ЎЗБЕКИСТОН РЕСПУБЛИКАСИ
ОЛИЙ ВА ЎРТА МАХСУС ТАЪЛИМ ВАЗИРЛИГИ

ФАРҒОНА ДАВЛАТ УНИВЕРСИТЕТИ

ФИЗИКА-МАТЕМАТИКА ФАКУЛЬТЕТИ

5480100 – амалий математика ва информатика йўналиши
09.305-гуруҳ талабаси Камолова Гулноранинг
ҳисболаш усуллари фанидан

КУРС ИШИ

Мавзу: Ҳисоблашга мўлжалланган алгоритмлар

Фарғона – 2012

MUNDARIJA

KIRISH

Yosh avlodning kompyuter savodxonligi darajasi ilmiy-texnika taraqqiyotini rivojlantirishga katta ta'sir etadi. Shunga ko`ra oliy ta'lim tizimi kurslarida kompyuterlardan foydalanish masalasi dolzarb bo`lib qolmoqda.

Hozirgi kunda ishlab chiqarishning turli texnologiyalaridagi muammolarni hal qilishda zamonaviy hisoblash texnikasi vositalarini keng qo`llash har bir mutaxassisdan, xoh u texnolog bo`lsin xoh iqtisodchi bo`lsin albatta zamonaviy kompyuterlardan, hamda axborot texnologiyalaridan kerakli darajada foydalanish mahorati va tegishli ma'lumotga ega bo`lishni talab qiladi.

Hisoblashga mo’ljallangan algoritmlar orqali ishlab chiqarishning turli sohalariga ta`alluqli bo`lgan masalalarni echish algoritmlarini va dasturini tuzish yo`llari hamda usullarini bilib olinadi.

Informatikada masala echish tushunchasi deganda axborotlarni qayta ishlab, natijani oldindan belgilangan ma'lum bir ko’rinishga olib kelish tushuniladi.

Kompyuterdan foydalanib masalani echish - yaratilgan algoritmga asoslangan holda dastlabki ma'lumotlar ustida avtomatik tarzda amallar bajarilib izlangan natija (natijalar) ko’rinishiga keltirish demakdir.

Biz bu bitiruv malakaviy ishimizda masalalarni kompyuterda echish bosqichlari, masalalarni kompyuterda echishning algoritmlash bosqichi, algoritm tushunchasi, algorim xossalari, algoritm turlari va hisoblshga oid bir qancha amaliy masalalarni hal qilishni nazarda tutganmiz.

I.1. Masalalarni kompyuterda echish bosqichlari

Kompyuterdan foydalanib "ilmiy - texnik masalani echish" tushunchasi keng ma'nodagi so’z bo’lib, quyidagi bosqichlarga bo’linadi.

Maqsad - bosqichlarning qaysi birlarini mutaxassis kompyuterdan foydalanmasdan va qaysi bosqichlarini kompyuterdan foydalanib bajarishini aniqlash hamda bosqichlarni to’la o’rganib chiqishdan iborat.

Ilmiy - texnik masalalarni (ITM) kompyuterdan foydalanib echish bosqichlari:

1. Masalaning qo’yilishi va maqsadning aniqlanishi;
2. Masalani matematik ifodalash;
3. Masalani echish uslubini ishlab chiqish, sonli usullarni tanlash;
4. Masalani echish algoritmini ishlab chiqish;
5. Ma'lumotlarni tayyorlash va tarkibini aniqlash (tanlash);
6. Dasturlash;
7. Dastur matnini va ma'lumotlarni axborot tashuvchiga o’tkazish;
8. Dastur xatolarini tuzatish;
9. Dasturning avtomatik tarzda kompyuterda bajarilishi;
10. Olingan natijalarni izohlash, tahlil qilish va dasturdan foydalanish uchun ko’rsatma yozish;

"Informatika" kursida 1- 4 bosqichlar qisqa ma'noda, xususiy holatlar, ko’p uchraydigan murakkab bo’lmagan muammolar uchun tushuntiriladi.

Bu bosqichlar tom ma'noda to’laligicha mutaxassislikni egallash davomida maxsus kurslar vositasida o’rgatiladi.

8- va 9-bosqichlarni bajarishda mutaxassis kompyuterdan foydalanadi.

7-bosqichda kompyuterdan foydalanish ham, foydalanmaslik ham mumkin.

ITM ni kompyuterda echish bosqichlarini alohida ko’rib chiqamiz.

1-bosqich. Masalaning qo’yilishi va maqsadni aniqlash.

Xalq xo’jaligining muayyan sohasi (texnika, iqtisod, lingvistika, ta'lim va x.k.) bo’yicha ishlayotgan (ishlagan) malakali va etakchi mutaxassis tomonidan bajariladigan ish, masalani qo’yish va maqsadni aniqlash uchun malakali mutaxassis bir necha kun, oy, xattoki yillab izlanishi mumkin.

Qo’yilgan maqsadni amalga oshirish uchun kerakli ma'lumotlar tarkibi (strukturasi), tuzilishi, ifodalanishi aniqlangan bo’lib, ular orasidagi bog’lanishlar aniq ifodalangan bo’lsa, “masala qo’yilgan” deb aytiladi.

2-bosqich. Masalani matematik ifodalash.

Bu bosqichda masalani echish uchun kerakli va etarli bo’lgan dastlabki ma'lumotlarning tarkibi, tavsifi, turi, tuzilishi xsobga olingan xolda matematik terminlarda ifodalanadi xamda masalani echishning matematik modeli yaratiladi.

Buning uchun har xil (sohasiga qarab) matematik apparat ishlatilishi mumkin.

Masalan iqtisod soxasidagi mutaxassislar - chiziqli dasturlash, dinamik dasturlash, stoxastik dasturlash, bashorat (prognoz) qilish bilan bog’liq masalalarni echish matematik apparatini bilishlari kerak; texnika soxasidagi mutaxassislar oddiy differentsial tenglamalar va ularning tizimlari, mexanikaning chegaraviy (kraevie) masalalarini, gaz dinamikasiga oid masalalarni, integral ko’rinishdagi

masalalarni ifodalash va echish uchun ishlatiladigan matematik apparatni to’liq tushunib etgan bo’lishi kerak.

Mutaxassis o’z soxasini xar tomonlama yaxshi o’rgangan va amaliy jixatdan puxta o’zlashtirgan va qo’llaniladigan har xil matematik apparatning barcha imkoniyatlarini to’liq tushunib yetgan va amaliyotga qo’llay oladigan bo’lishi kerak.

Bu bosqichda 2 ta asosiy savolga javob topish kerak:

1. Masalani ifodalash uchun qanday matematik strukturalar maqsadga muvofiq keladi?

2. Echilgan o’xshash masalalar bormi?

Tanlangan matematik struktura (apparat)da masalaning elementida ob'ektlari to’la ifodalanishi zarur.

3-bosqich. Masalani echish usulini ishlab chiqish, sonli usulni tanlash.

Agar dastlabki ma'lumotlar bilan izlanayotgan natijalar (miqdorlar, ma'lumotlar) o’rtasida aniq bog’liqlik (qonuniyat) o’rnatilgan bo’lib va masalani echish uslubi ishlab chiqilgan bo’lsa yoki o’sha bog’lanishni amalga oshirish uchun tayyor sonli usul (lar) tanlab olinib (masala uchun, masalaning bir qismi uchun) masalaning echish uslubi yaratilgan bo’lsa, “masalani echish uslubi ishlab chiqilgan” deyiladi.

Bunda: X - dastlabki ma'lumotlar; Y - natija, maqsad funktsiyasi, izlanayotgan miqdor (lar) bo’lsa, ular orasidagi bog’lanish Y= f (X) kabi olinishi mumkin.

f-dastlabki ma'lumotlar bilan natijani bog’lovchi qonuniyat, qoidalar majmuasi, ya'ni X ma'lumotlar ustida bajariladigan amallar ketma-ketligi yoki tanlab olingan usul.

Masalani echishning ishlab chiqilgan uslubi yoki tanlab olingan usulning to’g’riligi, samaradorligi keyingi bosqichlarda tekshirib aniqlanadi.

4-bosqich. Masalani echish algoritmini yaratish.

4-bosqichda asosan masalani echish algoritmi yaratiladi. Masalani echish algoritmi kompyuterning imkoniyatlarini, echish aniqligini xamda masalani kompyuterda echish vaqtini va qiymatini xisobga olgan xolda yaratilsa maqsadga muvofiq kelgan bo’lar edi.

Masalaning algoritmini yaratishda oraliq ma'lumotlarni iloji boricha kamaytirish, tashqi qurilmalar bilan bo’ladigan aloqalarni minimumga keltirish kerak.

Dasturning samarador va unumdorligi, masalani echish algoritmining qanchalik puxta tashkil qilinganligiga bog’liq.

3-4 bosqichlar bir-biri bilan jips, mustahkam bog’langan. Ya'ni yaratilgan uslubni har xil usullar bilan amalga oshirish mumkin, shu sababdan masalani echish uslubi va algoritmining bir nechta variantlari bo’lishi mumkin va keraklisi tanlab olinadi.

Murakkab masalaning algoritmini yaratishda qadamma-qadam oydinlashtirish uslubidan foydalangan ma'qul, har bir qadamda algoritmning tarkibi sodda va tushunarli bo’lib qolishiga erishmoq kerak.

Masalani algoritmlash jarayonida, algoritmning ba'zi bo’laklarini, lavhalarini, mantiqan alohida qismlarini ifodalashda tipik algoritmlar va amaliyotda tekshirilgan algoritmlardan iloji boricha ko’p foydalangan ma'qul.

Algoritmlashda modullik printsipidan foydalanish algoritmni o’qishda va dasturlashda qulayliklar yaratadi. Oxir oqibatda masalani echish algoritmi ishchi holatga keltiriladi, ya'ni algoritm grafik ko’rinishda biror algoritmik til vositasida ifodalash darajasiga keltiriladi.

Masalani algoritmlash - masalani kompyuterdan foydalanib echish algoritmini yaratish jarayonidir.

Algoritmlash - masalani echish bosqichi bo’lib, masalaga qo’yilgan shart va talablar asosida oxirgi natijani, masalaning echimini olish uchun ishlab chiqilgan algoritmlarni yaratish bilan shug’ullanadigan informatikaning bo’limidir.

5-bosqich. Ma'lumotlarni tayyorlash va tarkibini aniqlash.

Ma'lumotlarni tasvirlash usulini tanlash algoritmning bajarilishi bilan chambarchas bog’langan. Shu sababdan ma'lumotni tasvirlashning shunday turini, usulini tanlash kerakki, masalani echish jarayoni sodda va tushunarli bo’lsin.

Ma'lumotlar oddiy o’zgaruvchilar ko’rinishida (bu xol juda kam uchraydi), massiv ko’rinishida, aloxida ma'lumot fayllari (ketma-ket o’qiladigan yoki bevosita o’qiladigan) ko’rinishida axborot tashuvchida joylashgan bo’lishi mumkin.

6-bosqich. Dasturlash.

Masalani ishchi xolatga keltirilgan echish algoritmini tanlangan algoritmik til vositasida ifodalash (tavsiflash, tasvirlash) “dasturlash” deyiladi.

