 1 5 3 4	6 5 4
6 1 2 3 4 5 6	6

5. Satr ketma-ket keluvchi simvollardan tashkil topgan. Bitta simvol bir nechta joyda kelishi mumkin. Bu satrdan bir xil simvollarni o‘chirib joriy satrni chiqaring.

Kiruvchi	**Chiquvchi**
abccdeef	abdf
abccrrrrbf	af

6. M x N o‘lchamli to‘g‘ri to‘rtburchak berilgan. Uning har bir katagi # yoki 0 bo‘lishi mumkin. Berilgan katakdan chiqishning eng kam uzunligini chiqaring. Masalan, quyidagi labirint uchun javob oltiga teng.





Kiruvchi	**Chiquvchi**
5 6	6

##### #0#0# #100# ##0## #000# #0###	

7. Xemming ketma-ketligi berilgan. Bu ketma-ketlik o‘z ichiga 2, 3 va 5 ga bo‘linadigan natural sonlarni o‘z ochiga oladi. Bu ketma-ketlikning dastlabki N tasini chiqaruvchi dastur tuzing.

Kiruvchi	**Chiquvchi**
4	2 3 4 5
8	2 3 4 5 6 8 9 10
10	2 3 4 5 6 8 9 10 12 15

8. Xemming ketma-ketligi berilgan. Bu ketma-ketlik o‘z ichiga 2, 3 va 5 ga bo‘linadigan natural sonlarni o‘z ochiga oladi. Bu ketma-ketlikning dastlabki N ta hadining yig‘indisini chiqaruvchi dastur tuzing.

Kiruvchi	**Chiquvchi**
4	14
8	47
10	74

9. Xemming ketma-ketligi berilgan. Bu ketma-ketlik o‘z ichiga 2, 3 va 5 ga bo‘linadigan natural sonlarni o‘z ochiga oladi. Bu ketma-ketlikning berilgan N sonidan katta bo‘lgan eng kichik elementini va uning raqamini chiqaring.

Kiruvchi	**Chiquvchi**
4	5 4
8	9 7
10	12 9

10. Balandligi h=30 sm bo‘lgan qutiga, o‘lchami h=3 sm bo‘lgan yog‘och taxtadan (yog‘och taxta qutini yuzasi bilan bir xil o‘lchamda) nechta kiritilganda quti to‘lishini ko‘rsatuvchi dasturni tuzing.
11. 10 ta elementli bir o‘lchamli massiv berilgan. Ushbu massiv stek xususiyatiga ega bo‘lib, unga 11-element kiritilganda stek to‘lib ketganini ma’lum qiluvchi dastur tuzing.
12. Stek o‘lchami 5 ga teng. Unga 3 ta element kiritilganda top ko‘rsatkich qiymati nechaga teng bo‘ladi.
13. Stekda 7 ta element mavjud. Mazkur stekni elementlardan bo‘shatish dasturini tuzing.
14. Stekda 10 ta element mavjud. Elementlarni bitta-bitta tanlovchi dastur tuzing.
15. Avtomobilni maksimum tezligi 220 km/s. Mazkur avtomobil tezligi 60 km/s, 80 km/s, 100 km/s tezliklardan ortib borganda, mazkur tezlik qanday aholi punktlarida va qanday yo‘llar uchun mo‘ljallanganligi haqidagi xabarni ekranga chiqaruvchi dastur tuzing. Bunda har bir tezlik ko‘rsatkichi stekga element sifatida kiritilsin.
16. Qo‘shish (+) arifmetik amali asosida stekka element kiritish dasturini tuzing. Elementlarini kiritish jarayoni quyidagicha: birinchi va ikkinchi elementlar stekka kiritilgandan so‘ng, uchinchi elementni dastur birinchi va ikkinchi elementlar yig‘indisidan hosil bo‘lsin. To‘rtinchi element esa ikkinchi va uchinchi elementlar yig‘indisidan hosil bo‘lsin. Shu tariqa stekka 7 ta element kiritish dasturini tuzing.

Stek o‘lchami	Kiritilayotgan raqam va amal	Stek tarkibidagi elementlar
1	3	3
2	1	3,1
3	+	3,1,4,
4	+	3,1,4,5
5	+	3,1,4,5,9
6	+	3,1,4,5,9,14
7	+	3,1,4,5,9,14,23

17. Ayirish (-) arifmetik amali yordamida stekka element kiritish dasturini tuzing. Yuqoridagi variantdagi kabi birinchi va ikkinchi elementlari foydalanuvchi tomonidan kiritiladi. Qolgan elementni esa dastur o‘zi qo‘shni elementlarni juftligi ayirmasidan hosil bo‘lgan qiymatlarni kiritib borish natijasida stekni to‘ldirsin. Bunda ham stek elementlari 7 ta bo‘lguncha davom etsin. Bu jarayon quyidagi jadvalda keltirilgan:

Stek o‘lchami	Kiritilayotgan raqam va amal	Stek tarkibidagi elementlar
1	3	3
2	1	3,1
3	-	3,1,2,
4	-	3,1,2,1
5	-	3,1,2,1,1
6	-	3,1,2,1,1,0
7	-	3,1,2,1,1,0,1

18. Algebrik ifodadagi raqamlar ustida bajarilgan arifmetik amallar va raqamlar o‘rtasidagi mos munosabatlarni o‘rnatish orqali (har bir amal natijasida hosil bo‘lgan qiymat stekni bitta elementga bo‘lsin) stekka element kiritish dasturini tuzing. Bu jarayon quyidagi jadvalda keltirilgan:

Kiritilayotgan raqam yoki belgi	Stek tarkibidagi elementlar
4	4
6	4,6
3	4, 6,3
-	4,3
*	12
8	12, 8
6	12, 8,6
-	12, 2
2	12,2,2
/	12, 1
+	13

a) (1+2)-3*(4+6)+(7-5)+45/9. bunda har bir amal bajarish natijasida hosil bo‘lgan son stekka bitta element bo‘lib kiradi.

b) a=2, b= 3, s=4, d=5 raqamlar berilgan quyidagi ifodani hisoblash jarayonidagi har bir amal bajarilganda hosil bo‘lgan qiymatni stekka bitta element qilib kiritish dasturini tuzing: -a*b +s - d/a +c*d.

