

**O'ZBEKISTON RESPUBLIKASI
OLIY VA O'RTA MAXSUS TA'LIM VAZIRLIGI**

**A.Navoiy nomidagi
Samarqand davlat universiteti**

Mexanika-matematika fakulteti

“Axborotlashtirish texnologiyalari” kafedrası

Aminov I.B.

Informatika va dasturlash

Ma'ruzalar matni

Samarqand-2010

1-ma'ruza

Mavzu: Kirish. Informatika va informatsion texnologiyalarning asosiy tushunchalari, rivojlanish tarixi va axborotlashgan jamiyat taraqqiyotidagi roli

Reja:

1. Informatika to'g'risida ma'lumotlar.
2. Informatika fanining predmeti.
3. Informatikaning asosiy qismlari.
4. Informatika va HT ning rivojlanish tarixi.

Darsning maqsadi:

1. Talabalarda informatika to'g'risida umumiy va asosiy tushunchalarni hisil qilish.
2. Informatika va informatsion jarayon to'g'risida tasavvurga ega bo'lish.
3. Informatika va HT ning rivojlanish tarixi bo'yicha ma'lumotga ega bo'lish.

Tayanch iboralar: *informatika, dokumentalistika, informasion texnologiya, mexanik mashina, elektron hisoblash mashinasi, EHM avlodlari.*

Dars o'tish vositalari: *sinf doskasi, o'quv-uslubiy qo'llanmalar, ma'ruza matnlari, komputer, Power Point dasturi, ma'ruza bo'yicha slaydlar.*

Dars o'tish usuli: *namoyish dasturiy vositalat ishtirokida, jonli muloqotli ma'ruza.*

Darsning xrono xaritasi – 80 minut.

Tashkiliy qism: *xonaning tozaligi, jihozlanikishi, sanitariya holati. Talabalarning davomati*– 2 minut.

Talabalar bilimni baholash: *o'tilgan mavzuni qisqacha takrorlash, talaba bilan savol javob o'tkazish* - 10 minut.

Yangi mavzu bayoni - 55 minut.

Mavzuni o'zlashtirish darajasini aniqlash va mustahkamlash – 10 minut.

Sinov savollari – 5 minut.

Uyga vazifa berish – 3 minut.

Ma'ruza bayoni

Informatika to'g'risida gap ketganda, har bir kishi qandaydir axborotlar haqida fikr borayotganligini va bu axborotlar nimagadir yoki kimgadir tegishli ekanligini tushunadi. Bu axborotlar qayerdan olingan, qanday saqlangan va ularni manbai qayerda ekanligi ko'pchilikni qiziqtirishi mumkin.

Qadimda insonlar, qog'oz mavjud bo'lmagan vaqtlarda, o'sha davrga tegishli ma'lumotlarni toshlarga, daraxt barglariga, gildan (loydan) yasalgan tablichkalarga yozib qoldirishgan. Bu ma'lumotlar esa cherkovlarda yoki ibodatxonalarda saqlangan. Shuning uchun xam bu ma'lumotlarga ega bo'lish, ularni o'rganish hammaga ham nasib qilavermagan. O'qishni yoki yozishni bilmagan kishilar ulardan foydalana olishmagan.

Vaqt o'tishi bilan insonlar o'zgardi, jamiyatning tuzumi o'zgardi. Jamiyatda o'qimishli kishilar ko'paydi. Yangidan-yangi axborotlar hosil bo'ldi. Bu ax-

borotlarni ko'paytirish va ularni barcha qiziquvchi kishilarga tarqatish muammosi hosil bo'ldi. Axborotlarni tarqatish vositasi sifatida asosan kitoblardan, rasmlardan va hokazolardan foydalanila boshlandi. Chop etilgan kitoblardan yoki rasmlardan kishilar jamiyatda ro'y berayotgan o'zgarishlar, yangiliklar va voqyealar to'g'risida axborotlarga ega bo'lishdi.

Yillar o'tishi bilan hosil bo'ladigan axborotlar hajmi ortdi. Axborotlarni yig'ish, qayta ishlash va tarqatish uchun nashriyotlar, tipografiyalar qurildi, ya'ni informasion sanoatga asos solindi.

Ko'plab tirajda chop etilayotgan kitoblar, gazeta va jurnallar kishilarni katta hajmdagi axborotlardan xabardor qilib turishdi.

Matbuot bilan birgalikda radio, keyinchalik esa televedeniya axbo-rotlarni uzatish vositasi bo'lib qoldi.

Axborot tushunchasi nimani anglatadi? Qadimgi Gresiyada bu termin "ko'rinishni berish", "tasavvurlash" ma'nosida tushunilgan.

Ko'rinishni berish – bu rasmlar orqali konstruktorlik g'oyalarni hosil qilish, kinofilmlar yaratish, gramplastinkalar ishlab chikish va hokozalardan iborat. Insonlarning kuzatish natijalari va egallagan bilimlarini bir ko'rinishdan boshqa ko'rinishga o'tkazilishi-bu axborotlarni qabul qilish yoki saqlash va ishlash uchun juda qulaydir.

Tavsiflash– buni modellashtirish yoki model yasash ma'nosida tushunish mumkin. Bu qanaqa model, u qanday ko'rinishda ifodalanganligi qo'yilgan aniq masalaga, uning tiliga va usuliga bog'liq bo'ladi.

Masalan, fizik fotoplastistinka elementar zarrachalarining qoldirgan trayektoriyasini o'rgana turib, formula va tenglamalardan iborat matematik modelni yaratadi, ruxshunos esa insonlarni turli vaziyatlardagi holatlarini tavsiflovchi model hosil qiladi.

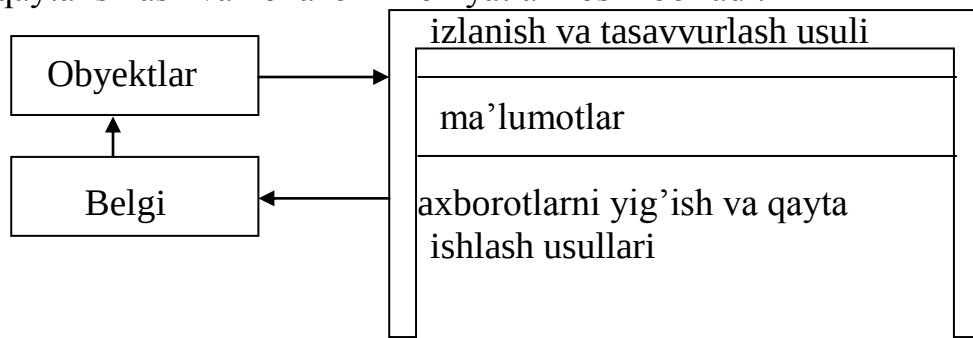
Har bir inson o'zining ish faoliyati davomida u yoki bu ko'rinishda model-lashtirish bilan yoki kelayotgan, chiqayotgan axborotlarni tasvirlash bilan shug'ullanishadi.

XX asrda fan- texnikaning va sanoatning rivojlanishi natijasida hosil bo'ladigan va saqlanadigan axborotlar hajmi shu darajada ortib ketdiki, natijada ularning hammasini insonlar qabul qilishi, saqlashi va qayta ishlashi mumkin bo'lmay qoldi.

Hosil bo'layotgan axborotlarni sinflarga bo'lish, saqlash, axborot-larni harakatlanish qonuniyatini yaratish muammosi hosil bo'ldi. Bu muammoni hal qilish uchun olib borilgan izlanishlar natijasi sifatida **informatika** deb atalmish fan paydo bo'ldi. Boshlang'ich bosqichda **informa-tika** kutubxona ishining bazasi hisoblangan va ko'p yillar yordamida uni mukammallashtirish nazariyasi va amaliyoti bilan shug'ullanib kelgan.

Informatika o'rganilayotgan obyekt va u to'g'risidagi bilimlar oralig'idan joy egallab qoladi. Haqiqatan ham, inson atrof muhitni o'rgana turib, axborot oladi, uni biron narsaga belgilab, yozib yoki saqlab qo'yadi. **Axborot** tashuvchi sifatida adabiyot, magnit lentalar, kartalar, sxemalar ishlatilishi mumkin. Axborotlarni qayta ishlash orqali, bizni o'rab turgan dunyo to'g'risida bilimga ega bo'lamiz, na-

tijada yangi izlanish usullarini yaratish, yangi axborotlarga ega bo'lish, ularni saqlash, qayta ishlash va hokazo imkoniyatlar hosil bo'ladi.



Axborotlarni yig'ish, saqlash, qayta ishlash va tasvirlashning samarali usullarini yaratish informatikaning asosiy maqsadlaridan biri bo'lib qoldi. Asrimizning 50-yillarigacha masalaning bunday qo'yilishi haqiqat bo'lib, axborotlarni yig'ish va qayta ishlash usullari bo'yicha umumiylik yo'qdek edi. Tibbiyotda, geografiyada, fizikada, filosofiyada va boshqa sohalarida axborotlarni yig'ish va qayta ishlashda bog'liqlik yo'q edi. Ko'pchiliklarning fikricha matematika bilan fizika, ximiya bilan tibbiyot o'rtasida bog'liqlik borligi tan olinar edi. **Kompyuterlarning** paydo bo'lishi bilan bu holat tezlik bilan sezilarli darajada o'zgardi.

Ko'pchilikka ma'lumki, birinchi **EHM (elektron hisoblash mashina-lari)** atom fizikasida, uchish va raketa texnikasida hisoblash ishlarini bajarish uchun yaratilgan. Keyinchalik ularni boshqa sohalarida: qishloq xo'jaligida, sanoatda, boshqarish sistemasida, tibbiyotda, o'quv jarayonlarida va hokazolarda qo'llanilishi yangi sanoat sohasi – **axborotlarni elektron qayta ishlash usullari va vositalari sohasi** yaratilishiga va uni tez sur'atlar bilan rivojlanishiga asos bo'ldi.

Yangi elektron hisoblash mashinalari, ular bilan muloqot qilish-ning yangi vositalari va usullari yaratildi. Hisoblash texnikalari ishlab chiqa-radigan sanoat korxonalari qurildi. Birinchi EHM lari hisoblash masalalarini yechish uchun yaratilgan, keyinchalik **sonli, matnli, grafik** va boshqa axborotlarni ham qayta ishlovchi **kompyuterlar** yaratila boshlandi.

Hisoblash texnikasi barcha sohalarida o'zining samaradorligini, keng imkoniyatli ekanligini va qulayligini ko'rsatdi.

Kompyuter va ular bilan muloqot qilish usullarini va metod-larining rivojlanishi natijasida, ularning qo'llanilish sohaları ken-gayib bormoqda.

Hozirgi kunda kompyuterlar barcha sohalarida, shu jumladan sanoatda, qishloq xo'jaligida, ishlab chiqarishning barcha tarmoqlarida, fan texni-kada, tibbiyotda, o'quv jarayonlarida keng qo'llanilib kelinmoqda.

Informatika nima? Agar u bizni o'rab turgan borliq to'g'risidagi axborotlarni yig'ish va qayta ishlashdan iborat bo'lsa, u holda uni ximiyadan, biologiyadan va boshqa fanlardan qanday ajratish mumkin. Balki boshqa barcha fanlar uning asosiy tarkibiy qismidir? Yo'q, **informatika** hych qanday fanni na ximiyani, na fizikani na tibbiyotni va boshqalarni o'z ichiga olmaydi. Lekin ularning har biri bilan juda ko'p umumiylikka ega. U boshqa fanlarga yordam berish uchun yaratilgan, ya'ni matematika fani bilan birgalikda ularni axborotlarni saqlash, uzatish, tekshirish va qayta ishlash usullari bilan ta'minlaydi. Haqiqatdan

ham, tajriba o'tkazish jarayonida olimlar turli xil vositalar yordamida axborotga ega bo'lishadi, uni saqlaydi, qayta ishlaydi va lozim bo'lsa undan boshqa jarayonlar uchun ham foydalanadi.

Axborotlarni qabul qilish, yozish, saqlash va qayta ishlash maxsus algoritmlar orqali amalga oshiriladi. Olingan natijalar tahlil qilingandan keyin kerakli xulosaga kelinadi. Biz hozir abstrakt tajriba haqida so'z yuritayapmiz. Aslida esa u fizik, geografik, ximik va boshqa tajriba bo'lishi mumkin. Datchiklar ma'lumotlarni to'g'ridan to'g'ri kompyuterga uzatishi mumkin yoki laborant asbob ko'rsatkichini avval daftarga yozib olib, so'ngra uni kompyuterga kiritadi. Asosiysi ma'lumotlarni xotira qurilmasiga yozish, saqlash mumkin va qayta ishlash uchun esa algoritmi mavjud bo'lishi kerak.

Ma'lumotlarni qayta ishlash tajribaning asosiy ajralmas qismi hisoblanadi. Axborotlarni qayta ishlash algoritmlari olimlar va mutaxassislar tomonidan tayyorlanadi. Dastur tuzuvchilar bilan birgalikda ular kompyuter uchun ma'lumotlarni qayta ishlovchi dasturlar tuzishadi. Algoritmlar yaratish va ular asosida kompyuter uchun dastur tuzish – ana shu bosqichning eng asosiysi hisoblanadi.

Keyingi bosqichda tekshirish uchun ma'lumotlarni qulay ko'rinishda ifodalash va chiqarish dasturlaridan foydalaniladi. (grafiklar qurish, jadvallar yaratish, ishchi rasmlarni tasvirlash chizish va hokazo). Ma'lumki, bunday dasturlar aniq sohaga mo'ljallanmagan va ular yetarlicha universal hisoblanadi.

Amaliyot shu narsani ko'rsatdiki, kompyuterlardan foydalanish natijasida ishlab chiqarishda va fan texnikada mehnat samaradorligi orta-di, ilmiy-texnik progressga juda kuchli ta'sir etadi.

Kompyuterlarning rivojlanishida asosan uchta bosqichni ajratish mumkin: **hisoblash, umumiy informasion va intellektual**. Fan va texnologiya hozirgi vaqtda uchinchi bosqichda, ya'ni **mashina intellektuali-ning** rivojlanish bo'sag'asida turibdi.

Mashina intellekti hayotga aqliy mehnat kishilari bajaradigan ishlarni bajaruvchi kompyuterlar ko'rinishida kirib bormoqda.

Yangi mashinalar ishlab chiqiladi, ko'plab mukammal dasturlar yaratiladi, mashina intellekti o'sadi, ya'ni bizni o'rab turgan olamni anglash va tekshirish uchun imkoniyatlar hosil bo'ladi.

Zamonaviy informatikani quyidagi **3 yo'nalishi** tashkil etadi:

1) Axborotlarni avtomatik yig'ish, saqlash, ishlash va uzatish usullari va algoritmlarini yaratish;

2) Axborotlarni qayta ishlash, tasavvurlash usullari va algoritmlarini yaratish.

3) Yuqoridagi ikkita yo'nalishni rivojlantirish uchun elektron hisoblash mashinalari va texnologiyalarni yaratish.

Ingliz tilida **informatika** so'zining sinonimi **computerscieni** (hisoblash fani) bo'lib, u informatikaning predmetini to'la yoritmaydi. **Informatika** termini bizga fransuz tilidan kelgan bo'lib, kompyuterlar va ularni qo'llanilishi haqidagi fanni anglatadi.

Hisoblash texnikasining rivojlanish tarixi.

Qadim zamonlardan beri insonlar o'z mehnatini yengillashtirishga, avtomatlashtirishga yoki mexanizasiyalashtirishga harakat qilishgan. Bu harakatlarning barchasi jismoniy kuchlarga yoki mehnatga qaratilgan bo'lib, insonlar aqliy ish faoliyatida hech qanday mexanizasiyadan (texnikadan) foydalanishga ehtiyoj sezmagani. Ilm-fanning va jamiyatning rivojlanishi natijasida katta hajmdagi axborotlarni qayta ishlashga yoki murakkab ko'rinishdagi arifmetik amallarni bajarishga zarurat tug'ildi, ya'ni murakkab hisoblash jarayonlarini bajarish uchun maxsus texnik vositalar yaratishga ehtiyoj sezildi.

Tarixda birinchi hisoblash texnikasining yaratilishi fransuz filosofi, yozuvchisi, matematigi va fizigi Blez Paskal nomi bilan bog'langan. U 1641 yilda qo'shish va ayirish amallarini bajaradigan mexanik hisoblagichni yaratdi.

1673 yilda mashhur nemes olimi Gorfrid Leybnis to'rt arifmetik amalni bajaradigan hisoblash mashinasini qurdi. G.Leybnis yaratgan mashinaning asosiy mexanizmlari XX asrning o'rtalarigacha ham ba'zi bir mashinalarda qo'llanilib kelindi. Leybnis yaratgan mashina turiga, ko'paytirish amalni ko'p martalab qo'shish, bo'lish amalini ko'p martalab ayirish kabi bajaradigan barcha mashinalarni, shu jumladan birinchi EHM larini kiritish mumkin. Bu mashinalarning eng asosiy xususiyati insonlarga nisbatan hisoblashlarni juda yuqori aniqlikda va tezlikda amalga oshirishdan iborat edi. Bu mashinalarning yaratilishi insonlarning intel-lektual faoliyatini mexanizasiyalashtirish mumkinligini namoyish qildi.