Algoritmning xar bir mayda bo’lagi algoritmik tilning operatorlari yordamida, tilning sintaksis va semantika qoidalari asosida yozib chiqiladi. Algoritm mukammal tuzilgan bo’lsa dasturlashda qiyinchilik tug’ilmaydi. Dasturlash jarayonida quyidagi takliflar inobatga olinsa xatolarni tuzatish jarayoni engillashadi.

1. Dastur umumiy bo’lishi kerak, ya'ni ma'lumotlarni aniq biror turiga bog’liq bo’lmasligi kerak, massivning chegara parametrlarini tekshirmoq lozim. Massiv elementlarining soni 0 yoki 1 bo’lib qolish, yoki yuqori chegarasidan oshib ketish xolati.

2. O’zgarmas kattalik xamda o’zgaruvchi kattalik ko’rinishida ishlatish. (Biror o’zgarmas kattalikni boshqasi bilan almashtirish zarurati bo’lib qolsa, dastur matnini chaqirib o’zgartirish kerak - bu noqulay xolat EXE, COM fayllarida aslo mumkin emas).

Dasturda kiritiladigan ma'lumotlarni nazorat qilish qismi bo’lishi kerak.

3. Dasturdagi arifmetik amallarni kamaytirish va dasturning ishlashini tezlatish uchun:

- darajaga oshirish amallari ko’paytirish amali bilan almashtirilgani ma'qul;
- bir xil ma'lumot bilan xisoblanayotgan arifmetik (algebraik) ifodalarni bir marta xisoblab qiymatini biror o’zgaruvchida saqlab ishlatish.
- takrorlashlarni tashkil qilishda takrorlanishning chegarasini tekshirish uchun ifodalardan emas balki oddiy o’uzgaruvchilardan foydalanish.

- takroriy xisoblashlar tarkibida uchraydigan va takrorlanish davomida qiymatini o’zgartirmaydigan ifodalarni takrorlanishdan tashqarida xisoblash.

4. Dasturning xar bir bo’lagi, moduli qismiga tushuntirishlar yozilgan bo’lishi kerak. Dasturdagi tushuntirishlar, masalani echish ketma-ketligini ifodalovchi mantiqiy ketma-ketlikdan iborat bo’lmog’i kerak.

Dasturdagi modullar, qismlar aniq ko’rsatilgan bo’lishi kerak. Takrorlanish boshi va takrorlanish oxiri aloxida qatorda turgani ma'qul.

7-bosqich. Dastur matnini va ma'lumotlarni axborot tashuvchiga o’tkazish.

Kompyuter uchun axborot tashuvchi vositalar bo’lib: perfokarta, perfolenta, magnitli tasma, magnitli disk (egiluvchi magnitli disk, magnitli karta), fleshkalar xizmat qilishi mumkin.

Dastur matni aloxida maxsus qurilmalar yordamida yoki kompyuterdan foydalanib axborot tashuvchiga o’tkaziladi.

8-bosqich. Dasturning xatosini tuzatish.

Bu bosqich masalani kompyuterda echish bosqichlari ichidagi ko’p vaqt talab qiladigan, mutaxassisdan sabr, qanoat, chidam, aql, zakovat, mantiqiy tez fikrlash, algoritmik tilning barcha imkoniyatlarini, tuzatish (otladka) qilish uslubini, yo’llarini, masalaning mag’zini ikir-chikirlarigacha mukammal bilishni talab qiladigan murakkab izlanuvchan jarayondir.

Bu bosqich “dasturni test bo’yicha tekshirish” deb xam yuritiladi. Dasturning to’g’ri ishlashi va yo’l qo’yilgan xatoliklarni aniqlab tuzatish algoritmni yaratishda yo’l qo’yilgan kamchiliklarni bartaraf qilish xamda tanlangan usulning yaroqli yoki yaroqsiz ekanligini aniqlab beruvchi jarayondir.

Test - maxsus tayyorlangan dastlabki malumotlar bo’lib, ular ustida amallar bajarish bilan masalaning echimi-natija olinadi. Test tayyorlash juda murakkab ish bo’lib, qo’lda hisob-kitob ishlarini bajarishni talab qiladi xamda dasturning xamma qismlarini, bo’laklarini, modullarini tekshirish shart.

Dasturning xatosini tuzatish bo’yicha yo’l - yo’riqlar:

1. Maxsus tayyorlangan ma'lumotlar asosida dasturni qo’lda echib chiqish (imkoni bo’lsa) yoki mantiqan alohida bo’lgan bo’laklarini, modullarini qo’lda xisoblash.

2. Dasturni va uning bo’laklarini, modullarini test yordamida tekshirish.

3.Dasturning kerakli joylariga bosib chiqarish buyrug’ini qo’yish (tuzatishlardan keyin olib tashlanadi).

4. Dasturning xatolarini tuzatishda, muloqot rejimida bajarilganda (STOP) to’xtash buyrug’idan foydalanish.

5. Dasturlash tilini va amal bajaruvchi tizimi (AT)ning maxsus xatolarni tuzatish imkoniyatlaridan foydalanish.

6. Xatolarni tuzatish jarayonida kam xajmdagi ma'lumotlar bilan ishlashni tashkil qilish.

9-bosqich. Dasturning avtomatik tarzda kompyuterda bajarilishi.

Kompyuter xatolari tuzatilib tayyorlangan dastlabki ma'lumotlardan foydalangan xolda masalaning echimini (echimlarini) avtomatik tarzda xisoblaydi.

Agar natijalar masalaning echimi uchun yaroqli deb topilsa masalani echish tugallangan xisoblanadi, aks xolda yuqoridagi bosqichlar qaytadan ko’rib chiqiladi.

10-bosqich. Olingan ma`lumotlarni izohlash, tahlil qilish va dasturdan foydalanish uchun yo’riqnoma yozish.

Masalani echish natijasida olingan sonlar yoki sonlar massivi, matnlar yoki matn ko’rinishidagi massivlar xar taraflama izoxlanib, tushuntiriladi. Dasturdan foydalanish uchun ko’rgazma yozish quyidagilarni o’z ichiga oladi:

- Dastur ishlashi uchun ma'lumotlarni tayyorlash usuli, tuzilishi aniq belgilangan;
- Dasturning ishlashi uchun kompyuterni sozlash yo’llari;
- Dasturni ishga tushirish va ishlash paytida bo’ladigan savol-javoblar;
- Dasturni ishlash jarayonida kelib chiqadigan xar xil xolatlarni bartaraf qilish yo’llari aniq va puxta tushunarli qilib yozilgan bo’lishi kerak.

Masalani echishning uchta bosqichini quyidagi misollarda ko’rib chiqamiz.

1-MISOL.

1. Masalaning qo’yilishi va maqsadning aniqlanilishi. Koptok 29, 5 m / sek tezlik bilan tepaga tik ravishda tepilgan. U qancha balandlikka ko’tariladi? (Havoning qarshiligi xisobga olinmasin).

2. Masalani matematik ifodalash.

Berilgan: Vo = 29, 5 m / sek. ; V = Vo.

Koptokni balandlikka ko’tarilish xarakatini ifodalovchi qonuniyat:

h = Vo*t-g*t2 / 2 (1)

bu erda: t - koptokning ko’tarilish vaqti, sek. ; g - erkin tushish tezlanishi (9, 8 m / sek);

3. Masalani echish usulini ishlab chiqish.

Koptokning tezligi eng yuqori balandlikka etganda nolga teng bo’ladi:

V= 0. Fizika kursidan ma'lumki, tezlik yo’ldan vaqt bo’yicha olingan xosila.

V = dh / dt. (2)

(1) dan xosila olsak

V = Vo - g*t (3)

(3) -ni nolga tenglab t ning qiymatini topamiz:

t = Vo / g (4)

(4)-dan t ni topib (1) ga qo’yamiz.

2-MISOL.

1. Masalaning qo’yilishi va maqsadni aniqlash.

X 0 Y koordinata tekisligida Y=0, X=a, X=b to’g’ri chiziqlar va $Y = \sqrt{X}$ egri chiziq bilan chegaralangan shaklning yuzasi aniqlansin.

2. Masalani matematik ifodalash.

Masalaning qo’yilishidan ma'lumki bu shakl egri chiziqli trapetsiyadir. Uning yuzasini topish aniq integral yordamida quyidagicha xisoblanadi:

$$S = \int_a^b \sqrt{x}dx$$

bu erda: a - integralning quyi chegarasi; b - integralning yuqori chegarasi.

3. Masalani echish usulini ishlab chiqish (tanlash).

Bu turdagi masalalarni echishda to’rtburchaklar, trapetsiya yoki Simpson taqribiy usullaridan biri tanlab olinadi va yuza xisoblanadi.

I. 2. Masalalarni kompyuterda echishning algoritmlash bosqichi.

“Algoritmlash” deganda masalani biri ketidan boshqasini bajariladigan xamda oldingisining natijalari keyingilarining bajarilishida ishlatiladigan bosqichlar ketma-ketligiga keltirish tushuniladi. Ayni paytda bu bosqichlardagi amallarni kompyuter bajara olishi ko’zda tutilishi kerak.

Kengroq ma`noda qaraydigan bo’lsak algoritmlash, o`zidan oldingi bosqich - masalani echish usulini tanlash bosqichi xam, o`zidan keyingi bosqich - kompyuterning xususiyatlarini xisobga olgan xolda boshlang’ich, oraliq va natijaviy axborotlarni tuzilishining ifoda shakllarini tanlashni xam o’z ichiga oladi.

Algoritmlash bosqichining natijasi masalani echish algoritmi bo’ladi, yani bu bosqichda masalani echish algoritmi ishlab chiqiladi. Bunda masalani matematik qo’yilishi va tanlangan usul qidirilayotgan natijani olishga olib keladigan xarakatlar ketma-ketligini aniqlash uchun asos bo’lib xizmat qiladi.

I.3. “Algoritm” tushunchasi va unga misollar.

Algoritm deb, masalani echish uchun bajarilishi lozim bo’lgan amallar ketma-ketligini aniq tavsiflaydigan qoidalar tizimiga aytiladi.

Boshqacha aytganda, algoritm –boshlang’ich va oraliq ma`lumotlarni masalani echish natijasiga aylantiradigan jarayonni bir qiymatli qilib, aniqlab beradigan qoidalarning biror bir chekli ketma-ketligidir.

Buning moxiyati shundan iboratki, agar algoritm ishlab chiqilgan bo’lsa, uni echilayotgan masala bilan tanish bo’lmagan biron bir ijrochiga, shu jumladan kompyuterga xam bajarish uchun topshirsa bo’ladi va u algoritmning qoidalariga aniq rioya qilib masalani echadi.

Masalan, ko’rib o’tilgan birinchi misolni echish algoritmini quyidagicha bayon qilsa bo’ladi:

1) kompyuter xotirasiga Vo va g o’zgaruvchilarning sonli qiymatlari kiritilsin;

2) t ning qiymati t=Vo / g formula bilan xisoblansin;

3) h ning qiymati $h=Vo\ t - g\ t^2 / 2$ (1) formula bilan xisoblansin;

4) t va h o’zgaruvchilarning sonli qiymatlari ekranga yoki qog’ozga chiqarilsin;

5) xisoblash to’xtatilsin.