19. N ta akrobat navbatda ko‘rgazmali chiqishida minora hosil qilishdi. Ushbu minorani har bir bosqichidagi akrobatlarni sonli miqdori stekka bittadan element bo‘lib kirsin. Birinchi bosqich stekni birinchi elementga, ikkinchi bosqich ikkinchi va h.k., bo‘lsin.
20. Elementlar kiritilib, navbat shakllantirilsin. 15 ta elementdan iborat navbatni tashkil qiluvchi dasturni tuzing.
21. Elementlar kiritilib, navbat shakllantirilsin. Navbatning birinchi va oxirgi elementlaring o‘rnini almashtiruvchi dasturni tuzing.
22. 6 ta elementga mo‘ljallangan navbatga elementlar kiritish dasturini tuzing.
23. Navbatni o‘rtasidagi element o‘chirilsin. Agar navbat elementlari soni toq bo‘lsa bitta element, aks holda ikkita element navbatdan o‘chirilsin.
24. Elementlar kiritilib, navbat shakllantirilsin. Navbatni juft o‘rinda turgan elementlarini o‘chiruvchi dastur tuzilsin.
25. Elementlar kiritilib, navbat shakllantirilsin. Navbat o‘rtasiga '+' belgini joylashtiruvchi dastur tuzilsin.
26. Elementlar kiritilib, navbat shakllantirilsin. Eng kichik element topilib va undan keyin "0"ni joylashtiruvchi dastur tuzilsin.
27. 7 ta elementga mo‘ljallangan navbat berilgan. Elementlar kiritilib, navbat shakllantirilsin. Navbatning eng katta element topilib, undan keyin "***" belgini joylashtirilsin hamda navbatdagi o‘rnatilgan o‘lcham saqlanishini ta’minlovchi dastur tuzilsin.
28. 10 ta elementga mo‘ljallangan navbat berilgan. Elementlar kiritilib, navbat to‘ldirilsin va undagi eng kichik elementni topib o‘chiruvchi dastur tuzilsin.

29. Elementlar kiritilib, navbat shakllantirilsin. Ushbu navbatni o‘lchami 9 ga teng. Navbatdagi birinchi elementga teng barcha elementlar o‘chirilsin.
30. Elementlar kiritilib, navbat shakllantirilsin. Navbatning oxirgi elementiga teng barcha elementlarni o‘chiruvchi dastur tuzilsin.
31. Elementlar kiritilib, navbat shakllantirilsin. Navbatning o‘lchami 9 ga teng va eng katta elementga teng barcha elementlarni o‘chiruvchi dastur tuzilsin.

Nazorat uchun savollar

1. Ro‘yxatlar turlarini ko‘rsating.
2. Stek qanday prinsipda ishlaydi?
3. Navbat deb qanday strukturaga aytiladi?
4. Stekka misollar keltiring.
5. Stek qayerda ishlatiladi?
6. Navbatga misollar keltiring.
7. Navbat qayerda ishlatiladi?
8. Ro‘yxatga misollar keltiring.
9. Ro‘yxat qayerda ishlatiladi?

Sinflar va obyektlar. Konstruktor va destruktor. Sinf usullari

Ishning maqsadi

1. C++ da sinf yaratish va foydalanish bilan tanishish.
2. Turli sinflarni yaratish (Borland C++, Dev C++, Borland Builder).
3. Sinf o‘zgaruvchilari va funksiyalarini yaratish bilan tanishish.
4. Sinflardan foydalanishni o‘rganish.

Topshiriq

1. Har bir talaba jurnaldagi tartib raqami bo‘yicha vazifalarning sinfini yaratadi va dasturda ishlatishi lozim.

2. Har bir vazifani bajarishda foydalanilgan o‘zgaruvchi va funksiyalar haqida hisobotda ko‘rsatish lozim.
3. Hisobot shaklida oldin vazifa, uni bajarishda foydalanilgan funksiyalar, dastur matni va bajarishdan hosil bo‘lgan sinflarni keltirish lozim.
4. 7-laboratoriyada beriladigan barcha vazifalarni bitta hisobot shaklida topshiring.

Hisobot shakli

1. Laboratoriya ishining nomi.
2. Laboratoriya ishidagi topshiriq raqami
3. Topshriqni bajarishda foydalanilgan funksiyalar tavsifi.
4. Topshiriq kodi.
5. Topshiriq natijasi (bu laboratoriya ishi uchun grafik, shakl va animasiya).
6. Dasturning elektron versiyasi (Albatta bu dasturlar kompyuterda o‘qituvchiga ko‘rsatiladi. O‘qituvchini talab va takliflariga ko‘ra dastur to‘g‘rilanadi va so‘ngra hisobot tayyorlanadi.).