Hisoblash mashinalarini yaratish va ularni mukammalltirish fan- texnikaning asosiy muammolaridan biriga aylandi.

Birinchi hisoblash mashinalarining paydo bo'lishi va ularga bo'lgan talabning oshishi, ularni yaratish texnologiyasini mukammalltirish ishlarini tezlatib yubordi. 1821 yil Tomas degan konstruktor hisoblash vositalarini ko'plab ishlab chiqarishni yo'lga qo'ydi va bu asboblarni **arifmo-metr**lar deb atala boshlandi.

Arifmometr boshqa mashinalarga nisbatan yuqori hisoblash tezligiga ega bo'lib, ikkita sakkiz raqamli sonlarni 18 sekundda ko'paytirish imkoniga ega bo'lgan.

XX asrda ko'plab hisoblash vositalari yaratildi, ularning ishonchliligi, hisoblash aniqligi va tezligi, ular bilan muloqot qilish imkoniyatlari oshirildi.

Hisoblash mashinalarini mukammallashtirishda va yaratishda rus olimlarining ham xizmatlari beqiyosdir. XVIII asr oxirlarida Ye.Yakobsonning mashinasi tayyorlandi, 1828 yilda Slobodskiy F.M. hisoblash vositasini yaratdi, 1846 yilda Kummer, P.L.Chebyshev mashinalari yaratildi. Peterburglik injener V.Odner tomonidan yaratilgan tishli arifmometr hisoblash texnikasining rivojlanishida muhim rol o'ynadi.

1873 yildan boshlab ishlab chiqarila boshlangan bunday asboblarni, shunchalik mukammal tuzilishga ega bo'lib, ko'p yillar davomida o'zgarishsiz qo'llanib kelindi.

Bunday hisoblash mashinalari insonlarning hisoblash ishlarini yengillashtirishi bilan birga, ularsiz mashina hisoblash ishlarini bajara olmas ham edi.

XIX asrning boshlarida ingliz matematigi Charlz Bebbidj prinsipial yangi turdagi hisoblash mashinasining asosini yaratdi va uning asosiy holatlari (xususiyatlari)ni bayon qilib berdi:

1. Mashina raqamli axborotlarni saqlash uchun “ombor”ga ega bo’lishi kerak (hozirgi kompyuterlarda bu xotira qurilmasi).

2. Mashinada “ombor”dan olingan sonlar ustida amallar bajaruvchi qurilma mavjud bo’lishi kerak (Zamonaviy kompyuterlarda bu arifmetik qurilma hisoblanadi).

3. Mashinada amallarning bajarilishi ketma-ketligini boshqa-ruvchi qurilma, ya’ni boshqarish qurilmasi bo’lishi kerak.

Mashinada boshlang’ich ma’lumotlarni kiritish va hosil bo’lgan natijalarni chiqaruvchi qurilmalari, ya’ni kiritish – chiqarish qurilmalari mavjud

Bundan 150 yil ilgari surilgan bu g’oyalarni XIX asrda ro’yobga chiqa-rib bo’lmagan bo’lsada, lekin zamonaviy kompyuterlarda bularning barchasi o’z aksini topgan. Bebbedj mexanik arifmometrlar asosida shunday turdagi mashinalarni yaratishga harakat qilgan. Lekin mexanizmlarning yo’qligi va juda qimmatligi tufayli bu ishni oxiriga yetkaza olmadi.

XX asrning 40-50 yillarida elektronika, avtomatika va matematik mantiq sohasida juda katta yutuqlarga erishildi, ya’ni birinchi elektron hisoblash mashinalari yaratildi.

1944 yilda amerikalik fizik G.Ayken loyihasi asosida avtomatik hisoblash mashinasi Mark-1 ni yaratish bo’yicha ishlar tugallandi. Bu mashinada asosan elektron mexanizmlardan foydalanilgan. Katta hajmga va og’irlikka ega bo’lgan bu mashina bor yo’g’i 10 ta arifmometr kuchiga ega edi. Ishlash tezligining va imkoniyat darajasining pastligi tufayli bu mashinadan amaliyotda foydalanish mumkin bo’lmadi.

Releli mashinalarning ishlash tezligining pastligi tufayli ular hisoblash ishlashlarini avtomatlashtirish ishida unchalik muhim o’rinni egallamadi. Lekin faqatgina elektronika hisoblash texnikasiga, oldingi yaratilgan elementlarga nisbatan juda yuqori tezlikda sifatli ishlash imkoniyatini yaratdi. Elektron sxemalar asosida yaratilgan hisoblash mashinalari, hisoblash texnikasini yaratishda yangi yo’nalishlarni ochib berdi. Bu mashinalar elektron hisoblash mashinalari (EHM) deb atala boshlandi.

1946 yilning boshlarida Amerika Qo’shma Shtatlarida, asosiy elementi elektron lampalardan iborat bo’lgan, dunyoda birinchi elektron hisoblash mashinasi namoyish qilindi. Bu mashina **ENIAK** deb nomlanib, ikkita o’n raqamli sonlarning ko’paytmasini **0,0028** sekundda bajarish tezligiga ega edi.

ENIAKning yaratilishi hisoblash texnikasini rivojlantirish bo’yicha izlanish va imkoniyatlar uchun yo’l ochib berdi.

Elektron hisoblash texnikasining rivojlanishida amerikalik mashhur matematik **Jon fon Neymanning** xizmatlari juda katta bo’ldi. **Neyman** tomonidan bildirilgan, sonlarni ifodalashda ikkilik sanoq sistemasidan foydalanish va foydalanilayotgan axborotlarni saqlash kabi g’oyalar hozirgi zamon hisoblash mashinalarida o’zining amaliy aksini topgan. **1949** yilda Angliyada Jon fon Neyman prinsiplari asosida elektron hisoblash mashinasi –**EDSAK** yaratildi va amaliyotda qo’llanildi.

Birinchi sovet EHMi 1950 yilda akademik S.A.Lebedev rahbarligida yaratildi. U **MESM (kichik elektron hisoblash mashinasi)** deb ataldi. Bir yildan keyin S.A.Lebedev rahbarligida **BESM (katta elektron hisoblash mashinasi)** yaratildi.

EHM avlodlari. EHM larning rivojlanish tarixini o'rganishda ular bir nechta avlodlarga bo'linadi.

50-yillarda ishlab chiqarilgan birinchi EHMLar **birinchi avlod** mashinalari hisoblanadi. Ularning asosini elektron lampalar tashkil qilgan bo'lib, ishlash tezligi va ishonchliligi juda past bo'lgan.

1955 yildan boshlab, **ikkinchi avlod** EHMLari paydo bo'ldi. Ularda elektron lampalarning o'rniga – yarim o'tkazgichlar – tranzistorlar qo'llanilgan. EHMLari hajm jihatidan kichiklashgan, ularga kam miqdorda elektroenergiya kerak edi. Ularning ishlash tezligi esa katta bo'lib, sekundiga bir necha o'n ming amalni bajarar edi. Ana shu davrdan boshlab hisoblash mashinalarida dasturlash tillaridan foydalanila boshlandi.

Bir necha vaqt o'tgandan keyin elektron sanoati integral sxemalar ishlab chiqara boshladi. Integral sxemalar- bu uncha katta bo'lmagan yarim o'tkazgichli kristallar bo'lib, ular bir necha yuzlab tranzistorlardan tuzilgan. Integral sxemalardan iborat EHMLari **uchinchi avlod** mashinalari hisoblanadi. Bu EHM katta xotiraga va yuqori ishlash tezligiga ega bo'lib, sekundiga bir necha milliongacha amalni bajaradi.

Zamonaviy kompyuterlar – bular **to'rtinchi avlod** EHMLari hisoblanadi. Ular 70-yillarning boshlarida yaratilgan. Ularning asosiy elementlari mikroprosessor va **katta integral sxemalar (KIS)** hisoblanadi. **KISlar** ham yarim o'tkazgichli kristallar bo'lib, bir necha yuz ming tranzistorlardan tuzilgan.

Mikroprosessorlarning yaratilishi - informatika fanidagi buyuk o'zgarishlardan biri hisoblanadi. Uning natijasi sifatida **shaxsiy kompyuterlar** paydo bo'ldi. Bu kompyuterlar o'zining xotirasining hajmi va ishlash tezligi jihatidan bir necha minglab birinchi avlod mashinalariga teng keladi.

Yuqorida keltirilgan to'rt avlod EHMLari bir-biridan keskin farq qiladi, lekin hammasida ham bir xil kamchilik mavjud. Bu kompyuterlarda ishlash uchun, ya'ni kerakli masalalarni yechish uchun, maxsus tildan foydalalanish kerak bo'ladi, aks holda kompyuter bilan muloqot qilib bo'lmaydi. Buning uchun yechiladigan masala biror dasturlash tilida yozilgan bo'lishi kerak.

Hozirgi vaqtda ko'plab mamlakatlarda **beshinchi avlod** EHMLarni yaratishga harakat qilinayapdi. Bu kompyuterlar inson tilini tushunadigan, tasvirlarni ko'radigan bo'lishi kerak. Bunday kompyuterlar paydo bo'lgan paytda, yechiladigan masalalar tabiiy so'zlar orqali ifoda qilinadi, dastur tuzish va uni yechishni kompyuter o'zi bajaradi.

Sinov savollari

1. Informatika fani paydo bo'lishi to'g'risida qanday to'g'risida ma'lumotlarni bilasiz?.
2. Informatika fanining predmeti nima?.
3. Informatikaning asosiy qismlari qaysilari?.
4. Informatika va HT ning rivojlanish bosqichlari nimalardan iborat?

5. Informatika faninin asosiy xususiyatlari?

Uyga vazifa : *Ichlab chiqarish jarayonlarini axborotlashtirish va axborotlarni qayta ichlash doir misollar kelterish va ularni tushuntirib berish.*

Adabiyotlar

1. Shafrin Yu. Osnovi kompyuternoy texnologii. Uchebnoi posobie- M: 1997, - 560 b.
2. Raxmanqulova S.I. IBM PC shaxsiy kompyuterda ishlash.- T:, NMK "Sharq"-INSTAR, 1996 y.
3. Figurnov V. E. IBM PC dlya pozovatelya. - M; Finantsi i statistika, 1990 g.
4. Ortiqov A., Mamatqulov A. IBM PC kompyuteridan foydalanish.- Toshkent, "Qonis" 1992 y.
5. Volvachev A.K., Krisevich V. S. Programmirovaniye na yazika Paskal dlya personalnix EC EBM- Minsk, Visshaya shkola, 1989 g.
6. Vasyukova N.D., Tyulyaeva V.V. Praktikum po osnovam programmirovaniya. Yazik Paskal- M; Visshaya shkola, 1991 g.
7. Zuev E.A. Yazik programmiravaniya Turbo Paskal 6.0, 7.0- M; Radioi svyaz, 1993.
8. Faysman A. Professeionalnoe programmirovaniya na yazike Paskal- M; Nauka, 1989 g.
9. Aminov I. Paskal dasturlash tili. O'quv qullanma- SamDU, Samarqand, 1996 y.
10. Aminov I.B. Informatika va informatsion texnologiyalar. Ma'ruzalar matni. 2005 yil. SamDU.
11. Aminov I.B., Eshtemirov S., Nomozov F. Informatika va informatsion texnologiyalar fanidan laboratoriya ishlari. Uslubiy qo'llanma. Samarqand, SamDU, 2008 yil.

2-ma'ruza

Mavzu: *Axborotlarni qayta ishlash texnologiyalari.*

Reja:

1. Axborot tushunchasi ,turlari va o'lchov birliklari.
2. Sanoq sistemalari.
3. Sanoq sistemalarida amallar bajarish.
4. Axborotlarning kompyuterda tasvirlanishi.

Darsning maqsadi:

1. Talabalarda axborot tushunchasi, uning turlari to'g'risida umumiy va asosiy ko'nikmalarni hosil qilish.
2. Sanoq sistemalari to'g'risida mukammal tasavvurga ega bo'lish.
3. Sanoq sistemalari va ular ustida amallar bajarish bo'yicha malakalarni shakllantirish.

Tayanch iboralar: *axborot, axborotlarning turlari, axborotning o'lchov birliklari, bit, bayt, kod, ikkilik kod, sanoq sistemasi, kodlashtirish.*

Dars o'tish vositalari: *sinf doskasi, o'quv-uslubiy qo'llanmalar, ma'ruza matnlari, komputer, ma'ruza bo'yicha slaydlar.*

Dars o'tish usuli: *namoyish dasturiy vositalat ishtirokida, jonli muloqotli ma'ruza .*

Darsning xrono xaritasi – 80 minut.

Tashkiliy qism: xonaning tozaligi, jihozlanikishi, sanitariya holati. Talabalarining davomati – 2 minut.

Talabalar bilimini baholash: o‘tilgan mavzuni qisqacha takrorlash, talaba bilan savol-javob o‘tkazish - 10 minut.

Yangi mavzu bayoni - 55 minut.

Mavzuni o‘zlashtirish darajasini aniqlash va mustahkamlash – 10 minut.

Sinov savollari – 5 minut.

Uyga vazifa berish – 3 minut.

Ma‘ruza bayoni

EHM lari bu elektron raqamli qurilma. Elektron qurilma, shuning uchunki – har qanday axborot EHMda elektr signallar yordamida qayta ishlanadi. Raqamli qurilma shuning uchunki, har qanday axborot sonlar yordamida tasvirlanadi.

Sonlarni yozish uchun qandaydir sanoq sistemalaridan foydalanish kerak bo‘ladi.

Sanoq sistemalari. Sanoq sistemasilari deganda sonlarni yozish va tanlash qoidalarining majmuasi tushuniladi. Sanoq sistemalari ikki turga bo‘linadi: pozitsion va pozitsion bo‘lmagan. Pozitsion bo‘lmagan sanoq sistemada son yozuvida ishlatiladigan raqam yozuvda egallagan joyidan qat’iy nazar bir xil qiymatni bildiradi. Rim sanoq sistemasi bunga misol bo‘ladi: I-bir, V-besh, X-o‘n, L-ellik, S-yuz, D-besh yuz, M-ming bildiradi. **Masalan 267 quyidagicha yoziladi:** CCLXVII

Pozitsion sistemada sonni ifodalash uchun ma‘lum miqdordagi raqamlar ishlatiladi. Bu raqamlar sistemaning elementlarini, ularning soni esa sistemaning asosini tashkil qiladi. Har bir raqam ma‘lum bir son qiymatini beradi. Pozitsion sistemada sistema elementlarining ketma ketligi ko‘rinishida ifodalanadi.

Sanoq sistemasida raqamlar soni uning asosi deb yuritiladi.

Kundalik hayotimizda ishlatiladigan sonlar o‘nlik sanoq sistemasida ifodalanadi. Bu sistemasning asosi $R=10$ bo‘lib, son yozuvida 10 ta $\{0,1,2,3,4,5,6,7,8,9\}$ raqam ishlatiladi. Bu sistemada har qanday son o‘nta $0,1,2,3,4,5,6,7,8,9$ raqamlar orqali ifoda qilinadi. (o‘nlik sanoq sistemasi qadimda insonlarning hisoblash ishlarida ikki qo‘lning 10 ta barmog‘idan foydalanish orqali kelib chiqqan).

Bundan tashqari boshqa sanoq sistemalari ham mavjud bo‘lgan.

Masalan. Beshlik- asosi 5ga teng, ya’ni qo‘lning 5 barmog‘i, rim sanoq sistemasi: XXI asr, Qadimgi Vavilonda kishilar asosi 60ga teng bo‘lgan 60lik sanoq sistemasidan foylanishgan. Bu sistema hozirgi vaqtda ham burchak yoki vaqtni o‘lchash uchun ishlatiladi. Masalan. 1 soat 60 minut, 1 minut 60 sekund.

Yil hisoblarini hisoblashda 12 lik sanoq sistemasidan foydalanamiz. Bir yilda 12 oy. Angliyada kun vaqtini yozish ish sistemasi olib boriladi. Yarim kungacha 12 soat va yarim kundan keyin 12 soat.

EHMda sonlarni yozish uchun ikkilik sanoq sistemasi ishlatiladi. Bu sistemada bor yo‘g‘i ikkita: 0 va 1 raqamlari bo‘lib, uning asosi 2 ga teng. Kompyuterda elektr signallarini 0-signal yo‘q va 1- signal bor ko‘rinishida ifodalash mumkin bo‘lganligi uchun, ikkilik sanoq sistemasi ishlatiladi.

Har qanday son raqamlar ketma-ketligi ko‘rinishida yoziladi:

o‘nli sistemada 1985_{10} , $137,85_{10}$

ikkili sistemada 1001_2 , $11001,01_2$

sakkizli sistemada 6534₈, 234, 05₈

Bu yerda sonlarning indeksi uning asosini bildiradi.