Masalaning qo’yilishida koptok 29, 5 m /sek bilan tepilsa, degan shart bor edi. ya`ni, Vo=29, 5 va g =9, 81 bo’lsa, t va h qancha bo’ladi? (Talabalarning o’ziga echish taklif etiladi: t=3 sek, h=43, 35 m.) Natija xammada bir xil chiqadi.

Ikkinchi misolning echish algoritmi quyidagicha bo’ladi:

1) kompyuter xotirasiga a va b ning qiymati kiritilsin;

2) to’g’ri to’rtburchaklar soni n kiritilsin;

3) to’rtburchaklar asosi (eni) xisoblansin: h= (b-a)/n

4) 1-to’rtburchak balandligi (bo’yi) aniqlansin: x1qa

5) 1-to’rtburchak yuzi xisoblansin: S1=sqr (x1)*h

6) S1 ning qiymati eslab qolinsin;

7) 2-to’rtburchakka o’tilsin; x2= x1+h (balandligi shunga bog’liq)

8) 2-to’rtburchak yuzi xisoblansin: S2=sqr (x2)*h

9) S2 ning qiymati S1 ning qiymatiga qo’shib qo’yilsin va yig’indi eslab qolinsin;

10) n-to’rtburchakka o’tilsin: xN = x (N-1)+h=b

11) n-to’rtburchak yuzi xisoblansin: Sn=sqr (b)*h

12) Sn ning qiymati S1, S2,. .., S (N-1) lar qiymatiga qo’shilsin;

Algoritmni ishlab chiqish uchun avvalo masalaning echish yo’lini yaxshi tasavvur qilib olish, keyin esa uni formallashtirish, yani aniq qoidalar ketma-ketligi ko’rinishida yozish kerak.

Algoritmni ishlab chiqishda masalani echish jarayonini shunday formallashtirish kerakki, bu jarayon etarli darajadagi oddiy qoidalarning chekli ketma-ketligi ko’rinishiga keltirilsin.

Masalan, biz ko’pincha ko’p xonali sonlar ustida asosiy arifmetik amallarni bajarishda vatandoshimiz Al-Xorazmiyning IX asrda yaratgan qoidalarini ishlatamiz. "Algoritm" atamasi ham ana shu buyuk matematik nomidan kelib chiqqan.

II. ALGORITMNING ASOSIY XOSSALARI, IFODALASH USULLARI, TURLARI

II.1. Algoritmning asosiy xossalari.

Algoritm quyidagi asosiy xossalarga ega: uzluklilik, aniqlik, natijaviylik va ommaviylik.

UZLUKLILIK. Dastlabki berilgan malumotlarni natijaga aylantirish jarayoni uzlukli ravishda amalga oshiriladiki, bunda vaqtning xar bir keyingi keladigan daqiqasidagi miqdor (kattalik)larning qiymati vaqtning shundan oldingi daqiqasida bo'lgan miqdorlar qiymatidan ma`lum bir qoidalar bo'yicha olinadi.

ANIQLIK. Algoritmning xar bir qoidasi aniq va bir qiymatli bo'lishi zarurki, bunda vaqtning biror daqiqasida olingan miqdorlar qiymati vaqtning shundan oldingi daqiqasida olingan miqdorlar qiymati bilan bir qiymatli aniqlangan bo'ladi.

NATIJAVIYLIK. Algoritm masalaning echimiga chekli sondagi qadamlar ichida olib kelishi yoki masalani "echib bo'lmaydi" degan xabar bilan tugashi kerak.

OMMAVIYLIK. Masalaning echish algoritmi shunday yaratilishi kerakki, uni faqat boshlang'ich malumotlar bilan farqlanadigan masalalarni echish uchun xam qo'llanilishi kerak.

Bunda boshlang'ich malumotlar "algoritmni qo'llash soxasi" deb ataladigan birorta soxadan olinadi.

Masalan, yuqoridagi 1 - misolda koptok o'rniga boshqa narsani tik irg'itilsa va uning boshlang'ich tezligi malum bo'lsa, shu algoritm bilan u erishadagan balandlik aniqlanadi.

II. 2. Algoritmni ifodalash usullari va ularga misollar

Algoritmni ishlab chiqishda uni bir necha xil usul bilan ifodalab bersa bo'ladi. Shulardan uchtasi keng tarqalgan. Bular:

1. Algoritmni oddiy tilda ifodalash;
2. Algoritmni tuzim ko'rinishida ifodalash;
3. Algoritmni maxsus (algoritmik) tilda yozish.

II.2.1 Algoritmni oddiy tilda ifodalash

Algoritmlarni ifodalashning eng keng tarqalgan shakli - oddiy tilda so'zlar bilan bayon qilishdir. Bu nafaqat hisoblash algoritmlarida, balki hayotiy, turmushdagi "algoritm"larga ham tegishlidir.

Masalan, biror bir taom yoki qandolat mahsulotini tayyorlashning retsepti ham oddiy tilda tavsiflangan algoritmdir. Shaharlararo telefon - avtomat orqali aloqa o'rnatishning o'ziga xos algoritmidan foydalanasiz. Do'kondan yangi kir yuvish mashinasi yoki magnitofon sotib olinsa, ishni foydalanishning algoritmi bilan tanishishdan boshlaymiz.

Masalani kompyuterda echishda ham, ko'pincha matematika tilini ham o'z ichiga olgan tabiiy tildan foydalanish mumkin. Algoritmning bunday tildagi yozuvi

izlanayotgan natijaga olib keladigan amallar ketma-ketligi ko’rinishida bo’lib, odam tomonidan bir ma'noli idrok etilishi kerak. So’zlar bilan ifodalangan har bir amal “algoritmning qadami” deb ataladi. Qadamlar tartib nomeriga ega bo’ladi.

Algoritm ketma-ket, qadam-ba qadam bajarilishi kerak. Agar algoritm matnida "N sonli qadamga o’tilsin" deb yozilgan bo’lsa, bu algoritmning bajarilishi ko’rsatilgan N-qadamdan davom etishini bildiradi.

Ko’rinib turibdiki, yuqoridagi uchchala misol algoritmi ham oddiy tilda yozilgan ekan.

Algoritmlarni oddiy tilda ifodalash kompyuterga kiritish uchun yaramaydi. Buning uchun algoritmni kompyuter tilida shunday bayon qilish kerakki, masalan kompyuterda echish jarayonida bu algoritm ishni avtomatik boshqqarib turadigan bo’lsin.

Kompyuter tushunadigan shaklda yozilgan algoritm masalani echish dasturidir.

Algoritmni oddiy tilda yozishda to’rt xil amaldan: hisoblash, N- qadamga o’tish, shartni tekshirish, hisoblashning oxiri, shuningdek kiritish va chiqarish amallaridan foydalanilgan maqul. Bular ichida eng ko’p foydalaniladigani hisoblash amalidir.

II.2.2 Algoritmni tuzim ko’rinishida ifodalash.

Nisbatan murakkab masalalarni echishda algoritmdan muayyan kompyuter tilidagi dasturga o’tish juda qiyin. Bunday bevosita o’tishda algoritmning alohida qismlari orasidagi bog’lanish yo’qoladi, algoritm tarkibining asosiy va muhim bo’lmagan qismlarini farqlash qiyin bo’lib qoladi.

Bunday sharoitda keyinchalik aniqlash va to’g’rilash ancha vaqt talab qiladigan xatolarga osongina yo’l qo’yish mumkin.

Odatda algoritm bir necha marta ishlab chiqiladi, ba'zan xatolarni to’g’rilash, algoritm tarkibini aniqlashtirish va tekshirish uchun bir necha marta orqaga qaytishga to’g’ri keladi.

Algoritm ishlab chiqishning birinchi bosqichida algoritmni yozishning eng qulay usuli - algoritmni tuzim ko’rinishda ifodalashdir.

Algoritm tuzimi - berilgan algoritmni amalga oshirishdagi amallar ketma-ketligining oddiy tildagi tasvirlash elementlari bilan to’ldirilgan grafik tasviridir. Algoritmning har bir qadami tuzimda biror bir geometrik shakl - blok (blok simvoli) bilan aks ettiriladi.

Bunda bajariladigan amallar turiga ko’ra turlicha bo’lgan bloklarga GOST bo’yicha tasvirlanadigan turli xil geometrik shakllar - to’g’ri to’rtburchak, romb, parallelogramm, ellips, oval va hokazolar mos keladi.

Algoritm tuzimlarini qurish qoidalari GOST 19. 002-80 da (xalqaro standart ISO 2636-73 ga mos keladi) qat'iy belgilab berilgan. GOST 19. 003 -80 (ISO 1028-73ga mos) algoritm va dasturlar tuzimlarida qo’llaniladigan simvollar ro’yxatini, bu simvollarning shakli va o’lchamlarini, shuningdek ular bilan tasvirlanadigan funktsiyalarni (amallarni) belgilaydi.

Tuzim blok(simvol)lari ichida hisoblashlarning tegishli bosqichlari ko’rsatiladi. Shu erda har bir simvol batafsil tushuntiriladi.

Har bir simvol (blok) o’z raqamiga ega bo’ladi. U tepadagi chap burchakka chiziqni uzib yozib qo’yiladi. Tuzimdagi grafik simvollar hisoblash jarayonining rivojlanish yo’nalishini ko’rsatuvchi chiziqlar bilan birlashtiriladi.

Ba'zan chiziqlar oldida ushbu yo’nalish qanday sharoitda tanlanganligi yozib qo’yiladi. Axborot oqimining asosiy yo’nalishi tepadan pastga va chapdan o’ngga ketadi. Bu hollarda chiziqlarni ko’rsatmasa ham bo’ladi, boshqa hollarda albatta chiziqlarni qo’llash majburiydir. Blokka nisbatan oqim chizig’i (potok linii) kiruvchi yoki chiquvchi bo’lishi mumkin. Blok uchun kiruvchi chiziqlar soni chegaralanmagan.

Chiquvchi chiziq esa mantiqiy bloklardan boshqa hollarda faqat bitta bo’ladi. Mantiqiy bloklar ikki va o’ndan ortik oqim chizig’iga ega bo’ladi.

Ulardan har biri mantiqiy shart tekshirishining mumkin bo’lgan natijalarga mos keladi.

O’zaro kesiladigan chiziqlar soni ko’p bo’lganda, chiziqlar soni haddan tashqari ko’p bo’lsa va yo’nalishlari ko’p o’zgaraversa tuzimdagi ko’rgazmalik yo’qoladi. Bunday hollarda axborot oqimi chizig’i uzishga yo’l qo’yiladi, uzilgan chiziq uchlariga "birlashtiruvchi" belgisi qo’yiladi.

Agar uzilish bitta sahifa ichida bo’lsa, O belgisi ishlatilib, ichiga ikki tarafga ham bir xil harf-raqam belgisi qo’yiladi.

Agar tuzim bir necha sahifaga joylansa, bir sahifadan boshqasiga o’tish "sahifalararo bog’lanish" belgisi ishlatiladi.