Sinflar va sinf a’zolari

Yangi tip sinfni e’lon qilish bilan tuziladi. Sinf - bu bir – biri bilan funksional orqali bog‘langan o‘zgaruvchilar va metodlar to‘plamidir. Sinflarga amaliyotdan ko‘pgina misollar keltirish mumkin. Masalan, avtomabilni g‘ildirak, eshik, o‘rindiq, oyna va boshqa qismlardan tashkil topgan kollekstiya yoki haydash tezligini oshirish, to‘xtatish, burish imkoniyatlariga ega bo‘lgan obyekt deb tasavvur qilish mumkin. Avtomobil o‘zida turli ehtiyot qismlarni va ularni funksiyalarini inkapsulyastiya qiladi. Avtomobil kabi sinfda ham inkapsulyastiya qator imkoniyatlarni beradi. Barcha ma’lumotlar bitta obyektda yig‘ilgan va ularga osongina murojaat qilish, ularni o‘zgartirish va ko‘chirish mumkin. Sizning sinfingiz bilan ishlovchi dasturiy qismlar, ya’ni mijozlar sizning obyektingizdan, uning qanday ishlashidan tashvishlanmasdan, bemalol foydalanishlari mumkin.

Sinf o‘zgaruvchilarning ixtiyoriy kombinastiyasidan, shuningdek boshqa sinflar tiplaridan iborat bo‘lishi mumkin. Sinfdagi o‘zgaruvchilar o‘zgaruvchi – a’zolar yoki xossalar deyiladi. `Sar` sinfi o‘rindiq, radiopriyomnik, shina va boshqa o‘zgaruvchi - a’zolardan iborat. O‘zgaruvchi – a’zolar faqatgina o‘zlarining sinflarida yotadilar. G`ildirak va motor avtomobilning qanday tarkibiy qismi bo‘lsa, o‘zgaruvchi – a’zolar ham sinfning shunday tarkibiy qismidir.

Sinfdagi funksiyalar odatda o‘zgaruvchi a’zolar ustida biror bir amal bajaradilar. Ular funksiya – a’zolar yoki sinf metodlari deb aytiladi. `Mashina` sinfi metodlari qatoriga `Haydash()` va `Tuxtatish()` metodlari kiradi. `Mushuk` sinfi hayvonni yoshi va og‘irligini ifodalovchi o‘zgaruvchi – a’zolarga ega bo‘lishi mumkin. Shuningdek, bu sinfning funksional qismi `Uxlash()`, `Miyovlash()`, `SichqonTutish()` metodlaridan iborat bo‘ladi.

Funkstiya – a’zolar ham o‘zgaruvchi a’zolar singari sinfda yotadi. Ular o‘zgaruvchi a’zolar ustida amallar bajaradi va sinfni funksional imkoniyatlarini aniqlaydi.

Sinfni e’lon qilish

Sinfni e’lon qilish uchun `class` kalitli so‘zi, undan so‘ng ochiluvchi figurali qavs, so‘ng xossalar va metodlari ro‘yxati ishlatiladi. Sinfni e’lon qilish yopiluvchi figurali qavs va nuqtali vergul orqali yakunlanadi. Masalan, `Mushuk` sinfini quyidagicha e’lon qilish mumkin.

```
class Mushuk
  {
   unsigned int itsYosh ;
   unsigned int itsOgirlik ;
   void Miyovlash()
  }
```

`Mushuk` sinfini e’lon qilishda xotira zaxiralanmaydi. E’lon qilish, kompilyatorga `Mushuk` sinfini mavjudligini, hamda unda qanday qiymatlar saqlashi mumkinligi (`itsYosh`, `itsOgirlik`) va u qanday amallarni bajarishi mumkinligi (`Miyovlash()` metodi) haqida xabar beradi. Bundan tashqari, bu

e’lon qilish orqali kompilyatorga `Mushuk` sinfining o‘lchami, ya’ni har bir `Mushuk` sinfi obyekti uchun kompilyator qancha joy ajratishi lozimligi haqida ham ma’lumot beradi. Masalan, joriy misolda butun qiymat uchun to‘rt bayt talab qilinsa, `Mushuk` sinfi obyekti o‘lchovi sakkiz bayt bo‘ladi.

(`itsYosh` o‘zgaruvchisi uchun to‘rt bayt, `itsOgirlik` o‘zgaruvchisi uchun to‘rt bayt). `Miyovlash()` metodi xotiradan joy ajratishni talab qilmaydi.

Obyektni e’lon qilish

Yangi turdagi obyekt xuddi oddiy butun sonli o‘zgaruvchidek aniqlanadi. Haqiqatan ham ixtiyoriy butun sonli o‘zgaruvchi quyidagicha aniqlanadi:

```
unsigned int MyVariable
// ishorasiz butun sonni aniklaymiz
```

`Cat` sinfidagi obyekt esa quyidagicha aniqlanadi:

```
Mushuk Frisky  //  Mushuk obyektini aniqlaymiz.
```

Bu dasturiy kodlarda `unsigned int` tipidagi `MyVariable` nomli o‘zgaruvchi va `Mushuk` sinfining `Frisky` nomli obyekti aniqlandi.

Ko‘pgina hollarda sinf va obyekt tushunchalarini ishlatishda chalkashlikka yo‘l qo‘yiladi. Shuning uchun, `obyekt sinfning biror bir ekzemplyari (nusxasi)` ekanligini yana bir bor ta’kidlash joiz.