Sonlarni yozishda har raqamning qiymati uning turgan o'rniga bog'liq bo'ladi. Sonda raqam uchun joy razryad, sondagi raqamlar soni esa sonning razryadi deb ataladi.

Kompyuterda sonlar ikkilik, sakkizlik, o'n oltilik sistemalarda ham ifodalanishi mumkin.

Ikkilik sanoq sistemasi. Ikkilik sistema ham o'nlik sistema kabi pozision sistema bo'lib, unda sonlar faqat ikkita **1** va **0** yordami bilan ifodalanadi. Natural sonlar qatorining oldingi o'ttizta sonini ikkilik va sakkizlik va o'nlik sanoq sistemasida ifodasini yozamiz.

O'nlik sonlar	Ikkilik sonlar	Sakkizlik sonlar	O'nlik sonlar	Ikkilik sonlar	Sakkizlik sonlar
1	1	1	16	10000	20
2	10	2	17	10001	21
3	11	3	18	10010	22
4	100	4	19	10011	23
5	101	5	20	10100	24
6	110	6	21	10101	25
7	111	7	22	10110	26
8	1000	10	23	10111	27
9	1001	11	24	11000	30
10	1010	12	25	11001	31
11	1011	13	26	11010	32
12	1100	14	27	11011	33
13	1101	15	28	11100	34
14	1110	16	29	11101	35
15	1111	17	30	11110	36

Ikkilik sonlar ustida **qo'shish, ayirish, ko'paytirish va bo'lish** amallarini bajarish mumkin.

$$0 + 0 = 0$$

$$0 + 1 = 1$$

$$1 + 0 = 1$$

$$1 + 1 = 10$$

Misol. 111110₂ va 101₂ sonlarning yig'indisini toping. Bu sonlarni bir ustunga yozib, umumiy qoida bo'yicha qo'shamiz.

$$\begin{array}{r} 111110_2 \\ + \quad 101_2 \\ \hline \end{array}$$

$$1000011_2$$

Ayirish amali uchun quyidagi jadvaldan foydalaniladi.

$$0 - 0 = 0$$

$$0 - 1 = 1$$

$$1 - 0 = 1$$

$$10 - 1 = 1$$

Misol. $1011,101_2$ va $101,01_2$ sonlarining ayirmasini toping.

$$\begin{array}{r} 1011,101_2 \\ - 101,01_2 \\ \hline 110,011_2 \end{array}$$

Ikkilik sonlarni ko'paytirish jadvali quyidagicha.

$$0 \times 0 = 0$$

$$1 \times 0 = 0$$

$$0 \times 1 = 0$$

$$1 \times 1 = 1$$

Misol. 1010_2 va 11_2 sonlarning ko'paytmasini toping.

$$\begin{array}{r} 1010_2 \\ \times 11_2 \\ \hline 1010_2 \\ + 1010_2 \\ \hline 11110_2 \end{array}$$

Ikkilik sonlarini bo'lish amalini bajarishda ko'paytirish va ayirish jadvalidan foydalaniladi.

Sakkizlik sanoq sistemasining asosi 8 ga bo'lib, undagi 0,1,3,4,5,6,7 raqamlari orqali ifoda qilinadi. Qo'shish, ayirish, ko'paytirish va bo'lish amallari 8 lik sanoq sistemi qoidalari asosida bajariladi.

Masalan.

$$\begin{array}{r} 1) \ 732_8 \\ + 324_8 \\ \hline 1256_8 \end{array}$$

$$\begin{array}{r} 2) \ 732_8 \\ - 324_8 \\ \hline 406_8 \end{array}$$

O'n oltilik sanoq sistemasining asosi 16 ga teng bo'lib, undagi sonlar 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, V, S, D, E, F raqamlari orqali ifodalanadi. Bunda A- o'nni, V - o'n birni, S- o'n ikkini, D- o'n uchni, Ye- o'n to'rtini, F- o'n beshni bildiradi. Sonlarni bir sistemadan ikkinchi sistemaga o'tkazish qoidalari mavjud:

1. Ixtiyoriy sistemadan o'nlik sistemaga o'tkazish. Buning uchun son berilgan sistema asosining darajalari bo'yicha yoyiladi va yoyilma hisoblanadi:

Masalan, o'nli sistemadagi 2001 soni to'rt razryadli hisoblanadi. Razryadlar chapdan o'ngga qarab nomerlanadi va har bir razryadga asos darajasi mos keladi.

Razryad son	3	2	1	0	razryad nomi	asos darajasi
	2	0	0	1		
					birliklar:	10^0
					o'nliklar:	10^1

$\longrightarrow$ yuzliklar: 10^2
 $\longrightarrow$ mingliklar: 10^3

Har qanday sonni asos darajasi bo'yicha yoyib chiqish mumkin.

Masalan: a) $200110 = 2 \cdot 10^3 + 0 \cdot 10^2 + 0 \cdot 10^1 + 1 \cdot 10^0$

b) $389710 = 3 \cdot 10^3 + 8 \cdot 10^2 + 9 \cdot 10^1 + 7 \cdot 10^0$

Xuddi shunday ixtiyoriy sanoq sistemasidagi sonlarni o'nli sanoq sistemasida ifodalash mumkin. **Masalan,**

a) $100111_2 = 1 \cdot 2^5 + 0 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 32 + 0 + 0 + 4 + 2 + 1 = 39_{10}$

b) $11011,0112 = 1 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 + 0 \cdot 2^{-1} + 1 \cdot 2^{-2} + 1 \cdot 2^{-3} =$
 $= 16 + 8 + 2 + 1 + 0,25 + 0,165 = 27,32_{10}$

v) $3512_8 = 3 \cdot 8^3 + 5 \cdot 8^2 + 1 \cdot 8^1 + 2 \cdot 8^0 = 586_{10}$

g) $213,1_8 = 2 \cdot 8^2 + 1 \cdot 8^1 + 3 \cdot 8^0 + 1 \cdot 8^{-1} = 75,165_{10}$

d) $AVS_{16} = 1016^2 + 11 \cdot 16^1 + 12 \cdot 16^0 = 2560 + 176 + 12 = 2748_{10}$

Yuqorida keltirilgan amalga teskari amalni ham, ya'ni o'nli sanoq sistemasidagi sonlarni boshqa sanoq sistemalarida ham ifodalash mumkin.

2. O'nlik sistemadagi sonni R asosli sistemaga o'tkazish.

a) Butun sonni o'tkazish uchun berilgan son o'tkazilishi kerak bo'lgan sistema asosi R ga qoldiqli bo'linadi. Bo'linma nolga teng bo'lmasa, u yana asosga qoldiqli bo'linadi va h.k. jarayon bo'linma nolga teng bo'lganda tugatiladi va hosil bo'lgan qoldiqlar teskari tartibi sonning R-lik sistemadagi ifodasini beradi.

Masalan.

1. 437_{10} sonini ikkilik sistemada yozing:

son	Bo'luvchi	qoldiq
437	2	1
218	2	0
109	2	1
54	2	0
27	2	1
13	2	1
6	2	0
3	2	1
1	2	1

Agar qoldiqlarni teskari tartibda yozib chiqsak, kerakli natija hosil bo'ladi: $437_{10} = 110110101_2$

2. 746510_{10} sonini sakkizlik sistemada ifodalang:

son	Bo'luvchi	qoldiq
7465	8	1
933	8	5
116	8	4
14	8	6
1	8	0

Natija: $746510_{10} = 6451_8$

3. 98653_{10} sonini 16-lik sistemada ifodalang:

son	Bo'luvchi	qoldiq
98653	16	13

6165	16	5
385	16	1
24	16	8
1	16	0

Natija : $98653_{10} = 815D_{16}$

Misollar:

1) 25_{10} ni ikkilik sistemaga o'tkazing:

$$\begin{array}{r}
 25 \overline{) 2} \\
 \underline{24} \quad 12 \overline{) 2} \\
 1 \quad \underline{12} \quad 6 \overline{) 2} \\
 \quad \quad 0 \quad \underline{6} \quad 3 \overline{) 2} \\
 \quad \quad \quad 0 \quad \underline{2} \quad 1 \\
 \quad \quad \quad \quad 1
 \end{array}
 \qquad 25_{10} = 11001_2$$

2) 25_{10} ni sakkizlik sistemaga o'tkazing:

$$\begin{array}{r}
 25 \overline{) 8} \\
 \underline{24} \quad 3 \overline{) 8} \\
 1 \quad \underline{0} \\
 \quad \quad 3
 \end{array}
 \qquad 25_{10} = 31_8$$

3) 28_{10} ni o'n oltilik sistemaga o'tkazing:

$$\begin{array}{r}
 28 \overline{) 16} \\
 \underline{16} \quad 1 \overline{) 16} \\
 12 \quad \underline{0}
 \end{array}
 \qquad 28_{10} = 1S_{16}$$

1

b) *To'g'ri kasrni o'tkazish uchun u asos P ga ko'paytiriladi.* Keyin hosil bo'lgan sonning butun qismi ajratiladi va kasr qismi yana R ga ko'paytiriladi. Bu jarayon ko'paytmaning kasr qismi nolga teng bo'lganda yoki kerakligicha raqamlar hosil qilinganda to'xtatiladi va butun qismlar ketma- ketligi kasrning R- ning sistemadagi yozuvini beradi.

Misollar:

1) $0,3125_{10}$ sonini ikkilik sistemaga o'tkazing.

$$\begin{array}{r}
 0 \mid 3125 \\
 0 \mid 6250 \\
 1 \mid 250 \\
 0 \mid 50 \\
 1 \mid 0 \\
 2
 \end{array}
 \qquad 0,3125_{10} = 0,0101_2$$

2) $0,12_{10}$ sonini ikkilik sistemaga o'tkazing.

$$\begin{array}{r}
 0 \mid 12 \\
 0 \mid 24
 \end{array}
 \qquad 0,12_{10} = 0,000111 \dots_2$$

0 48

0 96

1 92

1 84

1 68

3) $0,225_{10}$ sonini sakkizlik sistemaga o'tkazing.

0|225

1|800

6|400

3|200

1|600

4|8

$0,225_{10} = 0,16214..._8$

v) *Aralash sonni o'tkazish uchun butun qismi va kasr qismi alohida alohida o'tkaziladi.*

Misol: $25,3125_{10}$ ikkilik sistemaga o'tkazing.

$25_{10} = 1101_2$ $0,3125_{10} = 0,0101_2$ $25,3125_{10} = 11001,0101_2$

g) *R-lik sistemadan –g- sistemaga o'tkazish uchun avval R-lik son 10-likka, keyin 10 likdan –g-likka o'tkaziladi.*

Misol: $11001_2 = 25_{10} = 31_8$

$34_8 = 28_{10} = 10_{16}$

Kompyuterlarda axborotlarni tasvirlanishi. Har qanday axborotni katta bo'lmagan oddiy qismlarga bo'laklarga bo'lish mumkin. Masalan, matn harflardan va belgilardan, son raqamlar ketma-ketligidan iborat bo'ladi. Harf-bu matnli axborotlarning, raqam esa – sonlarning elementlar bo'ligidir.

Axborot kodlangan deyiladi, agar uning ixtiyoriy bo'ligi sonlar ko'rinishida ifodalangan bo'lsa. Bunday sonlar kodlar deb yuritiladi. Matnning har bir harfini kodlarga almatirish orqali kodlash mumkin bo'ladi.

Kompyuter faqat kodlangan axborotlarni qayta ishlaydi. Axborotlar bilan ishlanganda uning kompyuter xotirada egallangan hajmini bilish kerak bo'ladi. Buning uchun o'lchov birliklaridan foydalaniladi.

Axborotlarning eng kichik o'lchov birligi **bit** hisoblanadi. Bir bit axborot- bu ikkilik raqam **0 yoki 1** dan iborat bo'ladi. **Bit** - axborotlarning eng kichik o'lchov birligidir. Shuning uchun kompyuterda axborotlarni elementar qismlarini qayta ishlash uchun kattaroq o'lchov birligi – **bayt** ishlatiladi. *Bir bayt* bu sakkiz razryadli kod orqalibo'lib, u yordamida qanday kattalikdagi butun sonni yozish mumkin degan savol hosil bo'lishi mumkin.

Bu son 8 bitning hammasi 1 ga teng bo'lgan sonidir. Uning o'nli sanoq sistemasidagi qiymatini aniqlash uchun, unga 1 sonini ham qo'shamiz ham ayiramiz.

$11111111_2 = (11111111_2 + 1) - 1 = 100000000_2 - 1 = 2^8 - 1 = 255$

Bundan ko'rinib turibdiki, 1 baytda **0 dan 255** tagacha bo'lgan **256** ta butun sonlardan bittasini saqlash mumkin ekan. Lotin alifbosi harflarini kodlash uchun **52 ta** son kerak bo'ladi. (**26 ta bosh va 26 ta kichik** harflar). Rus alifbosi harflarini kodlash uchun 66 ta son, unli raqamlarni kodlash uchun yana **10 ta** son kerak bo'ladi. Shunday qilib, **1 bayt** orqali lotin, rus alifbosi harflari, o'nli raqam-

lardan tashqari yana **128 ta** ($256-52-66-10=128$ *) bosh belgilarni (**nuqta, vergul, arifmetik amallar va boshqalar**) ham kodlash mumkin ekan.

1 bayt= 1 belgi.

Bir bayt – bu faqatgina axborotlarning o'lchov birligi emas balki kompyuter xotirasining elementar yacheykasi hamdir. Kompyuterlar xotirasi ana shunday yacheykalar ketma-ketligidan tuzilgan. Har bir yacheyka, ya'ni bayt o'zining adresiga ega. Adres bu yacheyka nomeridan va unda saqlanayotgan ikkilik koddan iborat bo'ladi. Prosessor axborotlarni qayta ishlayotganda xotiradan kerakli yacheykani nomer bo'yicha topadi, unda saqlanayotgan axborotni o'qiydi, kerakli amallarni bajaradi va hosil bo'lgan natijasi boshqa yacheykaga saqlaydi.

Kompyuter xotirasining hajmi baytlarda o'lchanadi, lekin buning uchun ko'p hollarda boshqa o'lchov birliklari ham ishlatiladi: **kilobayt (Kbayt) va megabayt (Mbayt)**.

1 K bayt= 2^{10} bayt=1024 bayt

1Mbayt= 2^{10} Kbayt=1024 Kbayt

Odatda yaxshi kompyuterlarning tezkor xotirasining hajmi **512 Kbayt yoki 640 Kbaytni** tashkil qiladi. Shu o'rinda yaxshi kompyuter uchun bu ko'pmi yoki ozmi degan savol tug'ilishi tabiiy. Bitta sahifaga taxminan **50 ta satr** joylashadi. Shuning uchun bitta sahifa taxminan $60 \times 50 = 300$ belgidan iborat yoki **3 Kbayt** axborotdan iborat bo'ladi. Demak kompyuterning tezkor xotirasiga $640:3=200$ varaqdan iborat matnni sig'dirish mumkin ekan. Quvvati yuqori kompyuterlarning tezkor xotirasi bir necha Mbaytgacha hajmga ega bo'lishi mumkin.

Biz yuqorida ta'kidlanimizdek bitta belgidan iborat bo'lgan axborot kompyuter xotirasining **1 baytida** saqlanadi.

Kompyuter sonli axborotlarni qanda tartibda qayta ishlashi, unda foydalanilayotgan prosessorga bog'liq bo'ladi.

Har bir prosessor ma'lum bir uzunlikdagi ikkilik kodlarni qayta ishlash uchun mo'ljallangan bo'ladi. Eng birinchi yaratilgan shaxsiy kompyuterlarda bir buyruq bilan faqat bir bayt axborotlarni qayta ishlovi prosessorlar qo'llanilgan. Bu kompyuterlar sakkiz razryadli hisoblangan. Ko'pchilik zamonaviy kompyuterlar **16 razryadli**dir. Bu kompyuterlarda bitta buyruq bilan **2 bayt** axborotni qayta ishlash mumkin.

16 –razryadli kompyuterlarda butun sonlarni saqlash va qayta ishlash uchun xotiraning **2 bayti** ishlatiladi. Bunday kompyuterlar qanday butun sonlarni qayta ishlashi mumkin degan savol hosil bo'lishi tabiiy. Bizga ma'lumki, butun sonlar musbat va manfiy bo'lishi mumkin. Shuning uchun son shiorasini qanday kodlash mumkin? Buning uchun **16 bitdan** bittasi, ya'ni chap tomondagi bit ishlatiladi. Agar u **0** ga teng bo'lsa, demak on ishorasi musbat, agar u **1** ga teng bo'lsa, u holda son manfiy bo'ladi. Quyida butun sonlarni saqlash sxemasi keltirilgan.

2 bayt=16 bit			
1	bit	son	15 bit sonning absolyut qiymat uchun
ishorasi uchun			

Agar sxema bo'yicha **15 bitning** hammasi **1ga** teng **7 bo'lsa**, u holda butun son eng katta absalyut qiymatiga ega bo'ladi:

$$(111\ 1111\ 1111\ 1111_2+1)-1=1000\ 0000\ 0000\ 0000_2=2^{15}-1=32767$$

16-razryadli kompyuterlar qayta ishlashi mumkin bo'lgan eng katta butun son **32767** ga teng.