Bunda axborot uzatilayotgan blokli sahifaga qaysi sahifa va blokka borishi yoziladi, qabul qilinayotgan sahifada esa qaysi sahifa va blokdan kelishi yoziladi.

Algoritm tuzimlarini qurishda quyidagi qoidalarga rioya qilish kerak.

Parallel chiziqlar orasidagi masofa 3 mm dan kam bo’lmasligi, boshqa simvollar orasidagi masofa 5 mmdan kam bo’lmasligi kerak. Bloklarda quyidagi o’lchamlar qabul qilingan: a=10, 15, 20; b=1, 5*a.

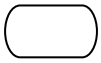
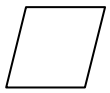
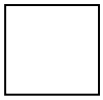
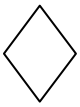
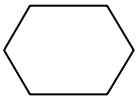
Agar tuzim kattalashtiriladigan bo’lsa, a ni 5 ga karrali qilib oshiriladi. Bu talablar asosan 10-bosqichda, dasturga yo’riqnoma yozishda rioya qilinadi. Algoritmlarni mayda-mayda bo’laklarga ajratishda hech qanday chegaralanishlar qo’yilmagan, bu dastur tuzuvchining o’ziga bog’lik.

Lekin, juda ham umumiy tuzilgan tuzim kam axborot berib, noqulaylik tug’dirsa, juda ham maydalashtirib yuborilgani ko’rgazmalilikka putur etkazadi.

Shuning uchun murakkab va katta algoritmlarda har xil darajadagi bir nechta tuzim ishlab chiqiladi.

Algoritmning tuzim tarzidagi ifodasining yana bir afzalligi undan uchinchi ko`rinish, ya'ni algoritmik tildagi ifodasi (dastur)ga o`tish ham juda oson bo`ladi. Chunki bunda har bir blok algoritmik tilning ma'lum bir operatori bilan almashtiriladi xolos.

Quyida asosiy bloklar uchun foydalaniladigan shakllar keltirilgan:

ShAKL	Qaysi xolda ishlatiladi	ShAKL	Qaysi xolda ishlatiladi
	Boshlanish va oxirida		Axborotni kiritish va chiqarish
	Xisoblashlar uchun		Natijani chop etish uchun
	Tarmoqlanish shartini tekshirishda		sikl boshlanishida

II. 2.3 Algoritmni maxsus tilda ifodalash.

Bu usulda algoritmni ifodalash uchun “dasturlash tillari” deb ataluvchi suniy tillar qo’llaniladi. Buning uchun ishlab chiqilgan algoritm shu tillar yordamida bir manoli va kompyuter tushuna oladigan ko’rinishda tavsiflanishi zarur.

Uning tarkibida cheklangan sondagi sintaksis konstruktsiyalar to’plami bor bo’lib, u bilan algoritm yaratuvchi tanish bo’lishi kerak. Ana shu konstruktsiyalardan foydalanib buyruq va ko’rsatmalar formal ifodalarga o’tkaziladi.

Zamonaviy dasturlash tillari kompyuterning ichki kompyuter tilidan keskin farq qiladi va kompyuter bevosita ana shu tilda ishlay olmaydi. Buning uchun dasturlash tilidan mashina tushunadigan tilga tarjima qiluvchi maxsus dastur - translyatordan foydalaniladi.

Dasturni translyatsiya qilish va bajarish jarayonlari vaqtlarga ajraladi.

Avval barcha dastur translyatsiya qilinib, so’ngra bajarish uslubida ishlaydigan translyatorlar “kompilyatorlar” deb ataladi. Dastlabki tilning har bir operatorini o’zgartirish va bajarishni ketma-ket amalga oshiriladigan translyatorlar “interpretatorlar" deb ataladi.

Dasturlashning ixtiyoriy tili belgilar majmuini va algoritmlarni yozish uchun ushbu belgilarni qo’llash qoidalarini o’z ichiga oladi.

Dasturlash tillari bir biridan alifbosi, sintaksisi va semantikasi bilan ajralib turadi.

Alifbo - tilda qo’llaniladigan ko’plab turli ramziy belgilar (harflar, raqamlar, maxsus belgilar)dir.

Tilning sintaksisi jumlalar tuzishda belgilarning bog’lanish qoidalarini belgilaydi, semantikasi esa ushbu jumlalarning mazmuniy izohini belgilaydi.

II. 3. Dasturlash tillari va ularni tasniflash.

Hozirgi kunda dasturlash tillarini u yoki bu belgisi bo’yicha tasniflash mumkin. Dasturlash tilining kompyuterga bog’liqlik darajasi bo’yicha tasniflash eng umumiy hisoblanadi .

Yuqorida aytilgan belgiga qarab, dasturlash tillari kompyutera bog’liq va kompyuterga bog’liq bo’lmagan tillarga bo’linadi.

Kompyuterga bog’lik tillar, o’z navbatida, kompyuter tillari va kompyuterga mo’ljallangan tillarga ajratiladi.

Dasturlash tilining kompyuter tiliga yaqinligi darajasini tariflash uchun til darajasi tushunchasi qo’llaniladi.

Kompyuter tili 0 daraja deb qabul qilingan bo’lib, sanoq boshi hisoblanadi. Odamning tabiiy tili “eng yuqori darajadagi til” deb qaraladi.

Kompyuterga bog’liq bo’lmagan tillar ham ikkita turga bo’linadi: birinchisi protseduraga mo’ljallangan tillar, ikkinchisiga - muammoga mo’ljallangan tillar.

Protseduraga mo’ljallangan tillar turli masalalarni echish algoritmlarini (protseduralarni) tavsiflashga mo’ljallangan; shuning uchun ular ko’pincha oddiy qilib “algoritmik tillar" deb ataladi.

Ushbu tillar echilayotgan masalalar xususiyatlarini to’la hisobga oladi va kompyuterning turiga deyarli bog’liq emas. Bu xildagi tillar tarkibi kompyuter tiliga qaraganda tabiiy tilga, masalan, ingliz tiliga yaqinroq.

Hozirgi kunda hisoblash, muhandis-texnik, iqtisodiy, matnli va sonli axborotlarni taxlil qilish va boshqa masalalarni echish tillari malum.

Masalan: FORTRAN tili 1954 yili ishlab chiqilgan bo’lib, FORmula TRANslator - formulalar translyatori degan manoni anglatadi va ilmiy va muhandis - texnik masalalarni hisoblashlarda qo’llaniladi.

ALGOL tili 1960 yili yaratilgan bo’lib, ALGORITMIC Langauge -algoritmik til degan ma'noni anglatadi va ilmiy-texnik masalalarni hisoblashlarda qo’llaniladi.

KOBOL tili 1959 yili yaratilgan bo’lib, Common Businees Oriented Langauge - “savdo-sotiq masalalariga mo’ljallangan til” degan ma'noni anglatadi. Korxona va tarmoqning moddiy boyligini, moliyasini, ishlab chiqargan mahsulotini hisobga olish bilan bog’liq iqtisodiy masalalarni echish uchun qo’llaniladi.

PASKAL tili 1971 yilda e`lon qilingan bo’lib, frantsuz olimi Blez Paskal nomiga qo’yilgan. Turli xildagi masalalar echimini olishda tartiblangan (strukturaviy) dasturlar tuzishda qo’llaniladi.

PL/1 tili 1964 yilda yaratilgan bo’lib, Programming Langauge/ 1 - 1-tartib raqamli dasturlash tili ma'nosini anglatadi. Ushbu til universal tillar turkumiga kiradi.

Bu tilda ishlab chiqilgan dasturlar kompyuterni yangisi bilan almashtirilganda qaytadan tuzib chiqilishi zarur emas.

BEYSIK (BASIC - Beginner's All Purpose Sumbolic Instruction Code - boshlovchilar uchun ko’p maqsadli dasturlash tili) hisoblash algoritmlarini yozish uchun qo’llaniladigan algoritmik tildir. Bu til 1965 yilda Dartmut kolleji xodimlari Kemini va Kurtslar tomonidan ishlab chiqilgan.

Protseduraga mo’ljallangan tillardan masalalarning matematik ifodalari, algoritmlar va dasturlash usullari bilan tanish bo’lgan mutaxassislar foydalaniladilar.

Bunda ulardan kompyuterning tuzilishini mukammal bilish talab qilinmaydi.

Muammoga mo’ljallangan tillar kompyuterda masala echish usullari va dasturlash usullari bilan tanish bo’lmagan foydalanuvchilar uchun yaratilgandir.

Foydalanuvchi masalani tariflashi, boshlang’ich malumotlarni berishi va natijani chiqarishning talab qilingan ko’rinishini aytishi kifoya.

II. 4. Algoritmning asosiy turlari.

Masala echimining algoritmi ishlab chiqilayotgan davrda asosan uch xil turdagi algoritmlardan foydalanib, murakkab ko’rinishdagi algoritmlar yaratiladi.

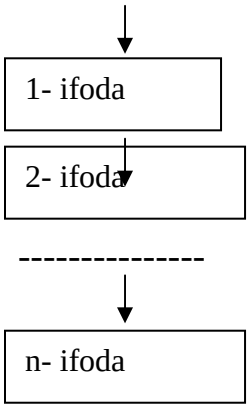
Algoritmning asosiy turlariga chizig’li (a), tarmoqlanadigan (b) va takrorlanadigan (c) ko’rinishlari kiradi.

Murakkab masalalarning echimini olish algoritmlari yuqoridagi turlarining barchasini o’z ichiga olishi mumkin.

Chiziqli turdagi algoritmlarda bloklar biri ketidan boshqasi joylashgan bo’lib, berilgan tartibda bajariladi. Bunday bajarilish tartibi “tabiiy tartib” deb ham yuritiladi.

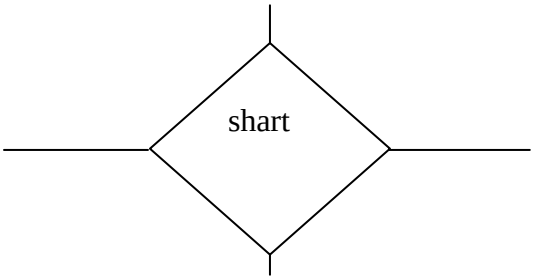
Yuqorida ko’rib o’tilgan birinchi misol chiziqli turdagi algoritmga misol bo’ladi. Amalda hamma masalalarni ham chiziqli turdagi algoritmga keltirib echib bo’lmaydi.

Chiziqli xisoblash jarayonining tuzimi quyidagicha ko`rinishda ifodalanadi.



Ko’p hollarda biron bir oraliq natijaga bog’liq ravishda hisoblashlar yoki u yoki boshqa ifodaga ko’ra amalga oshirilishi mumkin yani birorta mantiqiy shartni bajarilishiga bog’lik holda hisoblash jarayoni u yoki bu tarmoq bo’yicha amalga oshirilishi mumkin.Bunday tuzilishdagi hisoblash jarayonining algoritmi “tarmoqlanuvchi turdagi algoritm” deb ataladi.

Algoritmning bu konstruktsiyasi tuzimda yo`q va ha ko`rinishida ifodalanadi.