Sinf a’zolariga murojaat qilish imkoni

`Mushuk` sinfining real obyektini aniqlaganimizdan so‘ng, bu obyektning a’zolariga murojaat qilish zaruriyati tug‘ilishi mumkin. Buning uchun bevosita murojaat `(.)` operatori qo‘llaniladi. Masalan, `Frisky` obyektining `Weight` o‘zgaruvchi - a’zosiga `50` sonini o‘zlashtirmoqchi bo‘lsak quyidagi jumlani yozishimiz lozim.

```
Fresky.Weight=50;
```

`Meow()` metodini chaqirish uchun esa

```
Frisky.Meow();
```

jumlasini yozish lozim.

Qiymat sinfga emas obyektga o‘zlashtiriladi

C++ tilida berilganlar tipiga qiymat o‘zlashtirilmaydi. Qiymat faqatgina o‘zgaruvchilarga beriladi. Masalan, quyidagi yozuv noto‘g‘ridir:

```
Int=s  // noto‘g‘ri
```

Kompilyator `int` tipiga qiymat o‘zlashtirilishi xatolik ekanligi haqida xabar beradi. Xuddi shu nuqtai – nazardan quyidagi yozuv ham noo‘rindir:

```
Cat.itsYosh= 5 // noto‘g‘ri
```

Agarda `Mushuk` obyekt bo‘lmasdan sinf bo‘lsa, yuqoridagi ifodani ham kompilyator xato deb hisoblaydi. O‘zlashtirish amalini bajarishda xatolikka yo‘l qo‘ymaslik uchun oldin `Mushuk` sinfiga tegishli `Frisky` obyektini hosil qilish va uning `ItsYosh` maydoniga `5` qiymatini berish lozim.

```
Mushuk Frisky;
Frisky.itsYosh=5;
```

Sinf a’zolariga murojaat qilish imkonini chegaralash.

Sinfni e’lon qilishda bir nechta kalit so‘zlardan foydalaniladi. Ulardan eng muhimlari `publis (ochiq)` va `private` (yopiq) kalit so‘zlari bo‘lib, ular orqali obyektning a’zolariga murojaat qilish imkoniyati chegaralanadi.

Sinfning barcha metodlari va xossalari boshlang‘ich holda yopiq deb e’lon qilinadi. Yopiq a’zolarga faqatgina shu sinfning metodlari orqaligina murojaat qilish mumkin. Obyektning ochiq a’zolariga esa dasturdagi barcha funksiyalar murojaat qilishlari mumkin. Sinf a’zolariga murojaat qilish imkonini belgilash juda muhim xususiyat bo‘lib, bu masalani yechishda uncha katta tajribaga ega bo‘lmagan dasturlarchilar ko‘pincha qiyinchiliklarga duch keladilar. Bu holatni batafsilroq tushuntirish uchun mavzuning boshida keltirilgan masalamizga qaytamiz.

```
class Mushuk
 {
    unsigned int itsYosh;
    unsigned int itsOgirlik;
    void Miyovlash();
```

```
}
```

Bu tarzda sinfni e’lon qilishda `itsYosh` va `itsOgirlik` maydonlari ham, `Miyovlash()` metodi ham yopiq a’zo sifatida aniqlanadi. Dasturda yuqoridagi tartibda `Mushuk` sinfi e’lon qilingan bo‘lsa va bu sinf ekzemplyari bo‘lgan obyektning `itsYosh` a’zosiga `main()` funkstiyasi tanasidan turib murojaat qilsak kompilyator xatolik ro‘y berganligi haqida xabar beradi.

```
Mushuk Baroq;
Baroq.itsYosh = 5 // Xatolik!
// Yopik  a’zoga  murojaat kilish mumkin emas.
```

`Mushuk` sinfi a’zolariga dasturning boshqa obyektlari tomonidan murojaat qilish imkonini hosil qilmoqchi bo‘lsak, uni `public` kalitli so‘zi orqali amalga oshiramiz.

```
class Mushuk
   {
     public:
      unsigned int itsYosh;
      unsigned int itsOgirlik;
      void Meow( );
   }
```

Endi `public` kalit so‘zi orqali sinfning barcha a’zolari `(itsYosh, itsOgirlik, Miyovlash())` ochiq a’zo bo‘ldi.

Konstruktorlar va destruktorlar

Butun sonli o‘zgaruvchini aniqlashning ikki xil yo‘li bor. Birinchisi, oldin o‘zgaruvchini aniqlash, keyin esa unga biror bir qiymat o‘zlashtirishdir. Masalan,

```
int a;  // o‘zgaruvchini aniqlash
........    // bu erda boshqa ifodalar bor
a=7; // o‘zgaruvchiga qiymat o‘zlashtiramiz.
```

Ikkinchisi, o‘zgaruvchi aniqlanishi bilan birga unga darhol qiymat o‘zlashtiriladi. Masalan,

```
int a=7;  //o‘zgaruvchini e’lon qilamiz
                  //va unga qiymat o‘zlashtiramiz.
```

Qiymat berish amali o‘zgaruvchi aniqlanishi bilan unga boshlang‘ich qiymat o‘zlashtirilishini anglatadi. Keyinchalik, bu o‘zlashtirilgan qiymatni o‘zgartirishingiz ham mumkin.

Sinfning o‘zgaruvchi–a`zosiga qanday qiymat o‘zlashtirildi? Buning uchun sinfda konstruktor deb ataluvchi maxsus funksiya – a`zo ishlatiladi. Zaruriy vaqtda konstruktor bir nechta parametrni qabul qiladi. Lekin hech qanday tipdagi qiymat qaytarmaydi. Konstruktor – bu sinf nomi bilan ustma – ust tushadigan sinf metodidir.