Agar katta miqdordagi butun sonlarni qayta ishlash kerak bo'lsa, u holda ularni saqlash uchun 2 baytdan ziyod xotira lozim bo'ladi. Prosessor bitta buyruq bilan bunday axborotni qayta ishlay olmaydi. Shuning uchun 32767 dan katta sonlarni qayta ishlash uchun maxsus dastur tuzish kerak bo'ladi.

Butun sonlarni qayta ishlash jarayoni bo'yicha aytilgan fikrlar haqiqiy (kasr) sonlari uchun ham tegishli bo'ladi. Odatda 16-razryadli kompyuter kasr sonlarini qayta ishlay olmaydi, buning uchun maxsus dastur ishlatiladi.

Kompyuter xotirasida haqiqiy sonlar odatda 4 baytni egallaydi. Haqiqiy sonning o'zi eksponensial (qo'zg'aluvchan nuqtali) ko'rinishda tasvirlanadi. **Ma-**
salan,

-184.525 soni $-0.184525Ye+3$ ko'rinishida yoziladi. Bu yerda 184525 sonning mantissasi, 3 esa son tartib ($Ye+3$ yozuv 10^3 ni anglatadi).

4 baytdan iborat yacheykada sonning mantissasi ishorasi bilan tartibi ishorasi bilan saqlanishi lozim. Haqiqiy sonlarning saqlanilish sxemasi quyida berilgan.

4 bayt=32 bit	
7 bit tartib uchun	25 bit son mantissasi uchun

Bu sxema bo'yicha son eng katta absalyut miqdori $2^6-1=63$ ga teng, mantissasi eng katta qiymati $2^{24}-1=16777215$ ga teng. Bundan ko'rinib turibdiki, mantissa 8 tadan ortiq o'nli raqamli haqiqiy sondan iborat bo'la olmaydi. Kompyuter hisoblash jarayonida mantissadagi ortiqcha raqamlarni tashlab yuboradi. Shuning uchun haqiqiy sonlar bilan bajariladigan barcha hisoblashlar har doim taqriban hisoblanadi.

Sinov savollar.

1. Sanoq sistemasi nima? Uning qanday turlarini bilasiz?
2. Sanoq sistemasi ustida qanday amallarni bajarish mumkin?
3. Bir sanoq sistemasidan boshqasiga qanday o'tiladi?
4. Qanday axborot kodlangan deyiladi?
5. Axborotlarni eng kichik o'lchov birligi nima? Boshqa o'lchov birliklarini ham bilasizmi?
6. Kompyuter xotirasida butun sonlar qanday saqlanadi? Haqiqiy sonlar-chi?
7. Nima uchun haqiqiy sonlar ustida amallar taqriban bajariladi.

Uyga vazifa : Axborotlarni kodlashga va sanoq sistemalariga doir misollar keltirish va ularni tushuntirib berish.

II-bob. Informatika va informasion texnologiyalarning asosiy texnik va dasturiy vositalari.

3-ma'ruza

Mavzu: Axborot va kompyuter texnologiyalarining asosiy vositalari. Zamonaviy kompyuterlar, ularning asosiy xarakteristikalar, umumiy tuzilishi va foydalanish sohalari

Reja:

1. Kompyuterlar haqida umumiy ma'lumotlar.

2. Kompyuterlarning umumiy tuzilishi.
3. Kompyuterlarning asosiy qurilmalari va vazifalari.
4. Kompyuterlarning foydalanish sohalari.

Darsning maqsadi:

1. Talabalarda komputer va uning turlari to'g'risida umumiy va asosiy bilimlarni hosil qilish.
2. Kompyuterlarning asosiy qurilmalari va vazifalari to'g'risida mukammal tasavvurga ega bo'lish.
3. Kompyuterlarning foydalanish sohalari bo'yicha bilimlarni shakllantirish.

Tayanch iboralar: *kompyuter, shaxsiy kompyuter, prosessor, displey, klaviatura, xotira, printer, disk yurituvchi, disk, yupqa magnit disk, vinchester.*

Dars o'tish vositalari: *sinf doskasi, o'quv-uslubiy qo'llanmalar, ma'ruza matnlari, komputer, ma'ruza bo'yicha slaydlar.*

Dars o'tish usuli: *namoyish dasturiy vositalat ishtirokida, jonli muloqotli ma'ruza.*

Darsning xrono xaritasi – 80 minut.

Tashkiliy qism: *xonaning tozaligi, jihozlanikishi, sanitariya holati. Talabalarning davomati*– 2 minut.

Talabalar bilimni baholash: *o'tilgan mavzuni qisqacha takrorlash, talaba bilan savol javob o'tkazish* - 10 minut.

Yangi mavzu bayoni - 55 minut.

Mavzuni o'zlashtirish darajasini aniqlash va mustahkamlash – 10 minut.

Sinov savollari – 5 minut.

Uyga vazifa berish – 3 minut.

Ma'ruza bayoni

Hozirgi kunda kompyuterlarning imkoniyati kun sayin benihoya kengayib bormoqda. Ular barcha sohalarda: qishloq xo'jaligida, xalq xo'jaligida, sanoatda, ishlab chiqarishda, fan va texnikada, boshqarishda, o'quv dargohlarida va hokazolarda keng qo'llanilmoqda. Kompyuterlar orqali buxgalteriya hisoblari bajarilmoqda, bank faoliyati va ishlab chiqarish sistemalari boshqarilmoqda, hujjatlar chiroyli qilib chop etilmoqda, rasm va multfilmlar yaratilmoqda, musiqa yaratilib, ijro etilmoqda, banklararo aloqalar elektron pochta yordamida amalga oshirilmoqda, turli xil o'yinlar o'rnatilmoqda, turli predmetlar va chet tillari o'rganilmoqda va hokozalar. **Kompyuter** degani inglizcha **“computer”** so'zidan olingan bo'lib, **“hisoblagich”** degan ma'noni anglatadi.

Kompyuterlarning eng ko'p tarqalgan turlari quyidagilardan iborat:

Katta EHM – bu kompyuterlar katta hajmdagi axborotlarni qayta ishlash uchun mo'ljallangan bo'lib, o'zining ishonchligi, yuqori ishlash tezligiga ega ekanligi bilan boshqa kompyuterlardan ajralib turadi. Ular asosan yirik korxonalarda, xalqaro miqyosdagi banklarda, davlat tashkilotlarida va boshqa joylarda ishlatiladi. Ularga minglab displey va klaviaturalarni ham ulash mumkin.

Super EHM – bu kompyuterlar katta hajmdagi hisoblashlarga ega bo'lgan masalalarni yechish uchun mo'ljallangan. Ular asosan mudofa ishlarida, meteorologiya sohasida, geologiyada, astrologiya sohasida va hokazolarda ishlatiladi.

Mini EHM – bu kompyuterlar ham ma'lum bir sohaga tegishli masalalarni yechish uchun mo'ljallangan. Ular asosan davlat tashkilotlarida oliy o'quv yurtlarida, ma'lumotlarni qayta ishlash markazlarida qo'llaniladi. Ularga o'nlab, yuzlab displey va klaviaturalarni yoki shaxsiy kompyuterlarni qo'shish mumkin.

Shaxsiy kompyuterlar – bu kompyuterlar unchalik katta bo'lmagan hajmga va og'irlikka ega bo'lib, bitta kishi ishlashi uchun mo'ljallangan. Shaxsiy kompyuterga qo'shimcha vosita sifatida: modem, faks-modem, multimedia vositalari, telefon, lokal va global tarmoq vositalarini va boshqa qurilmalarni ulash mumkin. Hozirgi kunda **IBM PC va Pentium** turidagi kompyuterlar eng ko'p tarqalgan shaxsiy kompyuterlar hisoblanadi.

Cho'ntak kompyuterlari – bu kompyuter kichik hajmga va og'irlikka ega bo'lib, oddiy batareykalarda ishlashga mo'ljallangan. Ularda qattiq disk, disk yurituvchi yo'q bo'lib, minityur ko'rinishdagi klaviaturadan ma'lumotlarni kiritish uchun foydalaniladi.

Hozirgi vaqtda eng ko'p ishlab chiqarilayotgan va amaliyotga qo'llanilayotgan zamonaviy kompyuterlar shaxsiy kompyuterlar hisoblanadi.

Zamonaviy kompyuterlar asosan quyidagi asosiy qurilmalardan iborat bo'ladi:

1. **Sistemali blok.**
2. **Monityur**
3. **Klaviatura**
4. **Chop etish qurilmasi – printer.**
5. **Sichqoncha.**

1. **Sistemali blok.** Sistemali blok kompyuterning eng asosiy qurilmasi bo'lib, quyidagilarni o' ichigga oladi:

- **mikroprosessor** – ko'p hollarda prosessor deb yuritiladi. U hisoblashlarni bajaradi va kompyuter ishini boshqaradi;

- **xotira** – ma'lumotlarni va hisoblash natijalarini saqlash uchun ishlatiladi;

- **kiritish chiqarishni nazoratchilar** – bu qurilma displey, magnit disk-lar, sichqoncha va boshqa qurilmalar ishini nazorat qiladi;

- **kiritish-chiqarish portlari** – kompyuter va tashqi qurilmalar orasida ma'lumotlarni almashtirib turish uchun ishlatiladi;

- **vinchester (qattiq magnit disk)** – ma'lumotlarni doimiy saqlash uchun ishlatiladi;

- **yupqa magnit disklar (disketlar)** – ma'lumotlarni doimiy saqlash hamda ularni bir kompyuterdan boshqa kompyuterga o'tkazish uchun ishlatiladi.

Disketlarga vinchesterga nisbatan ancha kam miqdordagi axborotlar joylashadi.

Mikroprosessor. Mikroprosessor shaxsiy kompyuterlarning asosiy qurilmasi hisoblanadi. Bu qurilma asosiy "intelektual" ishlarni bajaradi, arifmetik va mantiqiy hisoblashlarni bajaradi. Shu bilan birga kompyuter ishini nazorat qilib boradi.

IBM PC va Pentium kompyuterlarida asosan Intel Corporation of Santa Klara firmasi tomonidan yaratilgan mikroprosessorlardan foydalaniladi.

Ular zamonaviyligiga qarab, quyidagi turlarga bo'linadi: **086; 286;386; 486; Pentium P5, Pentium-Pro-P6, Pentium-II, Pentium-III, Pentium-IV** va hokozalar. Bu raqamlar Intel firmasi prosessorlarining nomini bildiradi. Raqamlarga qarab prosessorlarning quvvatini ham aniqlash mumkin.

Bundan tashqari "*Epil (Apple)*" firmasi ham "*Makintosh (Macintosh)*" turidagi kompyuterlar ishlab chiqaradi. Bu kompyuterlar qo'shimcha ovoz berish qurilmasiga, mikrofonlar, dinamiklar, modellar bilan jihozlangan. Ayniqsa, hozirgi vaqtda bu firma juda baquvvat makintosh kompyuterlarini ishlab chiqarib, amaliyotga tadbiiq etmoqda. Makintosh kompyuterlari IBM kompyuterlariga qaraganda narxini ancha qimmatligi tufayli unchalik keng tarqalmagan.

Xotira. Kompyuterlarning xotira qurilmasi **ichki va tashqi** xotira qurilmalariga bo'linadi. Ichki xotira qurilmasi o'z navbatida 2 ga bo'linadi: **doimiy (ROM) va tezkor xotira (RAM).**

Doimiy xotiradan kompyuterning barcha imkoniyatlaridan to'liq foydalanish uchun ishlatiladigan dasturlar majmuasini doimiy saqlab turish uchun foydalaniladi. Doimiy xotirada saqlanayotgan ma'lumotlarni o'zgartirib bo'lmaydi.

Tezkor xotira (RAM) kompyuterning eng muhim elementlaridan biri bo'lib, ma'lumotlarni ma'lum vaqtga saqlab turish uchun ishlatiladi. Tezkor xotirada saqlanayotgan ma'lumotlar kompyuter o'chirilganda o'chib ketadi. Tezkor xotirada hisoblashlar juda tez bajariladi.

Kompyuterlarning imkoniyati tezkor xotiraning hajmi bilan belgilanadi. Tezkor xotiraning hajmi qancha katta bo'lsa, kompyuterning imkoniyati ham shunchalik yuqori bo'ladi. Zamonaviy kompyuterlar odatda 16 va 32 Mbayt hajmga ega bo'lgan tezkor xotira bilan ishlab chiqilmoqda.

Tashqi xotira qurilmasi ham ma'lumotlarni doimiy saqlash uchun ishlatiladi. Tashqi xotira qurilmasi sifatida **qattiq disk (NDD) va yupqa magnit disklar** ishlatiladi.

Qattiq disk (NDD). Qattiq disklar (**vinchester**) kompyuter bilan ishlash jarayonida foydalaniladigan ma'lumotlarni doimiy saqlash uchun ishlatiladi. Bu ma'lumotlar operatsion sistema dasturlari, ko'p qo'llaniladigan amaliy dasturlar majmuasi, hujjatlar muharriri, dasturlash tillari translyatorlari va hokazolardan iborat bo'lishi mumkin. Qattiq disklar boshqa xotira qurilmalaridan ishonchliligi; hajmining kattaligi va axborotlarni o'qish tezligining yuqoriligi bilan farq qiladi. Hozirgi vaqtda kompyuterlar **800 Mbaytdan 2,2 Gbaytgacha** va undan ham kattaroq hajmga ega bo'lgan disklar bilan ta'minlanmoqda.

Yupqa magnit disklar. Bu magnit disklar ham ma'lumotlarni saqlash va ularni bir kompyuterdan boshqa bir kompyuterga o'tkazish uchun ishlatiladi. Kompyuterlarda **3,5 va 5,25 dyuma** o'lchovli yupqa magnit disklar ishlatildi:

5,25 dyuym o'lchovli disketlar "**besh dyuym**", **3,5 dyuym** o'lchovli disketlar esa "**uch dyuym**"li disketlar deb yuritiladi. Uch dyuymli disketlarda ma'lumotlar saqlanishini ishonchliligi ancha yuqori. Disketlar asosan hajmi bilan bir-biridan farq qiladi.

Uch dyuymli disketlar **1,4 Mbayt** gacha, besh dyuymli disketlar esa **1,2 Mbaytgacha** hajmga ega bo'lgan axborotlarni saqlash uchun foydalaniladi.

2. Monitor. Monitor (**display**) ekranga sonli, matnli va grafik ma'lumotlarni chiqarish uchun ishlatiladi. Monitor oq-qora va rangli bo'li-shi mumkin. Ular rangi va ekranining o'lchovi bilan bir-biridan farq qiladi. Monitor **matnli va grafik** tartibida ishlashi mumkin.

Matnli tartibida monitor ekrani **25 satr** va har bir satrda **80 ta o'ringa** ajratiladi. Jami o'rinlar soni ixtiyoriy belgini yoki harfni hosil qilish mumkin. Matnli tartibda barcha amallar tez bajariladi.

Grafik tartibda ekranda alohida nuqtalardan iborat bo'lgan tasvir-lar oq-qora yoki rangli ko'rinishda ifodalanadi. Grafik tartibida ekranda matnlarni, rasmlarni, grafiklarni va hokazolarni hosil qilish mumkin.

Hozirgi kunda eng ko'p tarqalgan quyidagi rangli monitorlar ishlatiladi:

EGA – ekranda gorizontaal bo'yicha 640 nuqta va vertikal bo'yicha 350 nuqta hosil qiladi;

VGA –gorizontaal bo'yicha 640 nuqta va vertikal bo'icha 480 nuqta;

SVGA -640×480, 800×600, 1024×768 ta nuqta hosil qiladi.

3. Klaviatura. Klaviatura ma'lumotlarni kiritish va kompyuter ishini boshqarish uchun ishlatiladi. Kompyuterlar odatda **101** tugmachali standart klaviatura bilan ta'minlanadi. Klaviaturani shartli ravishda

4 guruh klavish (tugmacha)lariga bo'lish mumkin:

1. Harfli –raqamli va belgili tugmachalar. (bo'shliq, 0-9 raqam-lari, A-Ya lotin harflari, A-Ya ruscha harflar, +,-,*,/ maxsus belgilar va boshqalar).

2. Funksional tushmachalar: F1, F2, ... , F10.

3. Xizmatchi tugmachalar: Enter, Esc, tab, kursorni boshqarish tugmachalari Shift, Ctrl, Alt, PgUp, PgDn, Home, End va boshqalar.

4. Klaviaturaning yordamchi (o'ng klaviatura) tugmachalari.

Klaviatura tugmachalariga "**Vstavka-Simvol**" bo'limi yordamida ixtiyoriy belgilarni yoki alifbo harflarini o'rnatish mumkin.