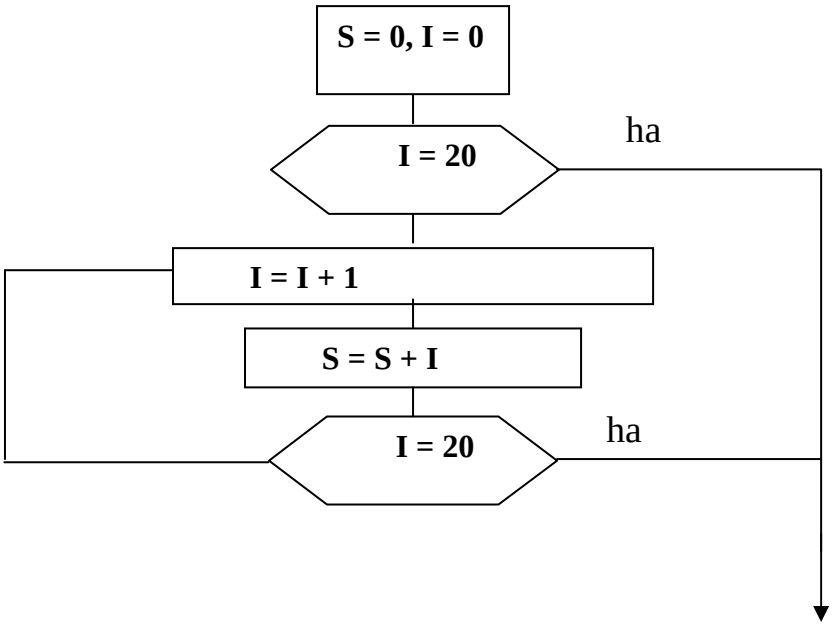


Ko’pgina hollarda masalalarning echimini olishda bitta matematik bog’lanishga ko’ra unga kiruvchi kattaliklarni turli qiymatlariga mos keladigan qiymatlarini ko’p martalab hisoblash to’g’ri keladi.

Hisoblash jarayonining bunday ko’p martalab takrorlanadigan qismi “takrorlanishlar” deb ataladi. Takrorlanishlarni o’z ichiga olgan algoritmlar “takrorlanuvchi turdagi algoritmlar” deb ataladi. Takrorlanuvchi turdagi algoritmni yozish va chizish o’lchamlarini sezilarli darajada qisqartirish takrorlanadigan qismlarni ixcham ifodalash imkonini beradi.

Yuqoridagi ikkinchi misol takrorlanuvchi turdagi algoritmlarga tegishlidir.

Quyida 1 dan to 20 gacha bo`lgan butun sonlar kvadratlari yig`indisini xisoblash algoritmini tuzim ko`rinishi keltirilgan.



III. HISOBLASHGA OID AMALIY MASALALARNING ALGORITMLARI

III. Amaliy masalalarning qo’yilishi

Bu bobda bir qancha amaliy masalalarning qo’yilishini keltiramiz. Maqsad bu masalalarning algoritmi va dasturini tuzishdan iborat.

1. ***«Кўндаланг диагонал»***. A(m,n) массивнинг, индекслари айирмаси берилган К сонига (К-манфий сон ҳам бўлиши мумкин) тенг бўлган (i-j=k), элементлари йиғиндисини топинг.
2. ***«Квадратчалар»***. Ҳар бир элементи 0,1,5 ёки 11 га тенг A(m, m) массив берилган. Ҳар бирида элементлари ҳар хил бўлган тўртликлар A(i,j), A(i+1,j), A(i,j+1), A(i+1,j+1) миқдорини топинг.
3. ***«Саноқ системалари»***. M(9) массивда қандайдир натурал соннинг I – саноқ системасида рақамлари ёзилган. (M(1)-бирлар хонаси ва ҳ.к.). Бирлар хонасидан бошлаб туриб, J-саноқ системасида бу сон рақамларини чоп этинг. I,J сонлар 10 дан ошмайди.
4. ***«Календар»***. Кун, ой, йилни билдирувчи учта a,b,c сонлар берилган. Шу куннинг йил бошидан ҳисоблангандаги n-тартиб рақамини топинг.

Кўрсатма: тартиб рақами 400 га бўлинадиган, шунингдек тартиб рақами 4 га бўлиниб, 100 га бўлинмайдиган йиллар кабиса йили ҳисобланади.

5. ***«Эгар нуқта»***. A(m,n) сонли массив берилган. Ўз сатрида энг кичик бўлган элемент ўз устунида энг катта бўлса, у эгар нуқта дейилади. Агар массивда эгар нуқта бўлса, у ётган сатр ва устун рақамини, агар ундай нуқта бўлмаса, нол сонини чоп этинг.
6. ***«Касрни қисқартириш»***. Натурал «м» ва «n» сонлар берилган. Умумий бўлувчига эга бўлмаган шундай натурал м1 ва n1 сонларни топиш керакки, $м_1/n_1$ = м/n бўлсин.
7. ***«Массивларнинг қўшилиши»***. «M» ва «N» сонлар ва иккита тартибга солинган массивлар: $a_1 \le a_2 \le ..... \le a_м$ ҳамда $в_1 \le в_2 \le ..... \le в_n$ берилган. Бу элементлардан тартибга солинган учинчи массивни ҳосил қилинг: $c_1 \le c_2 \le ..... \le c_{m+n}$ Кўрсатма: «M» ва «N» лар катта сонлар бўлганда дастурдаги амаллар миқдорига эътибор беринг.
8. ***«Мода»***. Бутун сонли A(n) массивда энг кўп учрайдиган сонни топинг. Агар бундай сонлар бир нечта бўлса, улардан биттасини аниқланг.

9. ***«Марказий қишлоқ»***. «К» та қишлоқ мавжуд. Агар *i*-қишлоқда тез ёрдам пункти жойлаштирилса, чақирув бўйича j-қишлоққа бориш $a_{ii}+a_{ij}$ ($1\le i,j\le k$, $i\ne j$) вақтни олади.. Шундай *i*-қишлоқ тартиб рақамини топингки, ундан энг узоқ қишлоққа бориш учун кам вақт сарфлансин. А(к,к) масссив берилган. Унда ҳамма a_{ij} элементлар нолдан катта ва a_{ij} - элемент a_{ji} элементга тенг бўлмаслиги мумкин.

10. ***«Тартиб индекслари»***. А(n) сонли массив берилган . 1, 2,...,n сонларнинг шундай, $i_1,i_2,...,i_n$ ўрин алмаштиришни топингки, натижада $a_{i1} \le a_{i2} \le ... \le a_{in}$ бўлсин.

11. ***«Ноллаштириш»***. Берилган икки ўлчовли А(м,n) массивда ноли бўлган сатр ва устун элементларини ноллар билан алмаштиринг.

Шарт. Ёрдамчи бир ўлчовли массивдан фойдаланиш мумкин, лекин ёрдамчи икки ўлчовли массив ишлатиш мумкин эмас .

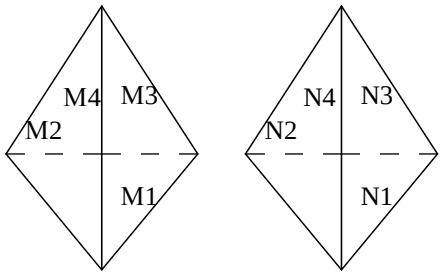
12. ***«Улгуржи харид»***. Пайпоқ жуфти 105 сўм, боғлами (12 жуфт) 1025 сўм, қутиси (12 боғлам) 11400 сўм туради. Харидор сотиб олмоқчи бўлган пайпоқларнинг n жуфт сонига кўра, харидор сотиб олиши керак бўлган n1,n2,n3 қути, боғлам, пайпоқлар жуфтини ҳисоблаб берувчи дастур тузинг. Тушунтириш. 11 жуфт пайпоқ ўрнига бир боғламни харид қилиш арзонга тушади.

13. ***«Тўнтарилган сонлар»***. А(N) сонли массив берилган. Массивнинг максимал узунликдаги кесмасини топиш керак. Унда биринчи сон охиргисига, иккинчи сон охиргисидан битта олдингисига ва ҳ.к. тенг бўлсин . Бу кесма узунлигини чоп этинг.

14. ***«Икки марта монотон»***. Сонларнинг А(м,n) массиви сатрлар ва устунлар бўйича камайиб бориш тарзида тартибланган, яъни ҳамма i=1,...,m лар учун, $a_{i1} \le a_{i2} \le ....$ ҳамма j=1,..., n лар учун, $a_{1j} \le a_{2j} \le ...$. Массив элементлари ичидан берилган «х» сонига тенг бўлганини топинг. Агар бундай элемент бўлмаса, «Йўк » деб чоп этинг. Мажбурий шарт. Ечимда амаллар сони m*n атрофида эмас, m+n атрофида бўлсин.

15. ***«Тетраэдрлар».*** Иккита тенг тўғри М ва N тетраэдрлар қиррасига М1,М2,М3,М4 ва N1,N2,N3,N4 сонлар 3-расмда кўрсатилган тартибда ёзилган. Тетраэдрларни бир хил сонлар ёзилган қирралари билан бир-

бирига мос тушириб, жойлаштириш мумкинми? «Ҳа» ёки «Йўқ» жавоб берилсин (1-расм).



1-расм

16. ***«Каср даври»***. «M» ва «N» натурал сонлар берилган. Ўнлик M/N каср даврини чоп этинг. Масалан, 1/7 каср учун давр 142857 га тенг, каср чекли бўлса, унинг даври битта 0 рақамига тенг бўлади.

III. Qo’yilgan amaliy masalalarning algoritmi va dasturi

Бу бўлимда дастурдан аввал масала ечимининг алгоритмлари келтирилади. Бу баъзан, фақат алгоритмик тилда ёзилиши қолган, тугалланган фикр, баъзи ҳолларда эса ечим мулоҳазаси, бошқа ҳолларда эса дастурни тушунтирувчи изоҳ бўлади. Лекин ҳамма ҳолларда ҳам дастурни ўқиш енгиллаштирилган, ўзлаштириш керак бўлган дастур усуллари баён этилади. Амаллар аниқлиги, одатда, доимий кўпайтувчи аниқлигида кўрсатилган.

1 – масала

Алгоритм. «Кўндаланг диагонал»

Иккита сон аниқлаймиз: P=max(1,1-k) ва g=min(n,m-k). Шунда изланаётган миқдор, ҳамма j=p,p+1,...,g лар бўйича A[k+j,j], элементлар йиғиндисига тенг бўлади (агар j нинг бундай қийматлари бўлмаса, яъни p>g бўлса, у нолга тенг). A массивнинг ҳамма m∗n элементлари қараладиган ечимни қониқарли эмас, деб қараши керак.