Sinfda konstruktorni e’lon qilinishi bilan destruktorlar ham aniqlanishi lozim. Agarda konstruktor sinf ob`ektini tuzish va uning o‘zgaruvchi – a’zolariga qiymat berish vazifasini bajarsa, destruktor mavjud ob`ektning xotiradan o‘chiradi. Destruktorlar sinf nomi oldiga tilda (~) belgisini qo‘yish orqali aniqlanadi. Destruktorlar hech qanday argument qabul qilmaydi va hech qanday qiymat qaytarmaydi.

Boshlang‘ich berilgan konstruktor va destruktorlar

Agarda siz konstruktor yoki destruktorni aniqlamasangiz, siz uchun bu ishni kompilyatorning o‘zi bajaradi. Standart konstruktor va destruktorlar birorta argument qabul qilmaydi va hech qanday amal bajarmaydi.

1- Namuna. Kasr sinfini yarating. Unda kamida 5 ta maydon va ularni ekranga chiqaruvchi, qayta ishlovchi usullarni yarating.

```
#include<iostream.h>
#include<conio.h>
using namespace std;
class kasr {
public:
int get(){
cout<<"\n"<<surat<<" "<<mahraj;
}
void set(int a, int b){
surat=a;
mahraj=b;
}
```

```
void addkasr(int a, int b, int c, int d){
surat= (a*d)+(c*b);
mahraj= b*d;
}
void subkasr(int a, int b, int c, int d){
surat= (a*d)-(c*b);
mahraj= b*d;
}
void multkasr(int a, int b, int c, int d){
surat= a*c;
mahraj= b*d;
}
void divkasr(int a, int b, int c, int d){
surat= a*d;
mahraj= b*c;
}
private:
int surat;
int mahraj;
};
//sinf konstruktori
kasr:: kasr (float mahraj)
{
 m= maxraj;
}
kasr::~kasr( ){ }

int main(){
kasr k;
int a; cin>>a;
int b; cin>>b;
k.set(a,b);
k.get();
k.addkasr(2,3,4,5);
k.get();
k.subkasr(3,2,6,2);
k.get();
k.multkasr(1,2,3,2);
k.get();
k.divkasr(1,2,4,2);
k.get();
getch();
return 0; }
```

7.1-variant. Sinf va obyektga oid topshiriqlar

7.1-jadval.

№	Masala sharti
1.	Talaba sinfini yarating. Unda kamida 5 ta maydon va ularni ekranga chiqaruvchi, qayta ishlovchi usullarni yarating.
2.	Avtomashina sinfini yarating. Unda kamida 5 ta maydon va ularni ekranga chiqaruvchi, qayta ishlovchi usullarni yarating.
3.	Mijoz sinfini yarating. Unda kamida 5 ta maydon va ularni ekranga chiqaruvchi, qayta ishlovchi usullarni yarating.
4.	Tovar sinfini yarating. Unda kamida 5 ta maydon va ularni ekranga chiqaruvchi, qayta ishlovchi usullarni yarating.
5.	Avia reys sinfini yarating. Unda kamida 5 ta maydon va ularni ekranga chiqaruvchi, qayta ishlovchi usullarni yarating.
6.	Dars sinfini yarating. Unda kamida 5 ta maydon va ularni ekranga chiqaruvchi, qayta ishlovchi usullarni yarating.
7.	Kitob sinfini yarating. Unda kamida 5 ta maydon va ularni ekranga chiqaruvchi, qayta ishlovchi usullarni yarating.
8.	Kompyuter sinfini yarating. Unda kamida 5 ta maydon va ularni ekranga chiqaruvchi, qayta ishlovchi usullarni yarating.
9.	Odam sinfini yarating. Unda kamida 5 ta maydon va ularni ekranga chiqaruvchi, qayta ishlovchi usullarni yarating.
10.	O`quv xonasi nomli sinfni yarating. Unda kamida 5 ta maydon va ularni ekranga chiqaruvchi, qayta ishlovchi usullarni yarating.
11.	Olimlar sinfini yarating. Unda kamida 5 ta maydon va ularni ekranga chiqaruvchi, qayta ishlovchi usullarni yarating.
12.	O`qituvchi sinfini yarating. Unda kamida 5 ta maydon va ularni ekranga chiqaruvchi, qayta ishlovchi usullarni yarating.
13.	Telefon sinfini yarating. Unda kamida 5 ta maydon va ularni ekranga chiqaruvchi, qayta ishlovchi usullarni yarating.
14.	Shahar sinfini yarating. Unda kamida 5 ta maydon va ularni ekranga chiqaruvchi, qayta ishlovchi usullarni yarating.
15.	Metro sinfini yarating. Unda kamida 5 ta maydon va ularni ekranga chiqaruvchi, qayta ishlovchi usullarni yarating.
16.	Nuqta sinfini yarating. Unda kamida 5 ta maydon va ularni ekranga chiqaruvchi, qayta ishlovchi usullarni yarating.
17.	Uchburchak sinfini yarating. Unda kamida 5 ta maydon va ularni ekranga chiqaruvchi, qayta ishlovchi usullarni yarating.
18.	To‘rtburchak sinfini yarating. Unda kamida 5 ta maydon va ularni ekranga chiqaruvchi, qayta ishlovchi usullarni yarating.
19.	Doira sinfini yarating. Unda kamida 5 ta maydon va ularni ekranga chiqaruvchi, qayta ishlovchi usullarni yarating.