4. Chop etish qurilmasi – printer. Printer axborotlarni qog'ozga chop etish uchun mo'ljallangan. Printerlar faqat matnli axborotlarni emas, balki grafiklarni, tasvirlarni, rasmlarni ham chop etish uchun ham ishlatiladi. Printerlarning juda ko'p xillaridan foydalaniladi. Ular rangli va rangsiz chop qilishni ta'minlaydi.

Ko'p ishlatiladigan printerlar **matrisali, struyinli va lazerli** printerlar hisoblanadi.

Matrisali (ignali) printerlar ilgarigi vaqtda eng keng tarqalgan printerlar bo'lib, chop etish sifatining uncha ham yuqori emasligi, ishlash jarayonida shovqin chiqarishi va rangli tasvirda chop etishga qodir emasligi bilan boshqa printerlardan farq qiladi. Matrisa printerlarda chop etish maxsus ignachalar orqali amalga oshiriladi. Ular **9** va **24** ignali bo'lishi mumkin. Ularning narxi ancha arzon bo'lganligi tufayli hozirga qadar qo'llanilib kelinmoqda.

Struyinli printerlar. IBM PC kompyuterlar uchun eng keng tarqalgan printerlar hisoblanadi. Bu printerda axborotlarni chop etishi uchun maxsus buyoqlar ishlatiladi. Struyinli printerlar matrisali printerdan shovqinsiz ishlashi, chop etishning sifati bilan keskin farq qiladi. Shu bilan birga rangli tasvirda chop etish imkoniyatiga ham ega.

Lazerli printerlar oq-qora va rangli tasvirda yuqori sifatli darajada chop qilishni amalga oshiradi. U matrisa va struyinli printerlar-dan chop etishning sifati va tezligi, hamda umuman shovqinsiz ishlashi bilan farq qiladi. Lazerli printerlarda chop etish tezligi matrisali va struyinli printerlarga qaraganda **4-5 baravar** yuqori bo'ladi. Lazerli printerlarda chop etilgan hujjatlarni to'g'ridan to'g'ri bosmaxonalarda ko'paytirishga berish mumkin.

5. Sichqoncha. Sichqoncha zamonaviy kompyuterlarning ajralmas qismi bo'lib qoldi. Ular kompyuterda ishlashni va uning ishini boshqa-rishni ancha yengillashtiradi.

Zamonaviy operasion sistemalar muhiti va DOS ning ko'plab das-turlari bilan ishlash uchun sichqonchadan keng foydalaniladi. Sichqoncha **ikkita yoki uchta** tugmacha bilan ta'minlangan bo'ladi. Sichqonchaning ostida turli yo'nalishlar bo'yicha harakatlantirish uchun **sharikcha** qo'llaniladi. Sichqonchani stol ustida harakatlantirganda unga mos ravishda ekranda **sichqoncha ko'rsatkichi (strelka)** harakat qiladi. Agar kerakli biror amalni bajarish kerak bo'lsa, sichqoncha ko'rsatkichi kerakli joyga keltiriladi va u yoki bu tugmachasi bosiladi.

Sinov savollar

1. Komputerning asosiy qurilmalari va ularning asosiy vazifasi?
2. Xotira qurilmasi va unig turlari?
3. Klaviaturada qanday amallarni bajarish mumkin?
4. Chop etish qurilmalari to'g'risida nimalarni bilasiz?

Uyga vazifa : *Komputerlarning barcha qurilmalarini va ularning ishlash nartibini to'liq o'rganish va tushuntirib berish.*

4-Ma'ruza

Mavzu: Zamonaviy operasion tizimlar to'g'risida umumiy ma'lumotlar.

Faylli sistema.

Reja:

1. Dasturiy ta'minoti va uning tarkibiy qismlari.
2. Operasion sistema va uning turlari.
3. Fayl va katalog tushunchasi.
4. Faylli sistema.

Darsning maqsadi:

1. Talabalarda dasturiy ta'minot va uning tarkibiy qismlari to'g'risida umumiy va asosiy bilimlarni hosil qilish.
2. Operatsion sistema va uning turlari tasavvurga ega bo'lish.
3. Fayl, katalog va faylli sistema tushunchasi, ular ustida amallar bajarish bo'yicha malakalarni shakllantirish.

Tayanch iboralar: *dasturiy ta'minot, operasion sistema, fayl, kata-log, qism katalog, faylli sistema, operasion sistema buyruqlari.*

Dars o'tish vositalari: *sinf doskasi, o'quv-uslubiy qo'llanmalar, ma'ruza matnlari, komputer, ma'ruza bo'yicha slaydlar.*

Dars o'tish usuli: *namoyish dasturiy vositalat ishtirokida, jonli muloqotli ma'ruza .*

Darsning xrono xaritasi – 80 minut.

Tashkiliy qism: xonaning tozaligi, jihozlanilishi, sanitariya holati. Talabalarining davomati – 2 minut.

Talabalar bilimini baholash: o‘tilgan mavzuni qisqacha takrorlash, talaba bilan savol-javob o‘tkazish – 10 minut.

Yangi mavzu bayoni – 55 minut.

Mavzuni o‘zlashtirish darajasini aniqlash va mustahkamlash – 10 minut.

Sinov savollari – 5 minut.

Uyga vazifa berish – 3 minut.

Ma‘ruza bayoni

Hozirgi kunda kompyuterlarning to‘liq imkoniyatlaridan foydalanish uchun bir necha minglab turli xil dasturlar yaratilgan bo‘lib, ular kompyuterning dasturiy ta‘minotini tashkil qiladi. Bu dasturlar bajaradigan funksiyalariga qarab **operasion sistemalar, dasturlash sistemalari va amaliy dasturlar sistemasi** kabi guruhlariga bo‘linadi. Dasturli ta‘minotni eng asosiy qismini operasion sistemalar tashkil qiladi.

Operasion sistema kompyuterning dasturiy ta‘minotiga kiruvchi, uning ishini ta‘minlovchi, masalalarni bajarilishini rejalashtiruvchi, foydalanuvchi bilan kompyuter o‘rtasida muloqot o‘rnatuvchi, uning barcha imkoniyatlarini boshqaruvchi dasturlar tizimidir.

Boshqacha aytganda, kompyuterlarning operasion sistemasi quyidagi **asosiy vazifalarni** bajaradi:

- kompyuterning turli ichki qurilmalari orasida ma‘lumotlar almashinuvchi amalga oshiradi;
- foydalanuvchi yoki sistemali dasturni bajarilishini ta‘minlaydi;
- klaviatura, displey, disk yurituvchi, chop etish va boshqa qurilmalar ishini yengillashtiradi;
- masalalarni tartib bilan ketma-ket bajarilishini rejalashtiradi;
- foydalanuvchi kompyuterdan to‘liq foydalanishi uchun barcha imkoniyatlarini yaratadi;
- operasion sistema mashina tilida yozilgan dasturlar to‘plamidan iborat bo‘lib, uning elementlarini yaratishda avtokod va hatto yuqori darajali dasturlash tillari ishlatilishi mumkin.

Dasturlar yaratilib bo‘lingach, translyasiya qilinishi, ya‘ni kompyuter tiliga o‘tkazilishi kerak. Operasion sistemaning asosiy qismi doimiy xotirada, qolgan qismlari esa tezkor va tashqi xotirada hamda maxsus magnitli disklarda saqlanadi.

Hozirgi kunda kompyuterlar uchun o‘nlab turli xil operasion sistemalar yaratilgan bo‘lib, ular asosan uchta asosiy guruhga bo‘linadi:

1. 8-razryadli mikroprosessorli kompyuterlar uchun SR/M operasion sistemasi. Bu operasion sistemaning yaratilishi kompyuterlar uchun boshqa operasion sistemalarning yaratilishiga asos soldi. SR/M operasion sistemasi 1974 yilda yaratilgan va ko‘plab 8-razryadli mikroprosessorli shaxsiy kompyuterlarga o‘rnatilgan. Uning asosiy yutug‘i juda soddaligida va turli xil kompyuterlar uchun tezroq moslash-tirilishidan iborat.

2. 16-razryadli mikroprosessorli kompyuterlar uchun operasion sistemalar. Hozirgi vaqtda MS DOS operasion sistemasi 16-razryadli mikropro-

sensorli kompyuterlar uchun eng ko'p qo'llaniladigan operasion sistemasi hisoblanadi. U o'zining imkoniyatlarining kengligi bilan SR/M operasion sistemadan farq qiladi. Bu turdagi operasion sistemadan farq qiladi. Bu turdagi operasion sistemalar asosiy ustunligi quyidagidan iborat: **buyruqlar tilining kengayganligi; faylli sistemaning iyerarxik ko'rinishga egaligi; tashqi qurilmalar bilan ishlash imkoniyatini mavjudligi** va hokazo. MS DOS operasion sistemasida ishlash uchun 60 Kbaytga yaqin xotira hajmi kerak bo'ladi.

3. 16-32 razryadli mikroprosessorli kompyuterlar uchun UNIX operasion sistemasi. Bu sistema 70 yillardan boshlab imkoniyatga ega ekanligi uchun 16 va 32 razryadli mikroprosessorli kompyuterlarda qo'llanila boshlandi.

UNIX sistemasi keng imkoniyatli faylli sistema va buyruqlar tili prosessoriga ega. Bu sistema o'zining imkoniyatligi, boshqa kompyuterga tez va samarali sozlanishi hamda foydalanuvchi bilan muloqotni soddaligi bilan farq qiladi.

Bundan tashqari, yuqorida sanab o'tilgan operasion sistemalar asosida aniq kompyuterga va ma'lum bir sohada qo'llanish uchun ko'plab boshqa operasion sistemalar ham yaratilmoqda. Bu sistemalar MSX DOS, mikro DOS, CP/M-80, PAFOS forth, mikros -80, XENIX, OC DBK va boshqa shu kabi sistemalarni keltirish mumkin.

Operasion sistemalarni qo'llanish sohalarini quyidagicha berish mumkin:

Qo'llanish sohasi	Kompyuter		
	8-razryadli	16-razryadli	32-razryadli
<i>O'quv jarayoni</i>	CP/M-80, Mikro DOS, Foth, MS DOS	MS DOS, DAFOS, INFOS, OSDVK	UNIX, MNOS INFOS
<i>Iqtisodiy-rejalashtirish hisoblashlar</i>	CP/M-80, Mikros-80	MS DOS, RAFOS, Mikros-86	UNIX, XENIX CP/M-68 K
<i>Injenerlik hisoblashlar</i>	CP/M-80, Mikros-80	MS DOS, INFOS, RAFOS, OSDVK	UNIX, DEMOS
<i>Ilmiy izlanishlar</i>	Maxsus operasion sistemasi	INFOS, DEMOS, MS DOS	UNIX, DEMOS

Operasion sistemalar quyidagi asosiy komponentalardan iborat bo'ladi: **fayllar sistemasi, tashqi qurilmalar drayveri va buyruqlar tili prosessori.**

Fayllar sistemasi. Fayl kompyuterlar xotira qurilmasida saqlanadigan ma'lumotlarning turi. Har bir fayl biror ko'rinishdagi ma'lumotni saqlash uchun ishlatiladi. Fayl-biror dasturdan, matndan, hujjatdan, grafik tasvirdan va hokazolardan iborat bo'lishi mumkin.

Kompyuter xotirasida saqlanayotgan fayllar **fayllar sistemasini** tashkil qiladi. **Fayllar sistemasi** kompyuterlarning operasion sistemasining eng asosiy tarkibiy qismi bo'lib, foydalanuvchilar kompyuter bilan muloqot qilishi, kerakli axborotlarni izlab topishi va samarali ma'lumotlar bazasini yaratish uchun qulay imkoniyatlarni yaratadi.

Fayllar operasion sistema tomonidan xotira qurilmalarida ajratilgan maxsus joylarda saqlanadi. Har bir fayl foydalanuvchi tomonidan berilgan o'z nomiga ega bo'ladi.

Fayl nomi 2 qismdan iborat bo'ladi, ya'ni **fayl nomi va kengaytmasi**. Faylni belgilash uchun unga **nom** va uning turini bildiruvchi **kengaytma** beriladi. **Nom** sakkiztagacha lotin harflaridan, belgilardan va raqamlardan iborat bo'lishi mumkin. **Kengaytma** esa uchtagacha belgidan iborat bo'ladi va u **fayl turini** ko'rsatadi. Nom va kengaytma bir-biridan **(.) nuqta** bilan ajratiladi. Nom kengaytmasi mavjud fayllar turidan, shu bilan birga ixtiyoriy nom kengaytmasidan ham iborat bo'lishi mumkin.

Operasion sistemalarda ishlatiladigan nom kengaytmalari quyidagicha:

- **.txt** - matnli fayllar;
- **.doc** - hujjatli fayllar;
- **.bas** - Beysik tilidagi dastur;
- **.c** - SI tilidagi dastur;
- **.pas** - Paskal tilidagi dastur;
- **.exe** - bajariluvchi fayl;
- **.com** - bajariluvchi buyruqli fayl;
- **.bat** - buyruqli fayl;
- **.bak** - matnli fayl nusxasi;
- **.dat** - ma'lumotli fayl;
- **.dbf** - ma'lumotlar bazasining tezkor fayli.
- **.xls** – Excel dasturidagi fayl.

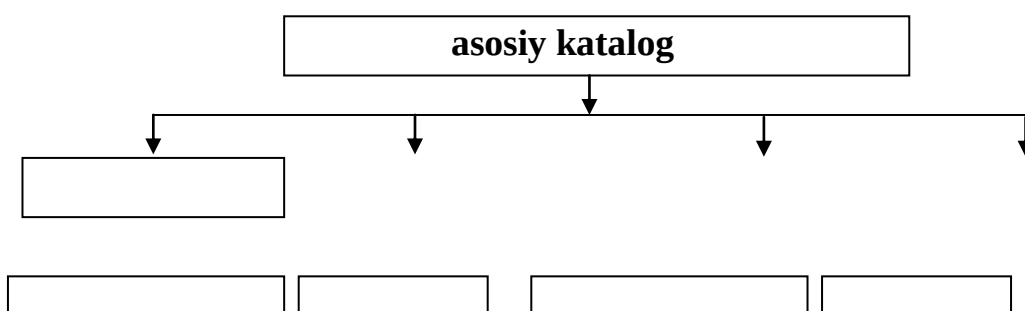
Xotira qurilmasida saqlanayotgan barcha fayllar ro'yxati **katalog**da joylashgan bo'ladi.

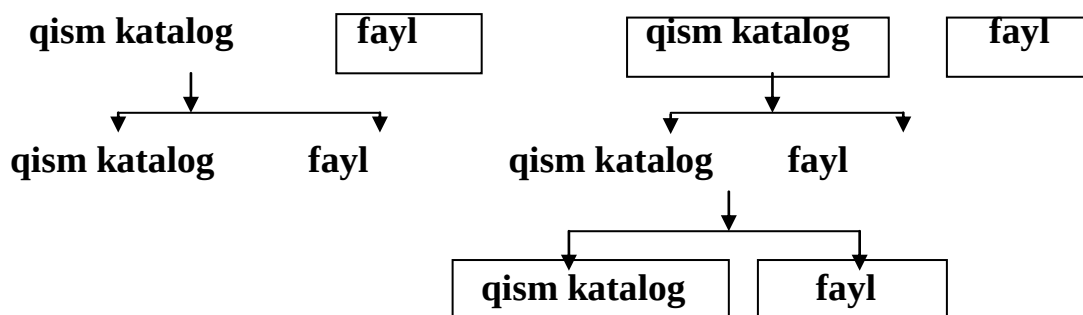
Katalog – yunoncha **katologos** - so'zidan olingan bo'lib, **ro'yxat** degan ma'noni bildiradi. **Katalog** –fayl nomlari va turlari joylashgan xotira qurilmasidagi maxsus joy.

Har bir katalogda bir necha fayllar joylashishi mumkin, ammo diskdagi fayllarning har biri yagona katalogda qayd etilgan bo'ladi.

Kataloglar fayllar haqidagi axborotlarni saqlash uchungina emas, balki diskdagi yuzlab, ba'zan esa minglab fayllarni tartib bilan saqlash, saralash, ularni izlash va foydalanishda qulaylik yaratish uchun ham zarurdir. Kataloglar o'z nomiga ega bo'ladi. **Katalog nomi** 8 tagacha belgidan iborat bo'lishi mumkin. Kataloglar ichma-ich joylashgan bo'lishi mumkin. Agar bir katalog boshqa bir katalog ichida joylashgan bo'lsa, boshqa katalog ichida joylashgan katalog ichida joylashgan katalog unga nisbatan **qism katalog** deb yuritiladi. Bunday holda faylli sistema **iyerarxik** ko'rinishga ega bo'ladi, ya'ni turli darajadagi kataloglar saqlanadi.

Diskdagi barcha katalog va fayllar ro'yxati saqlanadigan katalog **asosiy yoki ona kataloglar** deb yuritiladi. Unda birinchi darajali kataloglar va alohida fayllar saqlanadi. Birinchi darajali katalogda esa ikkinchi darajali katalog va alohida fayllar saqlanishi mumkin. Har bir katalog darajalari bo'yicha joylashgan kataloglar uchun **ona katalog** bo'lishi mumkin:





Bitta katalogda bir xil nomdagi bitta fayl saqlanishi mumkin. Lekin, bir xil nomdagi fayllar turli xil darajadagi kataloglarda saqlanilishi mumkin.