Дастур

```
Program Ko’ndalang_diagonal;
const NN=10;
```

```
        MM=10;
var
        m,n, j,k,p,g,s :integer;
        A : array [1..MM,1..NN] of integer;
begin
        writeln (‘M,N,K:=’);
        readln (m,n,k);
        for p:=1 to m do
                for j:=1 to n do
                begin
                        writeln (‘A [‘,P,’,’,j,’]:=’);
                        readln (A[p,j]);
                end;
                if k>0 then p:=1 else p:=1-k;
                   if k+n<m then g:=n
                                else
                                        g:=m-k;
                                        s:=0;
                for j:=p to g do s:=s+A[k+j,j];
                writeln(s);
end.
```

2 – масала

Алгоритм. «Квадратчалар».

Квадратчанинг бурчакларидаги ҳамма сонлар ҳар хил бўлса, уларнинг йиғиндиси 0+1+5+1=17. Лекин бунинг акси ҳам тўғри эканлигини таъкидлаш мумкин. Бу дастурни соддалаштиради.

Дастур

```
Program Kvadratchalar;
const MM=50;
var
        i,j,m,S :integer;
        A : array [1..MM,1..MM] of integer;
```

```
begin
    writeln (‘M:=’);
    readln (m);
    for i:=1 to m do
        for j:=1 to m do
        begin
            writeln (‘A [‘,i,’,’,j,’]:=’);
            readln (A[i,j]);
        end;
        S:=0;
    for i:=1 to m-1 do
        for j:=1 to m-1 do
            if (A[i,j]+ A[i,j+1]+ A[i+1,j]+ A[i+1,j+1]=R ) then S:=S+1;
        writeln (S);
end.
```

3 – масала

Алгоритм. «Саноқ системалари»

Берилган сон миқдорини Горнер схемаси бўйича топиш керак:

N=((...(M[9]∗i+M[8])∗i+...+M[2])∗i+M[1]

N, бутун сонлар учун, мумкин бўлган максимал миқдордан ошмайди, деб фараз қилинади. N сонлар разрядлари (хоналари) j-системада N ни j га бутун бўлишдаги қолдиқлар сифатида ҳосил қилинади:

k=N/j : M=N−K∗j : N=K ва ҳоказо.

M нинг қолдиқларини масала шартига кўра, эслаб қолиш ва чоп этиш шарт эмас.

Дастур

```
Program Sanoq_sistemalar;
var
    i, j,k : integer;
    n : real;
    m : array [1..9] of integer;
begin
    writeln (‘i,j:=’);
```

```
        readln (i,j);
        for k:=9 downto 1 do
        begin
              write ('M[',k,']:=');
              readln (m[k]);
        end;
        writeln;
        n:=0
        for k:=9 downto 1 do n :=n*i+M[k];
        writeln (trunc(n));
        repeat
        write (trunc (n), mod j,' ');
        n:= trunc(n) div j;
        until n=0;
        writeln
end.
```

4 – масала

Алгоритм «Календар» .

Кабиса бўлмаган йил ойларида кунларнинг сон бўйича M[1:11] жадвалини ҳосил қилиш қулай. «m» ни «n»га бўлишдаги қолдиқни аниқлаб берувчи MOD(m,n) ёки унга ўхшаш функцияни хам ишлатиш мумкин.

Дастур

```
Program Kalendar;
var
      a,b,c,i,j : integer;
      M : array [1..11] of integer;
      function D (x:integer) : boolean;
      begin
            D:=(c mod x)=0;
       end;
begin
      writeln ('a,b,c:=');
      readln (a,b,c);
      writeln;
```

```
    for i:=1 to 11 do
    case i of
        1,3,5,7,8,10:   M[i]:=31;
        4,6,9,11:       M[i]:=30;
        2:              M[i]:=28;
    end;
    j:=a;
    for i:=1 to b-1 do j:=j+m[i];
    if (b>2) and (D(4) and not D(100) or D(400)) then j:=j+1;
    writeln (j);
end.
```

5 – масала

Алгоритм «Эгар нуқта»

Агар mi i-сатрдаги a_{ij} элементларнинг энг кичик қиймати, Mj эса j устундаги a_{ij} элементларнинг энг катта қиймати бўлса,

$mi \le a_{ij} \le M_j$, бўлади.

Бу бир нечта мулоҳазалар қилишга имкон беради:

а) ихтиёрий сатрнинг минимуми ихтиёрий устун максимумидан катта эмас, яъни ҳамма вақт mi≤Mj шарт ўринли;

б)агар қандайдир i ва j учун mi=Mj тенглик бажарилса, минимумларнинг энг каттаси максимумларнинг энг кичиги билан мос келади, i сатрнинг j устун билан кесишишида эса эгар нуқта:

$mi=a_{ij}=Mj$ ётади; (*)

в)агар a_{ij}-эгар нуқта бўлса, унинг учун (*) шарт бажарилади ва, демак, минимумларнинг максимуми, максимумларнинг минимумига тенг бўлади.

Айтганларни дастурга жорий этиш керак. Ҳар бир сатрда минимум mi изланади, бу минимумлар ичидан максимум - Ма танланади ва у жойлашган i_o сатр эслаб қолинади. Агар минимум «Ма» нинг жорий қийматидан кичик бўлиб қолса, сатрда минимумни излашни тўхтатиш керак.

Кейин, максимал элемент топилган «Ма» қийматга тенг устун изланади, агар бундай устун бўлмаса, эгар нуқта ҳам йўқ. Амалда дастурда, агар устунда Ма элементдан катта элемент бўлмаса а) пунктга кўра, устун, изланаётган устун эканлиги кўрсатилади.

Агар дастурда бу ерда келтирилган мулоҳазалардан фойдаланилмаса, у анча секин ишлайдиган бўлади.

Дастур

```
Program Egar_nuqta
const nn=20; mm=20;
label 1,2;
var
      i,n,j,m,io,mi,ma :integer;
      a : array [1..mm,1..nn] of integer;
begin
      writeln (‘m,n=’);
      readln (m,n);
      for i:=1 to m do
        begin
            writeln (‘a[‘,i,’,’,j,’]=’);
            readln (a[i,j]);
        end;
        for i:=1 to m do
                  begin
                     if (i>1) and (a[i,j] <=ma) then goto 1;
                     if (j=1) or (a[i,j] <mi) then mi:=a[i,j];
                1 : end;
                     ma:=mi; io:=i;
                     for i:=1 to n do
                         begin
                                for i=1 to m do if a[i,j]> ma goto 2;
                                writeln (io:3,j:3); exit;
                      2: end;
                   writeln(0);
end.
```

6 – масала

Алгоритм «Касрни қисқартириш».

Евклид алгоритмини дастурлаш мумкин. Унга кўра m ва n сонларнинг энг катта умумий бўлувчисини топиш ва унга қисқартириш керак.

Евклид алгоритми моҳиятини тушунтирамиз, m1≥m2 бўлсин, (m1, m2) жуфтликнинг ихтиёрий умумий бўлувчиси (m2, m1-m2) жуфтликнинг, демак, (m2,m3) жуфтликнинг ҳам умумий бўлувчиси бўлади, бу ерда

m3 =m1-(m1 /m2) ∗ m2

m1ни m2га бўлишдаги қолдиқ бўлади, шунинг учун m3<m2 олдиндан маълум. Акси ҳам тўғри: (m2,m3) жуфтликнинг ҳар қандай умумий бўлувчиси (у.б.) (m1, m2) жуфтликнинг хам умумий булувчиси бўлади. Шунинг учун

у.б. (m1,m2) =у.б.(m2,m3)

у.б. (m1,m2) функцияда аргументни уни кичигига бўлгандаги қолдиғи билан кетма-кет алмаштира бориб,

у.б. (m1,m2) =у.б.(m2,m3)=... =у.б. (mк,0) =mк

кетма-кетлик ҳосил қилинади, унда m1≥m2>...> mk>0 ва mк бошланғич m1, m2 сонларнинг энг катта умумий бўлувчиси бўлади.

Агар n катта сон бўлмаса, j-энг кичик қийматли i/j =m /n касрни танлаб олиш мумкин. У, албатта, қисқармайди. Фақат касрлар тенглигини нисбатлар тенглиги билан эмас, кўпайтмалар тенглиги билан текшириш керак:

i ∗n =j ∗m

Бу таклиф этилган дастурга олиб келади.

Дастур

```
Program Kasrni_qisqartirish;
label 1;
var
    i,n,j,m :integer;
begin
    readln (m,n);
    writeln (‘m=’,m,’n=’,n);
    for j:=1 to n do
      begin
        i:=j*m div n;
        if i*n= j*m then goto 1;
```

```
    end;
  1: writeln ('m/n',i,'/',j);
end.
```

7 – масала

Алгоритм «Массивларнинг қўшилиши»

Бу муҳим масала m+n амалда бажарилиши керак. A ва B дан биринчи элементларини оламиз, улардан кичигини C массивга ёзамиз ва уни ўз массивидан навбатдагиси билан алмаштирамиз. Яна иккитадан кичигини танлаб, кейин C га киритамиз ва ҳоказо, ҳар бир таққослашдан кейин C га элемент қўшилади, демак, таққослаш m+n дан кичик бўлади. Фақат дастурнинг, массивлардан биттаси томом бўлганидан кейин ҳам, тўғри ишлашини таъминлаш керак.

Дастур

```
Program Massivlar_qo'shilishi;
const MM=100;
      NN=100;
      MN=200;
var
      m,n,i,j,k :integer;
      A : array [1..MM] of integer;
      B : array [1..NN] of integer;
      C : array [1..MN] of integer;
begin
      writeln ('M,N:=');
      readln (m,n);
      writeln ('array A:=');
      for i:=1 to m do readln(A[i]);
      writeln ('array B:=');
      for j:=1 to n do readln(B[j]);
      i:=1; j:=1;
      for k:=1 to m+n do
                        begin
```

```
            if ((i>m) or (A[i]>B[j])) and not (j>n) then
              begin
                c[k]:=B[j]; j:=j+1;
              end;
                                                  else
              begin
                c[k]:=A[i]; i:=i+1;
              end;
        end;
        writeln (‘array C:=’);
        for k:=1 to m+n do writeln (c[k]);
end.
```

8 – масала

Алгоритм «Мода»

Бошланғич массивни тартибга келтириш, шундан кейин бир қарашда қийматлар частотасини – энг кўп учрайдиган қийматни ҳисоблаш керак. Частотани ёзиш учун массив ташкил этишга ҳожат йўқ-тез учрайдиганлар номзодини эслаб қолиш ва унинг частотасини қаралган элементнинг топилган частотаси билан таққослаш етарли.

Агар қандайдир ҳолга кўра, тартиблаштиришни бажармаслик керак бўлса, қуйидагича йўл тутилади: навбатдаги a[i] (бошида бу a[1] элемент) ундан кейин келадиган элемент билан таққосланади. Бунда, биринчидан, a[i] қийматнинг пайдо бўлиш частотаси ҳисобланади, иккинчидан, a[i] га тенг бўлган a[j] элементлари кетма-кет a[i+1], a[i+2]... элементлари билан алмаштирилади. А массив охиригача қараб чиқилиб, a[i] элемент частотаси аниқланганидан кейин a[i+к] элемент частотаси топилади, a[i] ва a[i+к] ўртасидаги элементлар ўтказиб юборилади. Бу алгоритм самарали кўринади ва агар массивда ҳар хил элементлар бўлса, n∗n та амал бажарилиш керак бўлади. Бу фикрлар таклиф этилган дастурда амалга оширилган.