20.	Aylana sinfini yarating. Unda kamida 5 ta maydon va ularni ekranga chiqaruvchi, qayta ishlovchi usullarni yarating.
21.	Kub sinfini yarating. Unda kamida 5 ta maydon va ularni ekranga chiqaruvchi, qayta ishlovchi usullarni yarating.
22.	Ko‘pburchak sinfini yarating. Unda kamida 5 ta maydon va ularni ekranga chiqaruvchi, qayta ishlovchi usullarni yarating.
23.	Matematika sinfini yarating. Unda kamida 5 ta maydon va ularni ekranga chiqaruvchi, qayta ishlovchi usullarni yarating.
24.	Hayvonlar sinfini yarating. Unda kamida 5 ta maydon va ularni ekranga chiqaruvchi, qayta ishlovchi usullarni yarating.
25.	Marketlar sinfini yarating. Unda kamida 5 ta maydon va ularni ekranga chiqaruvchi, qayta ishlovchi usullarni yarating.
26.	Bekat sinfini yarating. Unda kamida 5 ta maydon va ularni ekranga chiqaruvchi, qayta ishlovchi usullarni yarating.
27.	Geometriya sinfini yarating. Unda kamida 5 ta maydon va ularni ekranga chiqaruvchi, qayta ishlovchi usullarni yarating.
28.	Printerlar sinfini yarating. Unda kamida 5 ta maydon va ularni ekranga chiqaruvchi, qayta ishlovchi usullarni yarating.
29.	Modemlar sinfini yarating. Unda kamida 5 ta maydon va ularni ekranga chiqaruvchi, qayta ishlovchi usullarni yarating.
30.	Kompleks sinfini yarating. Unda kamida 5 ta maydon va ularni ekranga chiqaruvchi, qayta ishlovchi usullarni yarating.

Nazorat uchun savollar

1. Ochiq (public) va yopiq (private) o‘zgaruvchi – a’zolar orasida qanday farq bor?
2. Sinfning funksiya a’zolari qachon yopiq bo‘lishi lozim?
3. Sinfning funksiya a’zolari qachon ochiq bo‘lishi lozim?
4. Agar sinf class so‘zi yordamida ta’riflangan bo‘lsa ko‘zda tutilgan bo‘yicha komponentalari qanday murojaat huquqiga ega bo‘ladi?
5. Qaysi holda sinf usullari joylashtiriluvchi funksiya hisoblanadi?
6. Agarda sinfning ikkita obyektini e’lon qilsak, ularning o‘zgaruvchi a’zolari qiymati turlicha bo‘lishi mumkinmi?
7. Konstruktorlar xossalarini ko‘rsating.

8. Sinf obyektini hosil qilishda qanday funksiya chaqiriladi?

III. MUSTAQIL TA’LIM MAVZULARI

1-SEMESTR UCHUN

1. Matematik amallardan foydalanishni ko’rsatuvchi dastur tuzing.
2. Mantiqiy amallardan foydalanishni ko’rsatuvchi dastur tuzing.
3. Nisbat amallardan foydalanishni ko’rsatuvchi dastur tuzing.
4. Munosabat amallardan foydalanishni ko’rsatuvchi dastur tuzing.
5. Son absolyut qiymatini shartli amal yordamida hisoblovchi dastur tuzing.
6. Berilgan eps aniqlikda umumiy xadi 1/n! bo’lgan ketma-ketlik yig’indisini hisoblovchi dastur tuzing.
7. Umumiy xadi x/n! bo’lgan ketma-ketlik n ta xadi yig’indisini hisoblovchi dastur tuzing.
8. Kiritilgan n ta son qat’iy o’suvchi ekanligini tekshiruvchi dastur yarating.
9. Kiritilgan n simvoldan nechtasi o’nli harf ekanligini switch operatori yordamida hisoblovchi dastur tuzing.
10. Rekursiya yordamida Paskal uchburchagini hisoblovchi funksiya tuzing. Bu funksiya yordamida uchburchakni ekranga chiqaruvchi funksiya tuzib dasturda foydalaning.
11. Berilgan massiv qat’iy kamayuvchi ekanligini tekshiruvchi funksiya tuzing va dasturda foydalaning.
12. Matrisani vektorga ko’paytirish funksiyasini tuzing va dasturda foydalaning.
13. Berilgan satr telefon raqami ekanligini aniqlovchi funksiya tuzing va dasturda foydalaning.
14. Berilgan satr o’zgaruvchi ekanligi tekshiruvchi funksiya tuzing va dasturda foydalaning.
15. Berilgan jumladan eng qisqa so’z ajratib oluvchi funksiya tuzing va dasturda foydalaning.
16. Abituriyent (ismi, tug’ilgan yili, yiqqan bali, attestat o’rta bali) strukturasini yarating. Struktura tipidagi massiv yarating.