Tashqi qurilmalar drayveri. Qurilmalar drayveri – kompyuter-ning tashqi qurilmalar bilan aloqasini ta'minlaydi. **Drayver** so'zi inglizcha so'zdan olingan bo'lib "**driver**" - **haydovchi yoki boshqaruvchi** degan ma'noni bildiradi.

Kompyuterlar katta hajmdagi tashqi qurilmalarga ega bo'ladi. Bular **display, klaviatura, disk yurituvchi, disklar va chop etish qurilma-laridir**. Bundan tashqari kompyuterga qo'shimcha kiritish-chiqarish qurilmalari: **sichqoncha, telefon, manipulyator, modem, lokal va global tarmoq, skaner** va boshqalarni ulash mumkin

Tashqi qurilmalar to'plami ishini to'liq ta'minlash opearasion sistemaning asosiy funksiyalaridan biridir. Bu funktsiyani bajarish uchun tashqi qurilmalarni boshqarish dasturlari – **drayverlar** ishlatiladi.

Standart tashqi qurilmalar drayveri kiritish-chiqarishning bazaviy sistemasini hosil qiladi va kompyuterning doimiy xotira qurilmasiga joylashgan bo'ladi. Kompyuter ishga tushirilgan paytda, bu qurilmalar kompyuterga qo'shilgan yoki qo'shilmaganligidan qat'iy nazar, ular kompyuter xotirasiga yuklanadi. Qo'shimcha qurilmalar drayverlari esa o'zlarining doimiy xotirasida saqlanadi va ular kompyuterga qo'shilgandan keyin ishga tushiriladi.

Buyruqlar tili prosessori. Har qanday operasion sistema o'zining buyruqlar tiliga ega bo'lib, ular yordamida foydalanuvchi u yoki bu turdagi amallarni ifoda qiladi. Foydalanuvchi buyruqlar tilida tuzilgan dasturini tahlil qilishni va bajarishni uchun operasion sistema tarkibiga kirgan maxsus dasturlar – buyruqlar tili prosessori orqali amalga oshiradi.

Buyruqlar tili prosessori – foydalanuvchi bilan kompyuter o'rtasida mu-loqot o'rnatuvchi maxsus dasturdir. Buyruqlar tili prosessori alohida buyruqlardan tashqari, foydalanuvchi tomonidan ilgari tuzilgan ancha murakkab ko'rinishdagi buyruqlar ketma-ketligini ham tahlil qiladi va bajaradi.

Sinov savollar

1. Operasion sistema nima va uni asosiy tashkil etuvchilari?
2. Operasion sistemasining asosiy vazifasi va imkoniyatlari?
3. Operasion sistemasining asosiy buyruqlari qaysilari?
4. Fayllar va katalog nima? Fayl nomi? Qism katalog?

Uyga vazifa : *Operatsion sistema va unig tashkil etuvchilarini mukammal o'rganib olish, fayl va kataloglarga doir misollar keltirish, ularni tushuntirib berish.*

III-bob. Algoritmash va dasturlash asoslari.

8-ma'ruza

Mavzu: Masalalarni kompyuterda yechish bosqichlari. Algoritm tushun-chasi, xossalari, berilish usullari va turlari. Chiziqli algoritmlari.

Reja:

1. Masalalarni kompyuterda yechish bosqichlari
2. Algoritm tushunchasi va uning xossalari.
3. Algoritmning berilishi usullari.
4. Algoritmning turlari. Chiziqli algoritm.

Darsning maqsadi:

1. Talabalarda axborotlashtirish jarayoni to'g'risida umumiy va asosiy bilimlarni hosil qilish.
2. Algoritm va algoritmning berilishi usullarito'g'risida mukammal tasavvurga ega bo'lish.
3. Algoritmning turlari va chiziqli algoritmlar tuzish bo'yicha ko'nikmalarni shakllantirish.

Tayanch iboralar: *algoritm, xossalari, berilish usullari, turlari, blok-sxema, chiziqli algoritm.*

Dars o'tish vositalari: *sinf doskasi, o'quv-uslubiy qo'llanmalar, ma'ruza matnlari, kompyuter, ma'ruza bo'yicha slaydlar.*

Dars o'tish usuli: *namoyish dasturiy vositalat ishtirokida, jonli muloqotli ma'ruza .*

Darsning xrono xaritasi – 80 minut.

Tashkiliy qism: *xonaning tozaligi, jihozlanikishi, sanitariya holati. Talabalarning davomati*– 2 minut.

Talabalar bilimni baholash: *o'tilgan mavzuni qisqacha takrorlash, talabala bilan savol javob o'tkazish* - 10 minut.

Yangi mavzu bayoni - 55 minut.

Mavzuni o'zlashtirish darajasini aniqlash va mustahkamlash – 10 minut.

Sinov savollari – 5 minut.

Uyga vazifa berish – 3 minut.

Ma'ruza bayoni

Masalalarni kompyuterda yechish bosqichlari. Har qanday masalani kompyuterga tayyorlash va uni o'tkazish quyidagi bosqichlarda amalga oshiriladi.

1. Masalani qo'yilishini aniqlash va matematik modelini ishlab chiqish.
2. Masalani yechishning sonli usulini tanlash.
3. Masalani yechish algoritmini ishlab chiqish.
4. Kompyuter uchun dastur tuzish.
5. Dasturni kompyuter xotirasiga kiritish, rostlash va tekshirish.
6. Hisoblash natijalarini qayta ishlash va tahlil qilish.

1. Masalaning qo'yilishini aniqlash va matematik modelini ishlab chiqish. Masalani yechishdan oldin uning qo'yilishi oydinlashtiriladi, ya'ni bunda uning maqsadi va yechilish shartlari aniqlanadi, boshlang'ich ma'lumotlar va natijalarning tarkibi asoslanadi. Bu ma'lumotlar asosida u matematik formulalar ko'rinishida ifoda qilinadi.

2. Masalani yechishning sonli usulini tanlash. Qo'yilgan matematik masalalar uchun ushning sonli yechish usulini tanlash kerak bo'ladi. Sonli usullar turli-tuman bo'lganligidan ularning eng samarali va qulayini tanlash kerak. Bu masala bilan matematikaning sonli usullar bo'limi shug'ullanadi. Yechish usulini tanlash masalaga qo'yilgan barcha talablarni va uni konkret kompyuterlarda hal qilish imkoniyatlarini hisobga olish kerak.

3. Masalani yechish algoritmini ishlab chiqish. Masalani yechish uchun tanlangan sonli usulning algoritmi ishlab chiqiladi, ya'ni masalani yechish uchun bajariladigan arifmetik va mantiqiy amallar ketma- ketligi yoritiladi. Masalani yechish algoritmlari ko'rgazmalibroq bo'lishi uchun, ular ko'p hollarda blok- sxema ko'rinishida ifodalanadi.

4. Kompyuter uchun dastur tuzish. Kompyuter uchun dastur masalaning umumiy yechimidir. U algoritmnining mashina buyruqlari ketma- ketligi shaklidagi yozuvdir. Buning uchun dasturlash tillari (Besk, Fortran, Paskal, SI va boshqalar) dan biri tanlanadi va unga mos dastur tuziladi. Tuzilgan dasturni sifatli bo'lishi va uni mashina xotirasidan kam joyni egallashi muhim ahamiyatga ega.

5. Dasturni kompyuter xotirasiga iritish, rostlash va tekshirish. Tuzilgan dastur kompyuter klaviaturasi orqali uning xotirasiga kiritiladi. Kiritilgan dasturni rostlash va tekshirish amalga oshiriladi, ya'ni yo'l qo'yilgan xatoliklar tuzatiladi.

6. Hisoblash natijalarini qayta ishlash va tahlil qilish. Bu bosqichdan tuzilgan dastur bo'yicha hisoblash bajariladi va hosil bo'lgan natija kompyuterning displey ekraniga chiqariladi yoki chop etish qurilmasi orqali qog'ozga chop etiladi. Natijalarni jadvallar, grafiklar yoki diagrammalar ko'rinishida hosil qilish mumkin. Hosil bo'lgan natija esa foydalanuvchi tomonidan tahlil qilinadi.

Algoritm tushunchasi. Avvalo **algoritm** tushunchasi 1X asrla yashab ijod etgan buyuk bobokalonimiz Muhammad al-Xorazmiy nomi bilan uzviy bog'liqligini tushuntirish lozim. **Algoritm** so'zi al-Xorazmiyning arifmetikaga bag'ishlangan asarining dastlabki betidagi **"Dixit Algoritmi"** (**"dediki al-Xorazmiy"** ning lotincha ifodasi) degan jumalardan kelib chiqqan. Shundan so'ng al-Xorazmiyning sanoq sistemasini takomillashtirishga qo'shgan hissasi, uning asarlari algoritm tushunchasining kiritilishiga sabab bo'lganligi ta'kidlab o'tiladi.

Algoritm nima degan savolga, u asosiy tushuncha sifatida qabul qilinganligidan, uning faqat tavsifi beriladi, ya'ni biror maqsadga erishishga yoki qandaydir masalani yechishga qaratilgan ko'rsatmalarning (buyruqlarning) aniq, tushunarli, chekli hamda to'liq tizimi tushuniladi.

O'quvchilarga turli hayotdan, fandan algoritmlarga misollar keltiriladi va bunga o'zlari tuzishga harakat qilishlari taklif etiladi. Masalan, atom tayyorlash, turli qurilmalarni ishlatish, sport musobaqasi yoki yo'l harakati qoidalari algoritmlarini keltirish mumkin. yoki matematik formula bo'yicha qiymat hisoblash algoritmi yoki kompyuterni ishlatish bo'yicha algoritm kabi misollar keltiriladi. Geometrik yasashga doir masala yechish algoritmi ham qarab chiqilishi mumkin. **Masalan**, kesma o'rtasiga perpendikulyar o'tkazish berilgan radiusga ko'ra aylana yasash va h.k. Bundan tashqari, birorta funksiya grafigini yasash algoritmi bunga misol bo'la oladi.

Masalan, chiziqli, kvadratik funksiya va h.k. funksiyalar grafiklarini yasash algoritmlari.

Bunda yana algoritmni bajarayotgan kishi – ijrochi, asosiy algoritmni aniqlashtiruvchi algoritm – **yordamchi algoritm** ekanligini ham ta’kidlab o’tish joiz. Umuman, algoritmning qanday maqsadga mo’ljallanganligidan qat’i nazar uni muvaffaqiyat bilan bajarish mumkinligini aytib o’tish lozimdir.

Algoritmning asosiy xossalari haqida quyidagilarni ta’kidlash mumkin:

1-xossa. Diskretlilik, ya’ni algoritmni chekli sondagi oddiy ko’rsatmalar ketma-ketligi shaklida ifodalash mumkin.

2-xossa. Tushunarlilik, ya’ni ijrochiga tavsiya etilayotgan ko’rsatmalar uning uchun tushunarli bo’lishi shart, aks holda ijrochi oddiy amalni ham bajara olmay qolishi mumkin. Har bir ijrochining bajara olishi mumkin bo’lgan ko’rsatmalar tizimi mavjud.

3-xossa. Aniqlik, ya’ni ijrochiga berilayotgan ko’rsatmalar aniq mazmunda bo’lishi lozim hamda faqat algoritmida ko’rsatilgan tartibda bajarilishi shart.

4-xossa. Ommaviylik, ya’ni har bir algoritm mazmuniga ko’ra bir turdagi masalalarning barchasi uchun yaroqli bo’lishi lozim. **Masalan**, ikki oddiy kasr umumiy maxrajini topish algoritmi har qanday kasrlar umumiy maxrajini topish uchun ishlatiladi.

5-xossa. Natijaviylik, ya’ni har bir algoritm chekli sondagi qadamlardan so’ng albatta natija berishi lozim.

Bu xossalarni mohiyatini o’rganish va konkret algoritmlar uchun qarab chiqish talabalarning xossalarni mazmunini bilib olishlariga yordam beradi.

Algoritmning tasvirlash usullari haqida gapirganda algoritmning berilish usullari xilma-xilligi va ular orasida eng ko’p uchraydiganlari quyidagilar ekanligini ko’rsatib o’tish joiz:

1. Algoritmning so’zlar orqali ifodalanishi.

2. Algoritmning formulalar yordamida berilishi.

3. Algoritmning jadval ko’rinishida berilishi, masalan, turli matematik jadvallar, loteriya yutuqlari jadvali, funksiyalar qiymatlari jadvallari bunga misol bo’ladi.

4. Algoritmning dastur shaklida ifodalanishi, ya’ni algoritm kompyuter ijrochisiga tushunarli bo’lgan dastur shaklida beriladi.

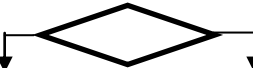
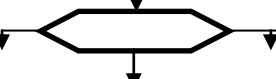
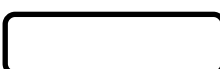
5. Algoritmning algoritmik tilda tasvirlanishi, ya’ni algoritm bir xil va aniq ifodalash, bajarish uchun qo’llanadigan belgilash va qoidalar majmui algoritmik til orqali ifodalashdir. Ulardan o’quv o’rganish tili sifatida foydalanilmoqda. Bulardan Ye-praktikum yoki Ye-tili algoritm ijrochisi algoritmik tili ham mavjud.

6. Algoritmning grafik shaklda tasvirlanishi. Masalan, grafiklar, sxemalar ya’ni blok - sxema bunga misol bo’la oladi. Blok sxemaning asosiy elementlari quyidagilar: **oval (ellips shakli)**-algoritm boshlanishi va tugallanishi, **to’g’ri burchakli to’rtburchak**-qiymat berish yoki tegishli ko’rsatmalarni bajarish. **Romb - shart** tekshirishni belgilaydi. Uning yo’naltiruvchilari tarmoqlar bo’yicha biri ha ikkinchisi yo’q yo’nalishlarni beradi, **parallelogramm**- ma’lumotlarni kiritish yoki chiqarish, **yordamchi algoritmga murojaat** - parallelogramm ikki tomoni

chiziq, *yo'naltiruvchi chiziq* - blok-sxemadagi harakat boshqaruvi, *nuqta-to'g'ri chiziq (ikkita parallel)* - qiymat berish.

Algoritmida bajarilishi tugallangan amallar ketma-ketligi **algoritm qadami** deb yuritiladi. Har bir alhoxida qadamni ijro etish uchun bajarilishi kerak bo'lgan amallar haqidagi ko'rsatma buyruq deb aytiladi.

Algoritmnlarni ko'rgazmaliroq qilib tasvirlash uchun blok-sxema, ya'ni geometrik usul ko'proq qo'llaniladi. Algoritmning blok-sxemasi algoritmning asosiy tuzilishining yaqqol geometrik tasviri: algoritm bloklari, ya'ni geometrik shakllar ko'rinishida, bloklar orasidagi aloqa esa yunaltirilgan chiziqlar bilan ko'rsatiladi. Chiziqlarning yunalishi bir blokdan so'ng qaysi blok bajarilishini bildiradi. Algoritmnlarni ushbu usulda ifodalashda vazifasi, tutgan o'rniga qarab quyidagi **geometrik shakl(blok)** lardan foydalaniladi.

Blokning atalishi	Belgilanishi	Tushunilishi
Hisoblashlar bloki (to'g'ri-to'rtburchak)		Hisoblash amali yoki hisoblash amallari ketma-ketligi
shartli blok (romb)		Shartlarni tekshirish
siklik jarayon (oltiburchak)		Siklning boshlanishi
qism dastur		qism dastur bo'yicha hisoblash, standart qism dasturi
birlashtirish (aylana)		Yo'nalish chizig'ini o'zgartirish
Ma'lumotlarni kiritish va chiqarish (parallelogramm)		Ma'lumotlarni kiritish va natijalarni chiqarish
Algoritmning boshi va oxiri (oval)		Boshlash, tamom, to'xtash
Chiqarish bloki		Ma'lumotlarni qog'ozga chiqarish

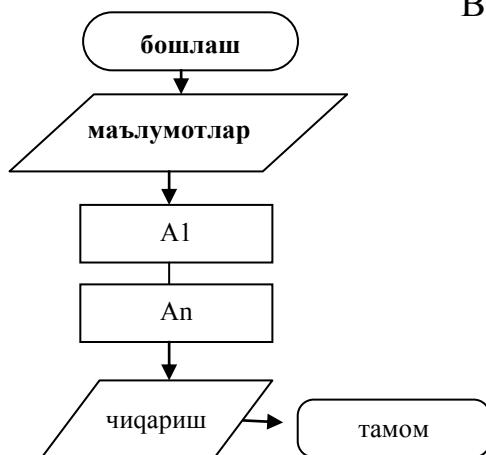
Algoritmilar berilishi va ifodalanishiga qarab: **chiziqli, tarmoqlanuvchi va takrorlanuvchi** turlarga bo'linadi.