Дастур

```
Program Moda;
const NN=100;
var
```

```
        n,m,am,i, j,k : integer;
        A : array [1..NN] of integer;
begin
        writeln (‘n:=’);
        readln (n);
        for i:=1 to n do readln(a[i]);
        writeln;
        m:=0; i:=1;
        while i+m<=n do
        begin
              k:=1;
              for j:=i+1 to n do
                 if a[j]=a[i] then
                      begin
                         a[j]:=a[i+k]; k:=k+1;
                      end;
              if m<k then
                      begin
                         am:=a[i]; m:=k ;
                      end;
              i:=i+k;
        end;
        writeln (am);
end.
```

9 – масала

Алгоритм «Марказий қишлоқ»

Бу масалани қуйидагича баён этамиз: А массивнинг ҳар бир і сатрида a[i,j] (j≠i) сонлари ичидан энг каттаси танланади ва у a[i,j] га қўшилади. Тегишли йиғиндини энг кичик қилувчи і топилиши керак. Масаланинг бундай қўйилишида унинг стандарт ечимга эгалигига йўл қўйилади.

Дастур

```
Program Markaziy_qishloq;
const KK=20;
```

```
var
        i,j,k,il,s,t :integer;
        A : array [1..KK,1..KK] of real;
begin
        writeln (‘k:=’);
        readln (k);
        for i:=1 to k do
                for j:=1 to k do
                begin
                    writeln(‘a[‘,i,’,’,j,’]=’);
                    readln(a[i,j]);
                end;
                for i:=1 to k do
                   begin
                      s:=0;
                      for j:=1 to k do
                          if (j<>) and (s<a[i,j]) then s:=a[i,j];
                      s:=s+a[i,j];
                      if (i=1) or (s<t) then
                                              begin
                                                 il:=i;
                                                 t:=s;
                                              end;
                   end;
        writeln (il);
end.
```

10 – масала

Алгоритм «Тартиб индекслари»

Ҳаммасидан осони «изланаётган тартиб индекслари» учун қўшимча I[1:N] массив ҳосил қилиш, аввал уни 1,2,...,N сонлари билан тўлдириш, кейин берилган А массивни, қандайдир усул билан тартибга келтириш, худди шундай тартибда I массив элементларининг ўрнини алмаштириш керак. А тартибга келтирилган вақтда, I изланаётган массив бўлади.

Келтирилган дастурда худди шу иш бажарилган, бу ерда тартибга келтиришнинг энг оддий усули ишлатилган: А массивда энг кичик элемент бор ва бу элемент биринчиси билан ўрнини алмаштириб боради. Кейин бу жараён A[2:N] кесмага ва ҳоказо қўлланилади.

Агар дастурни мураккаблаштириш керак бўлса, I массивсиз ишни бажариш мумкин, лекин А массив элементлари ўзгармаслиги керак. Бунда навбатдаги минимал элемент (MI) индексини чоп этиш ва эслаб қолиш етарли. Бунинг учун, албатта, биринчи шундай элементнинг индексини ва биринчи максимал элемент индексини олдиндан топиб қўйиш керак.

Дастур

```
Program Tartib_indekslari;
const NN=100;
label 1,2;
var
    j,i,m,mi,ma,k,n :integer;
    A : array [1..NN] of real;
begin
    Writeln('N:=');
    readln(n);
    for i:=1 to n do
    begin
      writeln ('a[',i,']:=');
      readln (a[i]);
    end;
    mi:=1; ma:=1;
    for k:=1 to n do
        begin
            if a[k]<a[mi] then mi:=k;
            if a[k]<a[ma] then ma:=k;
        end;
    writeln (mi);
    for m:=2 to n do
        begin
            i:=ma;
            for k:=1 to n do
```

```
                    begin
                        if (a[k]<a[mi]) or (a[k]=a[mi]) and (k<=mi) then
                                        goto 1;
                        if a[k]=a[mi] then
                                begin
                                        i:=k;
                                        goto 2;
                                end;
                                if a[k]<a[i] then
                                i:=k:
                    1 : end;
                    2 : mi:=i;
                    writeln (mi);
            end;
end.
```

11 – масала

Алгоритм «Ноллаштириш»

Масалани ечиш учун энг осон йўл, B[1:m] ва C[1:n] иккита ёрдамчи массивларни киритиш. Кейинчалик A массив элементлари кўздан кечирила бориб, B ва C массивларда ноллар учраган сатр ва устунлар белгилаб борилади, яъни: **if** a[i,j] =0 **then** b[i] =1 : c[j] =1 бажарилади.

Энди, A массив иккинчи марта кўздан кечирилаётганда, B ёки C массивларда белгиларга эга бўлган, сатр ва устунларда жойлашган элементлар ноллар билан алмаштирилади, яъни

if b[i] =1 **or** c[j] =1 **then** a[i,j] =0

Фақат битта C[1:n] ёрдамчи массивдан фойдаланадиган бошқа ечимни таклиф қилиш мумкин. Бу ҳолда A массив сатрлар бўйича кўздан кечирилади. Нол элементлари бўлган устунлар, аввалгидай C массивда белгиланади, кўрилаётган сатрда нол учрагани эса Z ўзгарувчида қаралади, яъни:

if a[i,j] =0 **then** c[j] =1 : Z=1

i сатр қараб чиқилгач, агар z-1 бўлса, сатр ноллаштирилади. Иш сўнгида C массив қаралади ва агар c[j] =1 бўлса, j устунлар ноллаштирилади.

Бу масалани ечишга хос хатоликлар :

1) А массив қаралади ва агар a[i,j] элемент нолга тенг бўлса, i сатр ёки j устун ёки униси ҳам, буниси ҳам ноллаштирилади. Кейинги қарашларда ноллаштирилган элементлар бошланғич нолга тенг элемент, деб қабул қилинади ва ортиқча ноллаштиришларни ҳосил қилади.

2) А массив қаралади ва агар a[i,j]=0 бўлса, навбатдаги c[k] элементга j киритилади. Бундан битта j рақам кўп марта киритилиши ва С массив, шартга зид равишда, m*n ўлчамда талаб қилиниши мумкин.

Дастур

```
Program Nollashtirish;
const NN=20;
      MM:=20;

var
      i,j,m,n :integer;
      Z       : boolean;
      A : array [1..MM,1..NN] of real;
      C : array [1..NN] of boolean;
begin
      writeln (‘M,N:=’);
      readln (m,n);
      for i:=1 to m do
         for j:=1 to m do
             begin
                write (‘a[‘,i,’,’,j,’]:=’);
                readln (a[i,j]);
             end;
         for j:=1 to n do C[j]:=false;
         for i:=1 to m do
             begin
                z:=false;
                for j:=1 to n do
                if a[i,j]=0 then
                                  begin
```

```
                        z:=true;
                        c[j]:=true;
                    end;
            if z then for j:=1 to m do a[i,j]:=0;
        end;
            for j:=1 to n do
                if c[j] then for i:=1 to m do a[i,j]:=0;
            for i:=1 to m do
                begin
                    for j:=1 to n do writeln(a[i,j]:4);
                    writeln;
                end;
end.
```

12 – масала

Алгоритм «Улгуржи харид»

Улгуржи харид орттирмаларсиз қуйидагича топилади:

n1 =n 144: m=n-n1∗144

n2 =m 12: n3=m-n2∗12

уни фақат икки йўл билан арзонлаштириш мумкин ортиқча боғлам олиб, ортиқча жуфтлик олмаслик ёки ортиқча қути олиб, боғлам ҳам, жуфтлик ҳам олмаслик керак. Агар n3∗1.05>10.25 бўлса ортиқча:

n2 =n2 +1: n3 =0

боғлам олиш керак.

Агар ҳосил бўлган харидда (эски ёки ўзгартирилганида) n2∗10,25+n3∗1,05>114.00 бўлса, n1=n1+1: n2=0: n3=0 ни ҳосил қилиш керак.

Бу ечим ҳар хил нархларда (қути 12 боғламдан, боғлам 12 жуфтдан арзон бўлган ҳолларда) яроқлидир.

Тест мисоллар

n	n1	n2	n3
9	0	0	9
10	0	1	0
131	0	11	0
134	1	0	0

Дастур

```
Program Ulgurji_savdo;
var
        n1,n2,n3,m,n :integer;
begin
        writeln (‘N:=’);
        readln (n);
        n1:=n div 144;
        m:=n-n1*144;
        n2:=m div 12;
        n3:=m-n2*12;
        if (n3*1.05 )>10.25 then

                begin
                    n2:=n2+1;
                    n3:=0
                end;
        if (n2*10.25 + n3*1.05 )>114 then
                begin
                    n1:=n1+1;
                    n2:=0;
                    n3:=0
                end;
        writeln (n1,’ ’,n2,’ ’,n3 );
end.
```

13 – масала

Алгоритм «Тўнтарилган сонлар»

Бу масала (N^2 амалда бажарилиш учун) маълум бир фикрлашни талаб этади. Тадқиқ қилинадиган кесмани унинг маркази билан бериш (бу N та вариантни беради) ва марказдан иккала томонга симметрик элементларни таққослаш керак (бу N та гуруҳдан катта бўлмаган амалларни талаб қилади). Шундай қилиб, ечим N^2 амалда (доимий кўпайтувчи аниқлигигача) топилади. Ечимни расмийлаштиришда кесма маркази ўрнига (у элементга ҳам,

элементлар ўртасига ҳам тўғри келиши мумкин) марказ билан қўшни бўлган икки элемент рақамини LN<PN олиш қулай. Бу рақамлар тушунарли тарзда силжийди:

(LN,PN)=(1,2),(1,3),(2,3),(2,4),...

таққосланадиган элемент рақамлари бошда L=LN : P=PN,

деб олинади ва кейин таққосланадиган A[L] ва A[P] элементлар тенг бўлгунича ёки L ва P нуқталардан биттаси (1,N) чегарадан чиққунча, ҳар қадамда L=L-1 :P=P+1 бажарилади. Шунда (LN,LP) маркази максимал кесма M=P-L-1 узунликда бўлади.

Дастур

```
Program To’ntarilgan_sonlar;
const NN=100;
var
      l,i,ln,pn,p,m,max,n     :integer;
                          z   :boolean;
      A : array [1..NN] of real;
begin
      writeln (‘n:=’);
      readln(n);
      for i:=1 to n do
          begin
            writeln (‘a[‘,i,’]:=’);
            readln (a[i]);
          end;
      max:=1; z:=true; ln:=1; pn:=2;
      while 2*(n*pn+1)+1>max do
          begin
            l:=ln; p:=pn;
            while (l>=1) and (p<=n) and (A[l]=A[p]) do
                  begin
                     l:=l-1; p:=p+1;
                  end;
            m:=p-l-1;
            if (max<m) then max:=m;
```

```
            if z then pn:=pn+1 else ln:=ln+1;
            z:=NOT z ;
          end;
      writeln (max);
end.
```

14 – масала

Алгоритм «Икки марта монотон»

i=1, j=n деб олинади, a[i,j] элемент (аввал бу, кейин навбатдагиси) x сони билан таққосланади.