17. Massiv elementidan bir elementni olib tashlab, yangi massiv elementini tashkil eting.
18. Xodim (ismi, lavozimi, tug’ilgan yili, oyligi) strukturasini yarating. Struktura tipidagi massiv yarating.
19. Ko’rsatilgan familiyali elementni olib tashlang va ko’rsatilgan raqamli elementdan so’ng element qo’shing.
20. Mamlakat (nomi, poytaxti, axoli soni, egallagan maydoni) strukturasini yarating. Struktura tipidagi massiv yarating. Ko’rsatilgan aholi sonidan kichik bo’lgan elementni olib tashlang va ko’rsatilgan nomga ega bo’lgan elementdan so’ng element qo’shing.
21. Davlat (nomi, davlat tili, pul birligi, valyuta kursi) strukturasini yarating. Struktura tipidagi massiv yarating. Ko’rsatilgan nomga ega bo’lgan elementni o’chiring va fayl oxiriga ikkita element qo’shing.
22. Inson (ismi, yashash manzili, telefon raqami, yoshi) strukturasini yarating. Struktura tipidagi massiv yarating. Ko’rsatilgan yoshga ega bo’lgan elementni o’chiring va berilgan telefon raqamidagi elementdan oldin element qo’shing.
23. Berilgan satr o’zgaruvchi ekanligini aniqlovchi funksiya tuzing va dasturda foydalaning. Funksiya tanasida faqat ko’rsatkichlar ustida amallardan foydalaning.
24. Matrisani vektorga ko’paytirish funksiyasini yaratib dasturda foydalaning. Matrisa ko’rsatkichlar massivi sifatida kiritilsin.
25. Dinamik ravishda o’quvchilar familiyalari va baholari massivlarini hosil qiluvchi funksiyalar yarating. Hamma a’lochilar familiyalarini chiqaruvchi funksiya tuzib dasturda foydalaning.
26. Uchburchak dinamik massiv yordamida Paskal uchburchagini hisoblovchi funksiya tuzing. Bu funksiyadan tashqari uchburchakni ekranga chiqaruvchi funksiya tuzib dasturda foydalaning.

27. Dixotomiya usuli yordamida f(x) = 0 tenglamani yechish uchun funksiya tuzing. Funksiyaga ko’rsatkich parametr sifatida uzatilsin.

2- SEMESTR UCHUN

1- Chiziqli, tarmoqlanuvchi va siklik jarayonli masalalarni algoritmlashtirish
2- Takrorlanuvchi masalalarni ishlash
3- Tarmoqlanuvchi masalalarni ishlash
4- Mantiqiy masalalarni ishlash
5- Massivlarga doir masalalarni ishlash
6- Strukturaga doir masalalarni ishlash
7- Saralash algoritmlariga doir masalalarni ishlash
8- Qidirish algoritmlariga doir masalalarni ishlash
9- Rekursiyaga doir masalalarni ishlash
10- Stekka doir masalalarni ishlash
11- Navbatga doir masalalarni ishlash
12- Uzun sonlarga doir masalalarni ishlash
13- Satrli massivlarga doir masalalarni ishlash
14- Alfavit va xizmatchi so’zlar haqida tushunchalar bering.
15- O’zgaruvchilar haqida bilimlaringizni bayon eting.
16- Konstantalar haqida tushunchalar bering.
17- Operatorlar to’g’risida to’liq ma’lumot bering.
18- Ma’lumotlarni kiritish va chiqarish haqida ma’lumot bering.
19- Arifmetik amallar nima, ularga misollar keltiring.
20- Mantiqiy amallar, qiymat berish amallari nima, ularga misollar keltiring
21- Razryadli va shartli amallar nima, ularga misollar keltiring.
22- Nisbat amallar va amallar ustivorligi nima ularga misollar keltiring.
23- Tiplar bilan ishlashga misollar keltiring.

24- Funksiyalar haqida ma’lumotlar bering.

IV. GLOSSARIY

Termin	O’zbek tilidagi sharhi	Ingliz tilidagi sharhi
!=	Teng emas operatori; mantiqiy inkor amali qiymati bilan birhil.	The inequality operator; compares values for inequality returning a bool.
#define	Makro deriktivalarni belgilash	a directive that defines a macro.
#include	Bir source fayl ichida boshqa bir faylag murojatni amalga oshirish mexanizmi	a mechanism for textual inclusion of one source file into another.
+=	add-and-assign operatori; masalan a+=b vazifazi jihatdan a=a+b bilan bir xil	add-and-assign operator; a+=b is roughly equivalent to a=a+b.
.c file	Dastur jodini o’zida jamlovchi fayl	file containing definitions.
.cpp file	Dastur jodini o’zida jamlovchi fayl	file containing definitions.
.h file	Sarlavha fayli	header file
address	Hotira manzili	a memory location
aggregate	Konstruktorsiz massiv yoki struktura	an array or a struct without a constructor
algorithm	Hisoblashning aniq ketma-ketligi	a precise definition of a computation.
and	&& mantiqiy “va” (ko’paytirish) operatori sinonimi	synonym for &&, the logical and operator.
ANSI	Amerika milliy standart agentligi.	The American national standards organization.
application	Umumiy maqsadga ega dasturlar to’plami	a collection of programs seen as serving a common purpose (usually providing a common interface to their users)
bit	0 yoki 1 qiymatga ega birlik hotira	a unit of memory that can hold 0 or 1.
bool	Mantiqiy tip. Bu tip faqat rost yoki yolg’on qiymat qabul qiladi	the built-in Boolean type. A bool can have the values true and false.

Borland C++ Builder	C++ tilida visual dasturlashga ihtisoslashtirilgan IDE muhiti.	Borland's implementation of C++ together with proprietary libraries for Windows programming in an IDE
bug	Xatolik termini	colloquial term for error.
byte	Xotiradagi bir nechta belgilar yig'indisi	a unit of memory that can hold a character of the C++ representation character set.
C++	Tizimli dasturlashni protsedurali qo'llab quvvatlovchi dasturlash tili.	a general-purpose programming language with a bias towards systems programming that supports procedural programming, data abstraction, object-oriented programming, and generic programming.C++ was designed and originally implemented by Bjarne Stroustrup.
char	Belgili tip. Har bir belgi 8 bit, ya'ni baytga teng.	character type; typically an 8-bit byte.
char*	Char massiviga ko'rsatkich	pointer to a char or an array of char. Typically assumed to point to a C-style string.
cin	Standart kiritish oqimi	standard istream.
class	Foydalanuvchi belgilaydigan tur, sinf. Sinf foydalanuvchi funksiyasi, foydalanuvchi ma'lumotlari va kontentlari bo'lishi mumkin.	a user-defined type. A class can have member functions, member data, member constants, and member types.
compiler	C++ da yozilgan buyruqlarni mashina tiliga o'girib beruvchi vosita.	the part of a C++ implementation that produces object code from a translation unit.
const	Faqatgina bir marta qiymat berish mumkin bo'lgan o'zgaruvchilarni e'lon qilsh.	attribute of a declaration that makes the entity to which it refers readonly.
copy()	Nusxa olish operatori	Copy operator
UML	Унификация қилинган моделлаштириш тили	Unified Modeling Language