Algoritmning turlari bilan tanishtirganda, avvalo hych qanday shart tekshirilmaydigan va tartib bilan faqat ketma-ket bajariladigan jarayonlarni ifodalaydigan **chiziqli algoritmilar** aytib o'tiladi.

Chiziqli algoritmilar. Chiziqli algoritmilar algoritmnlarning eng sodda va oddiy ko'rinishi hisoblanadi. Unida bajariladigan amallar ham buyruqlar ham buyruqlar ham qanday tartibda berilgan bo'lsa shunday tartibda ketma- ket bajariladi, ya'ni hych qanday shart tekshirilmasdan chiziqli algoritmnlarda buyruqlar ketma- ket tartib bilan bajariladi.

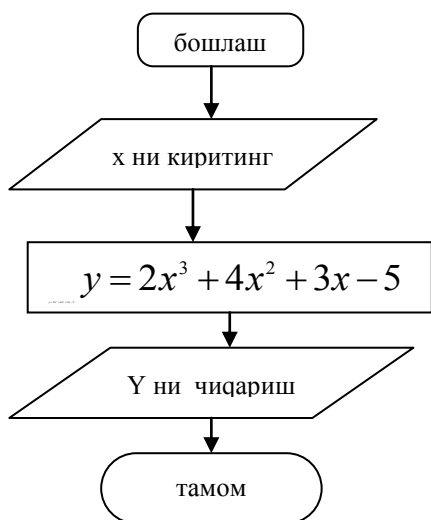
Chiziqli algoritmlarni quyidagi ko'rinishda ifodalash mumkin.

Bu yerda $A_1, \dots, A_n$ lar chiziqli algoritmlarda bajarilishi kerak bo'lgan buyruqlar ketma- ketligidir.



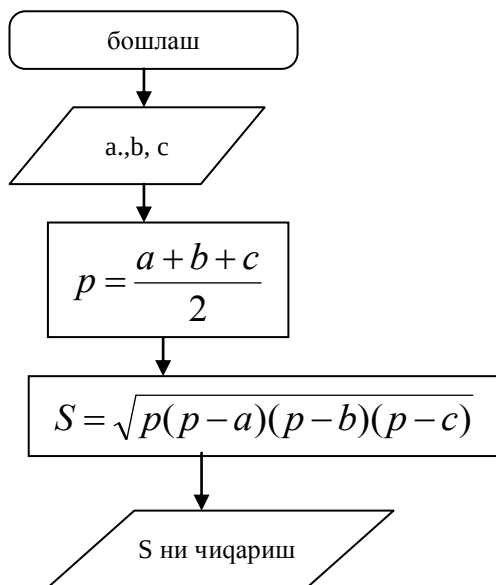
2- misol. $y = 2x^3 + 4x^2 + 3x - 5$ funksiyani x ning ixtiyoriy qiymatlarida hisoblash algoritmini tuzing.

Yechish. Algoritmning blok sxema ko'rinishda ifodalaymiz.



2- misol. $S = \sqrt{p(p-a)(p-b)(p-c)}$ Geron formulasidan foydalanib uchburchak yuzini hisoblash algoritmini tuzing.

Yechish. Algoritmni blok sxema ko'rinishda ifodalaymiz.



Sinov savollari:

1. Masalani kompyuterda yechishning asosiy bosqichlarini ayting?
2. Algoritm nima?
3. Algoritmga misollar keltiring.
4. Algoritmning asosiy xossalari.
5. Algoritmning tasvirlashning asosiy usullari.
6. Bu tasvirlash usullarining har biriga misollar keltiring.
7. Blok-sxema nima? Asosiy elementlarini ayting.
8. Chiziqli algoritmlar qanday ifodalanadi?
9. Algoritmning turlari.
10. Ijrochining ko'rsatmalar majmui nimalardan iborat?

Uyga vazifa :

1. *Oddiy kasrlarni ayirish algoritmini tuzing.*
2. *$u=5+(3-x)(2x+7)$ ning qiymatini hisoblash algoritmini tuzing.*
3. *Kesmani teng ikkiga bo'lish algoritmini tuzing.*
4. *Quyidagi algoritmda qanday natija hosil bo'ladi: $x:=2$; $x:=x+7$ $x=?$*

9-ma'ruza

Mavzu: Tarmoqlanuvchi va takrorlanuvchi jarayonlarga algoritmlar tuzish.

Reja:

1. *Tarmoqlanuvchi algoritmlar.*
2. *Takrorlanuvchi algoritmlar.*
3. *Murakkab algoritmlar bilan ishlash.*
4. *Mustaqil ishlash.*

Darsning maqsadi:

1. Talabalarda tarmoqlanuvchi va takrorlanuvchi jarayonlarni algoritmlash to'g'risida umumiy va asosiy ko'nikmalarni hosil qilish.
2. Tarmoqlanuvchi va takrorlanuvchi algoritmlar tuzish malakalarini shakllantirish.
3. Murakkab jarayonlarni algoritmlashni o'rganish.

Tayanch iboralar: algoritmning turlari, tarmoqlanuvchi va takrorlanuvchi algoritmlar.

Dars o'tish vositalari: *sinf doskasi, o'quv-uslubiy qo'llanmalar, ma'ruza matnlari, komputer, ma'ruza bo'yicha slaydlar.*

Dars o'tish usuli: *namoyish dasturiy vositalat ishtirokida, jonli muloqotli ma'ruza .*

Darsning xrono xaritasi – 80 minut.

Tashkiliy qism: xonaning tozaligi, jihozlanikishi, sanitariya holati. Talabalarining davomati – 2 minut.

Talabalar bilimini baholash: o‘tilgan mavzuni qisqacha takrorlash, talaba bilan savol-javob o‘tkazish – 10 minut.

Yangi mavzu bayoni – 55 minut.

Mavzuni o‘zlashtirish darajasini aniqlash va mustahkamlash – 10 minut.

Sinov savollari – 5 minut.

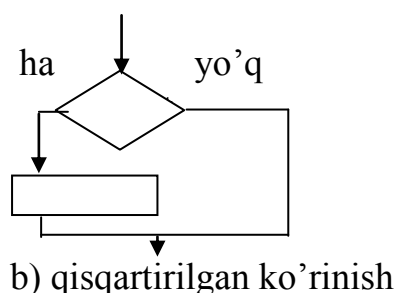
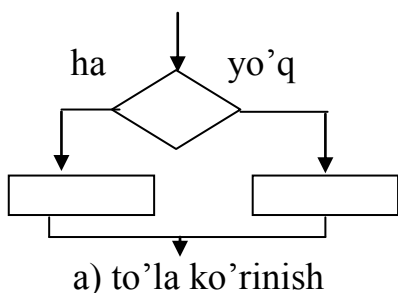
Uyga vazifa berish – 3 minut.

Ma‘ruza bayoni

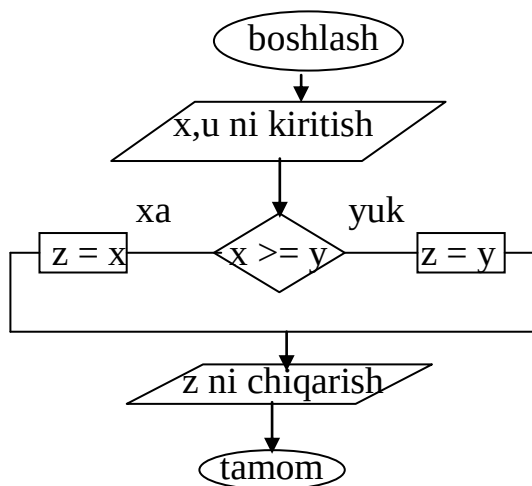
Tarmoqlanuvchi algoritmlar. Shunday hisoblash jarayonlari mavjud bo‘ladiki, bunda qo‘yilgan ayrim mantiqiy shartlarning bajarilishiga qarab, bu jarayonlar bir nechta tarmoqqa bo‘linadi va shu tarmoqlardan hеч bo‘lmaganda bittasi bajariladi. Ana shunday jarayonlar uchun algoritmlar tuzishda tarmoqlanuvchi algoritmlardan foydalaniladi.

Tarmoqlanuvchi struktura odatda qandaydir mantiqiy shartni tekshirish blokini o‘z ichiga oladi. Tekshirish natijasiga ko‘ra, tarmoq deb ataluvchi u yoki bu amallar ketma-ketligi bajariladi.

Tarmoqlanuvchi algoritm to‘la va qisqartirilgan ko‘rinishda berilishi mumkin. Ular quyidagicha sxema orqali ifodalanadi:

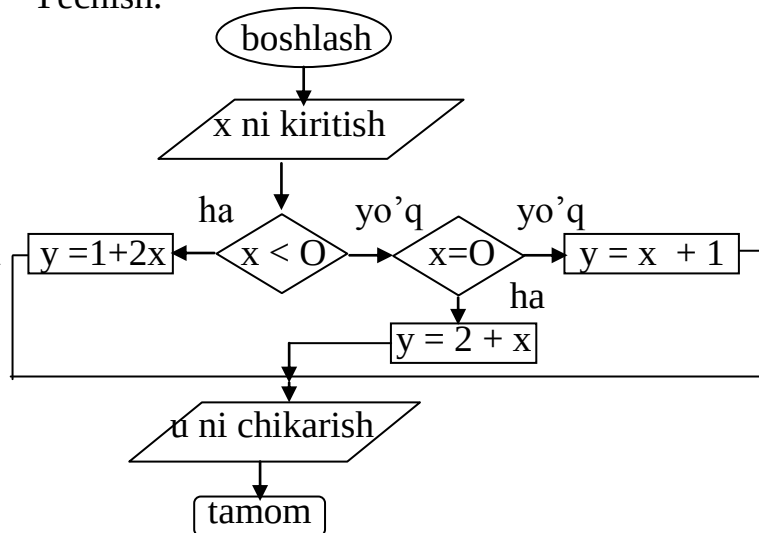


1-misol. Ixtiyoriy berilgan x va u sonlaridan eng kattasini topish algoritmini tuzing.



2- misol. $\begin{cases} 1+2x, & \text{agar } x < 0 \\ 2+x, & \text{agar } x = 0 \\ x+1, & \text{agar } x > 0 \end{cases}$ ifodaning qiymatini hisoblash algoritmini tuzing.

Yechish.

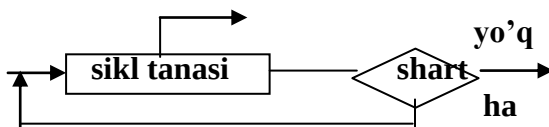


Takrorlanuvchi algoritmlar. Shunday hisoblash jarayonlari mavjud bo'ladiki, bunda uning ayrim bo'laklarini bir necha marta takroran hisoblashga to'g'ri keladi. Bunday jarayonlar uchun algoritmlar tuzishda takrorlanuvchi algoritmlardan foydalaniladi.

Hisoblash jarayonining ko'p marta takrorlanadigan qismi ichki **sikl tanasi (jismi)** deb yuritiladi.

Takrorlanuvchi algoritm 2 xil ko'rinishga ega bo'ladi:

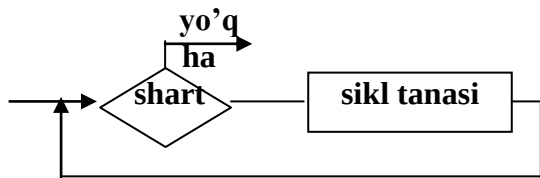
Sikl - gacha takrorlanuvchi algoritm quyidagi ko'rinishga ega:



a) sikl - gacha ;

Bu ko'rinishdagi algoritmda avval sikl tanasi bajarilib, so'ngra sikldan chiqish sharti tekshiriladi, ya'ni sikl tanasi qo'yilgan shart bajarilib turguncha takrorlanaveradi.

Sikl - hozircha takrorlanuvchi algoritm quyidagi ko'rinishga ega:



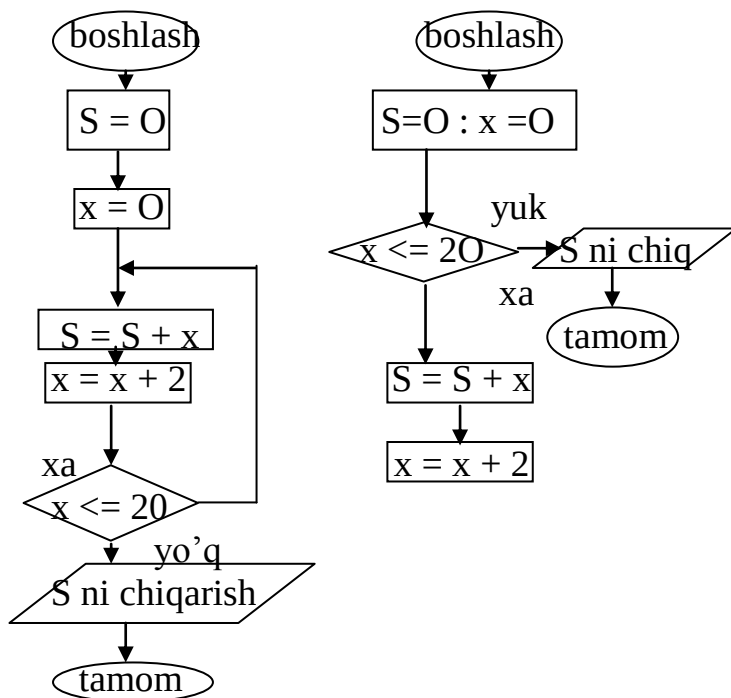
b) sikl - hozircha.

Bu ko'rinishdagi algoritmlarda avval shart tekshiriladi, so'ngra agar shart qanoatlantirsa, sikl tanasi bajariladi, aks holda hisoblash to'xtatiladi.

1 - misol. 0 dan 20 gacha juft sonlar yig'indisini hisoblash algoritmini tuzing.

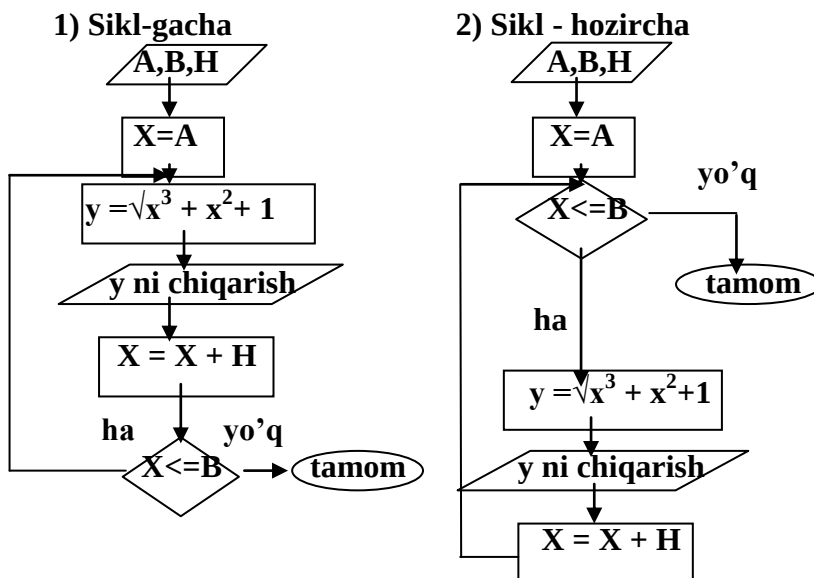
Yechish. **a) sikl - gacha**

b) sikl - hozircha



2-misol. $y = \sqrt{x^3 + x^2 + x}$ funksiyani x ning $[a, b]$ oraliqda $h=0.1$ qadam bilan o'zgarish qiymatini hisoblash algoritmini tuzing.

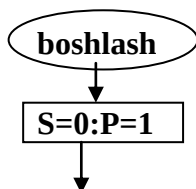
Yechish.

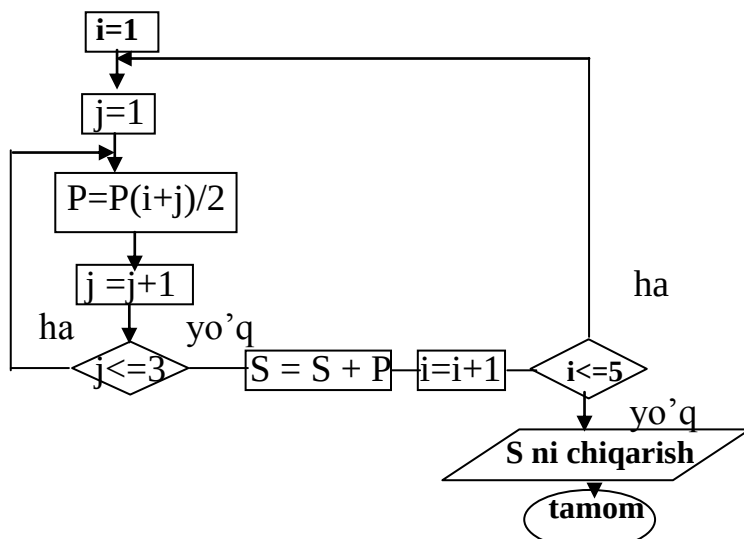


Bundan tashqari murakkab ko'rinishdagi takrorlanuvchi jarayonlar uchun algoritmlar tuzishda ichma-ich joylashgan takrorlanuvchi algoritmlardan foydalaniladi.