агар a[i,j] =x бўлса, жавоб топилади,

агар a[i,j] < x бўлса, i=i+1 деб олинади,

агар a[i,j] > x бўлса, j=j-1 деб олинсин.

Охирги икки ҳолда i ≤ m ва j ≥1 қолганлигини текшириш керак. Агар қолган бўлса, таққослашга қайтиш, акс ҳолда «йўқ» деб чоп этиш керак. Ҳар бир қадамда i оширилиб, ёки j камайтирилиб борилади. Демак, қадамлар m+n дан кўп бўлмайди.

Бу алгоритмнинг тўғрилигига ишонч ҳосил қилиш қийин эмас. Ҳақиқатдан ҳам, агар a[i,j]< x бўлса, x сони a[i:m,1:j] массивнинг i сатрида йўқ ва бу сатрни ташлаш мумкин. Агар a[i,j]> x бўлса, худди шу сабабга кўра j устунни ташлаш мумкин.

Дастур

```
Program Ikki_marta_monoton;
const MM=20;
      NN=20;
label 1;
var
      i, j, m, n, x :integer;
      A : array [1..MM,1..NN] of real;
begin
      writeln (‘m,n:=’);
      readln (m,n);
      writeln (‘x:=’);
```

```
        readln (x);
        for i:=1 to m do
        for j:=1 to n do
            begin
                writeln ('a[',i,',',j,']:=');
                readln (a[i,j]);
            end;
        i:=1; j:=n;
        while (i<=m) and (j>=1) do
            begin
                if a[i,j]=x then goto 1;
                if a[i,j]<x then i:=i+1 else j:=j-1;
            end;
        writeln ('Йўқ');
    1: writeln (i,' ',j);
end.
```

15 – масала

Алгоритм «Тетраэдрлар»

Ечим икки қисмга бўлинади: М1 билан мос келтириш учун N тетраэдрнинг N′ қиррасини ва М1 да қолдирадиган N тетраэдрнинг бурилишларини танлаш. Биринчиси 4 усул билан, иккинчиси 3 та усул билан бажарилади. Фақат, N' танланганидан кейин N нинг қолган қирраларини қайси биттасидандир бошлаб маълум тартибда (масалан, соат йўналишига тескари) ёзиб олиш, ечимнинг иккинчи қисмини функция кўринишида расмийлаштириш керак (дастурда келтирилган).

Ечимдан ҳамма 12 вариант бевосита ёзиб олинадиган жавоб қониқарли эмас.

Дастур

```
Program Tetraedrlar;
                                    var
                        m1,m2,m3,m4,n1,n2,n3,n4 :integer;
        function DA (i,j,k,l:integer) : boolean;
begin
```

```
        DA:=((m1=i) and (((m2=j) and (m3=k) and (m4=i))
        or ((m2=l) and (m3=j) and (m4=k))
        or ((m2=k) and (m3=l) and (m4=j))))
end;
begin
        writeln (‘m1,m2,m3,m4:=’);
        readln (m1,m2,m3,m4);
        writeln (‘n1,n2,n3,n4:=’);
        readln (n1,n2,n3,n4);
        writeln;
        if DA (n1,n2,n3,n4) or DA(n2,n1,n4,n3) or DA(n3,n1,n2,n4) or
                        DA(n4,n1,n3,n2)   then writeln(‘Ҳа’)
                                          else writeln(‘Йўқ’);
end.
```

16 – масала

Алгоритм «Каср даври»

Ечим шартда кўрсатилмаган қўшимча талабларга қаттиқ боғлиқ. «M» натурал сонни «N» натурал сонга бўлганда бўлинманинг бутун қисми ва қолдиқ ҳосил бўлади:

i=M/N : к=M-i∗N

Масалани ечиш учун аввал M/N касрдан M суратни қолдиққа алмаштириб, бутун қисмини йўқотамиз:

M= M-M/N∗N

Энди, ҳосил бўлган касрнинг i бўлинма ва M қолдиқ рақамларини кетма-кет ҳосил қилиш мумкин:

i= 10∗M/N : M=10∗M-i∗N ва ҳоказо.

Ҳар бир қолдиқ N дан ошмайди. Демак, ҳар хил қолдиқлар Nдан катта бўлмайди ва улар такрорлана бошлайди. Қолдиқ қачон такрорланса, бўлинмалар ҳам такрорлана бошлайди, давр бошланади. Такрорлашни илғаб олиш учун, энг осони D[1:N] массивни ташкил қилиш ва унга, ҳосил бўлиш тартибига кўра, қолдиқларни ёзиш ва ҳар бир янги қолдиқни барча олдингилари билан таққослаб бориш керак.

Баён этилган ечим тўғри, лекин яхши эмас. Ҳисоб бўйича К-қолдиққа К таққослаш, ҳамма қолдиқларга эса $N^2/2$ таққослаш кетади. Замонавий ЭҲМ

ларда 10^8 тартибли бутун M ва N ни ҳосил қилиш осон, лекин 10^{16} амалларни бажариш учун йиллар кетади, шу сабаб N сони учун D массивни ташкил қилишга келишилган экан, уни тозалаймиз ва навбатдаги M қолдиқнинг пайдо бўлишини D[M] элементда белгилаймиз. У пайтда К қолдиқ «янгилиги»ни текшириш битта таққослашни эгаллайди.

Лекин бу ечим ҳам жиддий камчиликка эга. N сони шундай катта бўлиши мумкинки, бунда N та амални бажариш мумкин бўладию, лекин Nта элементдаги массивни тезкор хотирага киритиш мумкин бўлмай қолади. Қолдиқлар учун массивни ташкил қилмай, даврни топишга уриниб кўрамиз. Дастлаб аввалги каби «M» дан «N» га бўлинадиган қисмини ажратамиз. Кейин бўлинманинг N рақамини ўтказамиз. Энди, давр бошлангани маълум бўлгач, битта ягона қолдиқни эслаб қоламиз ва бўлинманинг рақамлари такрорлана бошланмагунича чоп этамиз. Бу ишлар таклиф этилган дастурда келтирилган. Бу дастур олдингисидан қисқа.

Дастур

```
Program Kasr_davri;
var
     m,n,i,j,k :integer;
begin
     writeln (‘m,n:=’);
     readln (m,n);
     writeln;
     M:=m-(m div n)*n;
     k:=1;
     while (k<=n) or (j<>m) do
     begin
          if k=n then j:=m;
          i:= 10 * m div n;
          m:=10 * m-i * n;
          if k>=n then write(i);
          k:=k+1;
     end;
     writeln ;
end.
```

Xulosa

Informatikada masala echish tushunchasi deganda axborotlarni qayta ishlab, natijani oldindan belgilangan ma'lum bir ko’rinishga olib kelish tushuniladi.

Kompyuterdan foydalanib masalani echish - yaratilgan algoritmga asoslangan holda dastlabki ma'lumotlar ustida avtomatik tarzda amallar bajarilib izlangan natija (natijalar) ko’rinishiga keltirish demakdir.

Biz bu bitiruv malakaviy ishimizda masalalarni kompyuterda echish bosqichlari, masalalarni kompyuterda echishning algoritmlash bosqichi, algoritm tushunchasi, algorim xossalari, algoritm turlari va hisoblshga oid bir qancha amaliy masalalarning algoritmi va dasturini tuzdik. Biz bu diplom loyihani tuzishimiz jarayonida ko’pgina murakkab toifadagi masalalarning algoritmini va paskal dasturlash tilidagi dasturini puxta o’rganib chiqdik. Shu toiufadagi murakkab masalalar berilgan bo’lsa, qinalmasdan uning algoritmi va dasturini tuzishimga o’zimda kuch topa olaman. Bu algoritm va dasturlarni hayotga tadbiq qilish va kelajak avlodga o’rgatishni o’z oldimga maqsad qilib qo’ydim.

Foydalanilgan adabiyotlar ro’xati

1. O`.T.Haitmatov va b.Informatika va axborot texnologiyalari. O’quv qo’llanma. T. TKTI. 2005 y.
2. O`.T.Haitmatov va b. Informatika va axborot texnologiyalari fanidan laboratoriya ishlarini bajarish ushun uslubiy qo’llanma. T. TKTI. 2005 y.
3. Holmatov T.X.,Toyloqov N.I. Amaliy matematika,dasturlash va kompyuterning dasturiy ta’minoti. T.Mexnat, 2000 .
4. Kadirova N.R. Polatov A.M. Programmirovanie na yasike Paskal T.2004
5. Aripov M., Xaydarov A. Informatika asoslari T. “O`qituvchi” 2002 .
6. Alyaev Yu.A. i dr. Praktikum po algoritmizatsii i programmirovaniyu na yazik Paskal: Ucheb.posob. – M.: FIS, 2004. – 528s.
7. Bartenev O.V. Predpriyatie: programmirovanie dlya vsex bazovie ob'ekti i rascheti na odnoy diskete. – M.: Dialog – MIFI, 2003. – 464 s.
8. Faronov V.V. Turbo Paskal 7.0. Uchebnoe posobie. M.: Nolidj., 2002g.
9. Stavrovskiy A.B. Turbo Paskal 7.0. Kiev: BHV., 2000g.
10. Gurova L.I. Osnovi programmirovaniya. M., Fis, 1990.
11. Dokukina T.K. Programmirovanie i algoritmicheskie yaziki. Uchebnik.M., Mashinostroenie, 1988.
12. Chernov B.I. Programmirovanie na algoritmicheskix yazikax. M., Prosveshenie, 1991.
13. Donald Alkol. Yazik Paskal v illyustratsiyax. M., Mir, 1991.
14. Faronov V.V. Turbo Paskal (v 3x knigax). M.: MVTU-FESTO DIDAKTIK., 2000.
15. Fedorov A. Osobennosti programmirovaniya na Borland Pascal. Kiev:Dialektika, 1999.
16. Vasyukova N.D., Tyulyaeva V.V. Praktikum po osnovam programmirovaniya yazik Paskal. M.: Visshaya shkola., 1996.
17. Zavarikin V.M. Texnika vichisleniy i algoritmizatsiya. Uchebnoe posobie M., Prosveshenie, 1987

18. Dyakonov V.Yu. Sistemnoe programmirovanie. Uchebnoe posobie. M., VSh, 1990
19. Donald E. Knut. Iskusstvo programmirovaniya. (pod red. prof. Yu.V. Kazachenko) M., Vilnyus, 2001g.
20. http:// informatika. freent.uz
21. www.tehnolgi.ru
22. www.internet.ru
23. www.microsoft.com
24. http://www.ictcouncil.gov.uz –O'zbekiston Respublikasida qabul qilingan qonun va qarorlar.
25. http://www.gov.uz –O'zbekiston Respublikasi Xukumat portali.
26. http://www/ziyo.edv.uz - OO'MT Vazirligi Veb-sayti.
27. http://www/tsue.fan.uz - TDIU Veb-sayti.
28. http://www.uzinfocom.uz/lang/uzb - "UzInfoCom" Kompyuter va Axborot Texnologiyalarini Rivojlantirish va Joriy Etish Markazi ko'magida ishlab chiqilgan.