OMG	Объектларни бошқариш гуруҳи	Object Management Group
4GL	Тўртинчи авлод тили	Fourth-Generation Language
ANSI	Америка миллий стандартлаш институти	American National Standards Institute
AMPS		Advanced Mobile Phone Service
ERP	Корхона ресурсларини режалаштириш	Enterprise Resource Planning
CRM	Мижозлар билан ўзаро муносабатларни бошқариш	Customer Relations Management
SQL	Тузилмалашган сўровлар тили	Structured Query Language
OLAP	Ҳақиқий вақтда маълумотларга аналитик ишлов бериш	On-Line Analytical Processing
OLTP	Ҳақиқий вақтда транзакцияларга ишлов бериш	On-Line Transaction Processing
TCO	Эгалик қилишнинг ялпи қиймати	Total Cost of Ownership
JIT	Айни вақтида	Just-In-Time
LAN	Локал ҳисоблаш тармоғи	Local Area Network
MAN	Махаллий ҳисоблаш тармоғи	Metropolitan Area Network
WAN	Худудий ҳисоблаш тармоғи	Wide Area Network
ISO	Ҳалқаро стандартлаштириш ташкилоти	International Organization for Standardization
API	амалий дастурлаштириш махсус интерфейси	Application Programming Interface
WWW	Умумжаҳон ўргамчак тўри	World Wide Web
ASCII	Ахборот алмашишнинг Америка стандарти	American Standard Code for Information Interchange
LIFO	«Охирида келди, биринчи кетди» принципи	Last In, First Out
FIFO	«Биринчи келди, биринчи кетди» принципи	First In, First Out
PDA	персонал рақамли котиб	Personal Digital Assistant

V. ADABIYOTLAR RO'YXATI

I. Me'yoriy- huquqiy xujjatlar.

1. O'zbekiston Respublikasi Prezidentining «Oliy ta'lim muassasalarining rahbar va pedagog kadrlarini qayta tayyorlash va malakasini oshirish tizimini yanada takomillashtirish chora-tadbirlari to'g'risida» 2015 yil 12 iyundagi PF-4732-son Farmoni.
2. O'zbekiston Respublikasi Prezidentining 2010 yil 2 noyabrdagi "Oliy malakali ilmiy va ilmiy-pedagogik kadrlar tayyorlash tizimini yanada takomillashtirish chora-tadbirlari to'g'risida"gi PQ-1426-sonli Qarori.
3. Kadrlar tayyorlash milliy dasturi. O'zbekiston Respublikasi Oliy Majlisining Axborotnomasi, 1997 yil. 11-12-son, 295-modda.
4. O'zbekiston Respublikasi Prezidentining 2012 yil 24 iyuldagi "Oliy malakali ilmiy va ilmiy-pedagog kadrlar tayyorlash va attestasiyadan o'tkazish tizimini yanada takomillashtirish to'g'risida"gi PF–4456-son Farmoni.
5. O'zbekiston Respublikasi Vazirlar Mahkamasining 2012 yil 28 dekabrdagi "Oliy o'quv yurtidan keyingi ta'lim xamda oliy malakali ilmiy va ilmiy pedagogik kadrlarni attestasiyadan o'tkazish tizimini takomillashtirish chora-tadbirlari to'g'risida"gi 365- sonli Qarori.

II. Maxsus adabiyotlar.

1. Bjarne Stroustrup. Programming: Principles and Practice Using C++ (2nd Edition). Person Education, Inc. 2014. second printing, January 2015.
2. Harry Hariom Choudhary, Bjarne M Stroustrup. C++ Programming Professional.: Sixth Best Selling Edition for Beginner's & Expert's 2014.
3. Bjarne Stroustrup. The C++ Programming Language, 4th Edition. Person Education, Inc. 2013. Third printing, April 2014.
4. Nazirov Sh.A., Qobulov R.V., Bobojanov M.R., Raxmanov Q.S. C va C++ tili. "Voris-nashriyot" MCHJ, Toshkent 2013. 488 b.
5. Horstmann, Cay S. C++ for everyone / Cay S. Horstmann. Printed in the United States of America - 2nd ed. 2010. – P. 562.

6. Horton I. - Beginning Visual C++ 2012 / I.Horton. Published simultaneously in Canada. – 2012. –P. 988.

III. Internet resurslar.

1. http://www.stroustrup.com/4th.html
2. http://www.cplusplus.com/
3. http://acm.tuit.uz/ - дастурий ечим тўғрилигини автоматик тестловчи тизим.
4. http://acm.tuit.uz/forum/
5. http://acm.timus.ru/ – дастурларни тестловчи тизим.
6. http://codeforces.com/ - дастурий ечим тўғрилигини автоматик тестловчи тизим