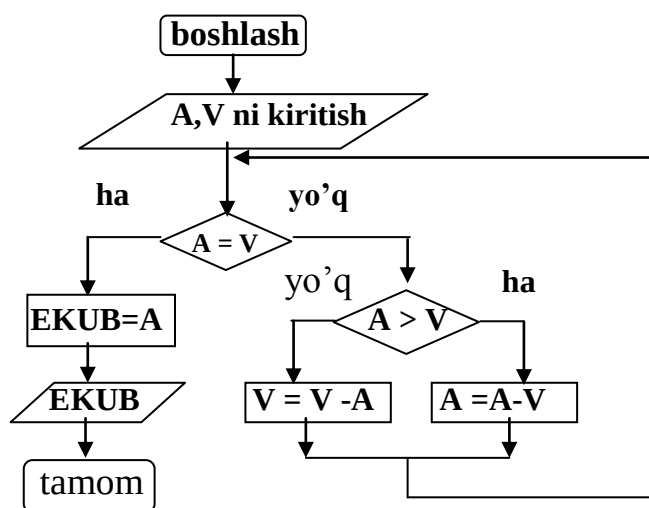
Misol. $S = \sum_{i=1}^5 \prod_{j=1}^3 i + j$ ifodaning qiymatini hisoblash algoritmini tuzing.

Yechish. Bu yerda i bo'yicha takrorlanuvchi algoritm tashqi, j bo'yicha takrorlanuvchi algoritm ichki sikllar deb yuritiladi.





Misol. Ikki a va v natural sonlari uchun eng katta umumiy bo'luvchi topilsin(blok-sxema ko'rinishi):



Sinov savollari:

1. Tarmoqlanuvchi algoritmlarning xususiyatlari va bajarish tartibi qanday?
2. Takrorlanuvchi algoritmlar deganda nimani tushunasiz?
3. Murakkab hisoblash jarayonlari uchun qanday algoritmlardan foydalani-ladi?
4. Ichma- ich joylashgan algoritmlarning bajarilish tartibini tushuntirib bering?

Uyga vazifa :

1. Berilgan sonning absolyut qiymatini aniqlab beruvchi algoritm tuzing.
2. Berilgan k butun sonning toq yoki juftligini aniqlab beruvchi algoritm tuzing.

10-ma'ruza

Mavzu: Dasturlash asoslari to'g'risida umumiy ma'lumotlar. Turbo-Paskal dasturlash tizimi, uning umumiy tuzilishi, asosiy "menyu" lari va ishlash qoidalari.

Reja:

1. Dasturlash tillari va ular to'g'risida umumiy ma'lumotlar.
2. Yuqori darajali dasturlash tillari.

3. Turbo-Paskal tizimi va uning umumiy tuzilishi.

4. Turbo-Paskal tizimida mustaqil ishlash.

Darsning maqsadi:

1. Talabalarda dasturlash, uning turlari to'g'risida umumiy va asosiy bilimlarni hosil qilish.

2. Turbo-Paskal tizimi va uning umumiy tuzilishi to'g'risida mukammal tasavvurga ega bo'lish.

3. Turbo-Paskal tizimida mustaqil ishlash malakalarini shakllantirish.

Tayanch iboralar: *dasturlash tili, dastur, translyator, son, miqdor, o'zgaruvchi, o'zgarmas, amallar, nom, identifikator, tur, oddiy va murakkab turlar, standart turlar.*

Dars o'tish vositalari: *sinf doskasi, o'quv-uslubiy qo'llanmalar, ma'ruza matnlari, kompyuter, ma'ruza bo'yicha slaydlar, Turbo-Paskal tizimi.*

Dars o'tish usuli: *namoyish dasturiy vositalat ishtirokida, jonli muloqotli ma'ruza.*

Darsning xrono xaritasi – 80 minut.

Tashkiliy qism: *xonaning tozaligi, jihozlanikishi, sanitariya holati. Talabalarning davomati*– 2 minut.

Talabalar bilimni baholash: *o'tilgan mavzuni qisqacha takrorlash, talabala bilan savol javob o'tkazish* - 10 minut.

Yangi mavzu bayoni - 55 minut.

Mavzuni o'zlashtirish darajasini aniqlash va mustahkamlash – 10 minut.

Sinov savollari – 5 minut.

Uyga vazifa berish – 3 minut.

Ma'ruza bayoni

Biz axborot, kod, kompyuter, blok- sxema, algoritm, ijrochi va boshqa tushunchalarni o'rgandik. Bular masalani kompyuterda hal qilishda kerak bo'ladigan tushunchalarni o'rgandik. Bular masalani kompyuterda hal qilishda kerak bo'ladigan tushunchalardir. Ammo kompyuter bilan muloqot qilish uchun, bulardan tashqari, kompyuter “tilini” ham bilishimiz talab qilinadi. Biror masalani kompyuterda hal qilish uchun, avval, uning algoritmini tuzishimiz, so'ng bu algoritmn kompyuter tushunadigan ko'rsatmalar va qonun qoidalar asosida yozishimiz lozim bo'ladi. Bu yozuv dastur deb ataladi.

Shunday qilib, dastur biror masalani yechishda kompyuter bajarishi mumkin bo'lgan ko'rsatmalarning izchil tartibidan iborat ekan. Kompyuter uchun dastur tuzish jarayoni dasturlash va dastur tuzadigan kishi dasturchi deb ataladi.

Kompyuter tushunadigan “til” dasturlash tili deb ataladi. (Eslang: algoritm ijrochisining ko'rsatmalar tizimi. Bu yerda kompyuter ijrochi sifatida qatnashmoqda.)

Hozircha yaratilgan dasturlash tillarini, ulardagi ko'rsatmalar, qonun qoidalarga ko'ra uch guruhga ajratish mumkin:

1. **Quyi darajadagi.**

2. **O'rta darajadagi.**

3. **Yuqori darajadagi.**

Quyi darajadagi dasturlash tillari da ko'rsatmalar raqamlar yordamida beriladi. Barcha amallar ma'lum raqamlar bilan ifodalangan bo'lib, ular mashina qurilmalari adreslar bilan bevosita ishlashga mo'ljallangan. Bu tilda dastur tuzuvchi amallarning sonli ifodasi va adreslarni puxta bilishi kerak.

O'rta darajadagi dasturlash tillarida ko'rsatmalar sifatida inson tiliga yaqin bo'lgan, ammo qisqartirilgan holda ishlatiladigan so'zlar tanlanadi. Bunda ham dasturchi adreslar bilan ishlashi lozim. Qisqartirilgan ko'rsatmalar mnemokodlar, bu turdagi dasturlash tillari esa assemblerlar deb ataladi.

Yuqori darajadagi dasturlash tillaridagi ko'rsatmalar inson tiliga yaqin bo'lgan so'zlardan iborat. Ularda amallarni bajarish avvalgilardan oson bo'lib, dasturchi, biror maxsus amal bo'lmasa, adreslar, qurilmalar bilan bevosita bog'liq ko'rsatmalarni bilmasligi ham mumkin.

Kompyuter amallarni bajarish uchun ularni raqamli ko'rinishga keltirish kerak bo'ladi. Bu vazifani translyatorlar deb nomlanadigan maxsus dasturlar bajaradi. Translyatorlar dasturlashtilida yozilgan dasturlarni mashina tiliga tarjima qiladi. Ular 2 xil bo'ladi: interpretator va kompilyator.

Keyingi yillarda juda ko'p yuqori darajadagi dasturlash tillari yaratilganligi sizlarga ma'lum. Ular qatoriga Paskal, Simula, dBase, KARAT, Si va boshqa tillarni kiritish mumkin. Dasturlash tillarining ko'payib borayotganligiga qaramay, har bir dasturlash tilining o'z maqsadi, vazifasi va qo'llanish sohalari bor. Masalan, hisoblashga bog'liq masalalarni yechish uchun Paskal, Fortran, belgili ma'lumot va ro'yxatlarni qayta ishlash uchun LISP, matnli ma'lumotlar bilan ishlash uchun SI, muloqot uchun Beysik Paskal, iqtisodiy masalalarni yechish uchun dBase, Kabol va ro'yxatlarni qayta ishlash uchun dasturlash tillaridan foydalaniladi.

Hisoblash texnikasining tez sur'atlar bilan rivojlanib borishi natijasida, ular barcha sohalarda, shu jumladan o'quv jarayonida ham keng qo'llanilmoqda. O'quv jarayonlarida kompyuterning barcha imkoniyatlaridan samarali foydalanish uchun dasturlash asoslari, ya'ni dasturlash tillarini mukammal o'rganish kerak bo'ladi. Ana shunday dasturlash tillaridan biri Paskal dasturlash tili bo'lib hisoblanadi.

Paskal dasturlash tili mini va mikro-EHM da qo'llanuvchi eng keng tarqalgan tillardan biridir. U 1971 yilda Shveysariya Oliy maktabi professori Niklaus Virt tomonidan yaratilgan. Bu til birinchi jamlovchi qurilmalar ixtirochisi Blez Paskal nomi bilan atalgan. Hozirgi kunda juda ko'plab hisoblash mashinalarida Paskal tili kompilyatori mavjud bo'lib, u tilni keng qo'llashga va amaliy dasturlash uchun imkoniyat yaratadi. Paskal tili dasturlarni tarkibiy(strukturali) ko'rinishda yozishga imkon beradi.

Dasturlarni Paskal tilida yozishda quyida qabul kilingan umumiy qoidalarga rioya qilinadi:

1. *Paskal tilida ham, boshqa dasturlash tillari kabi xizmatchi so'zlar mavjud bo'lib, ulardan o'zgaruvchilarni belgilash yoki o'zlari uchun mo'ljallangan maqsadlardan boshqa maqsadlarda foydalanish mumkin emas.*

2. *Paskal tilining barcha naqllarida katta va kichik harflar ajratilmaydi, chunki chop etish qurilmasida kichik harfli dasturlar matni katta harflarga almashtiriladi.*

3. Dasturlash jarayonida dasturning, o'zgaruvchilarning, o'zgarmaslar-ning va procedura- funksiyalarning nomlari boshlanishidagi 6-8 ta belgisi bilan farqlanadi. Shuning uchun uzundan uzun nomlardan foydalanish shart emas.

Turbo – paskal' tizimining umumiy tuzilishi, asosiy “menyu”si va ishlash qoidalari.

Paskal' dasturlash tilining birinchi versiyasi **1968 yilda** Stenford universitetida shved olimi Niklaus Virt tomonidan yaratilgan. 1982 yilda Amerikaning Borland firmasi tomonidan Turbo Paskal' tizimining yaratilishi, Paskal' tilining keng doirada qo'llanila boshlanishiga asos bo'ldi. Turbo Paskal' tili birinchi versiyaga qaraganda ancha kengaytirilgan bo'lib, unda tez ishlaydigan kompilyator matn muxariri bilan birlashtirilgan. Keyinchalik Turbo Paskal' tilining 3.0, 5.0 va 6.0 – versiyalari paydo bo'ldi.

1992 yilda Borland firmasi Turbo Pascal 7.0 – versiyani taqdim etdi. Bu versiyada kompilyatorning ishi ancha yaxshilangan, obyektga mo'ljallangan dasturlash tamoyili to'liq ishga tushirilgan va uning imkoniyatlari kengaytirilgan.

Dasturni ishga tushirish.

IDE (Integrated Development Environment) – integrallashgan vositalar qobig'i (**IVQ**) – Turbo Paskal' muhiti deb yuritiladi. Turbo Paskal' muhiti **Turbo.exe** fayli orqali **MS DOS** operasion tizimining buyruqlar satrida yoki **Windows** muhitining ishchi stolida ishga tushiriladi.

Turbo.EXE fayli ishlash uchun tayyor. Turbo Paskal' tilida dasturlashning muloqot (dialog) tilidan iborat. Bu tizim – matn muxarriri kompilyator, jamlovchi va yuklovchi qismlardan iborat. Muloqot tizimida normal ishlash uchun asosiy dasturlar kutubxonasi (**Turbo.TPL**) va ma'lumotlar majmuasi (**Turbo.ALP**) mavjud bo'lishi kerak.

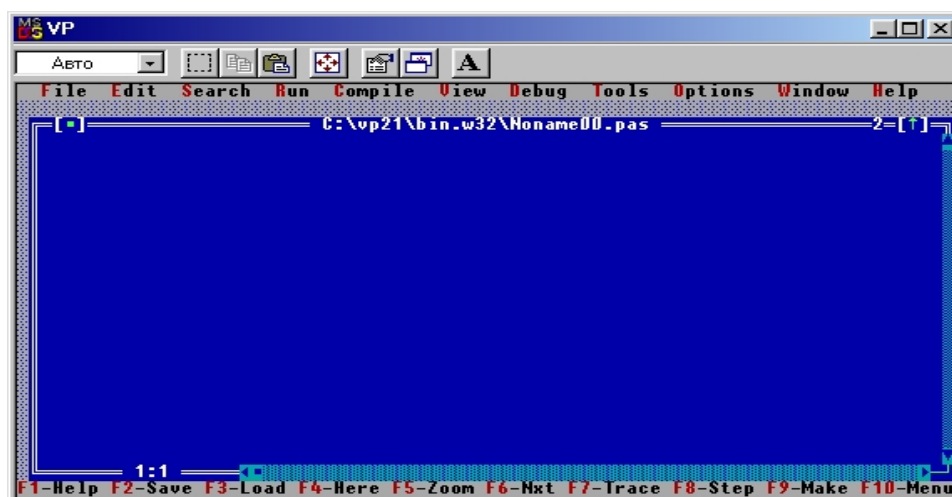
Turbo Paskal' tizimidan chiqish [**ALT+X**] tugmachalarini birgalikda bosish orqali amalga oshiriladi.

Turbo paskal' tizimining bosh “menyu” si.

Turbo Paskal muhiti ishga tushirilgandan keyin kompyuter ekranida uning oynasi paydo bo'ladi. Oyna sarlavxalar satri, bosh menyu va ishchi soxa (matn muxarriri) va buyruqlarsatridan iborat.

Sarlavhalar satri matn muharirida yozilgan dastur nomidan iborat bo'ladi.

Paskal' tilida tuzilgan dastur bilan ishlash uchun asosan bosh menyudan foydalaniladi. Bosh menyuga chiqish uchun **F10** tugmacha bosiladi.



File Edit Seatch Rum Compile Debid Tools Options Winlow Help

Bosh “menyu” quyidagi asosiy bo’limlardan iborat:

1. File – fayli amallar ma’nosi bo’lib, fayllar bilan ishlash amallarini bajarishga mo’ljallangan. **File** bilan ishlash amallari:

Open (F3) – mavjud faylni muharrirlash uchun yuklash. Bu bo’lim tanlangandan keyin ekranda muloqat oynasi hosil bo’ladi. Oynada yuklash lozim bo’lgan fayl nomi ko’rsatilib, **Enter** tugmachasi bosiladi. Fayl yuklangandan keyin u muharrirlash va uni bajarish uchun tayyor bo’ladi.

Save(F2) – faylni saqlash. Bu bo’lim ishga tushirilganda muharrir xotirasidagi fayl diskga yoziladi. Agar muharrirlanayotgan fayl standart **Noname.pas** nomiga ega bo’lsa, u holda saqlashda fayl nomini o’zgartirish imkoniyati mavjud bo’ladi.

Save as – muharrirlanayotgan faylni boshqa nom bilan saqlash. Ushbu buyruq amalga oshirilgandan keyin, muharrirlanayotgan faylga boshqa nom beriladi va u kompyuter xotirasiga yangi nom bilan saqlanadi.

New – yangi faylni yaratish. Bu buyruq berilgan integrallashgan muhit muharriri yangi fayl yaratish tartibiga o’tadi. Yaratilyotgan faylga **Noname. pas** nom beriladi. Faylni saqlayotgan unga ixtiyori nom berish mumkin bo’ladi.

Exit (Alt +x) – dasturlash muxitida ishni tugallash. Bu buyruq berilgandan keyin integrallashgan muhitda ish tugallaniladi va undan chiqiladi.

2. Edit – fayllarni muharrirlashni yu qo’shimcha amallari “**menyu**”si bo’lib, matnni muharrirlashning asosiy funksiyalarini bajaradi. Bu amallar quyidagilardan iborat:

- *matn bo’laklarini nusxalash;*
- *matn bo’laklarini o’rnatish;*
- *matn bo’laklarini o’chirish;*
- *muxarrirlanayotgan faylning boshlang’ich variantini tiklash ;*

3. Search – matn bo’laklarini izlash va almashtirish “**menyu**”si bo’lib, matn qismlarini izlashni, lozim bo’lganda ularni yangisi bilan almash-tirishni amalga oshiradi

4. Run – Paskal’ dasturini bajarish “**menyu**”si bo’lib, ishchi sohada joy-lashgan dasturni ishga tushiradi. Bundan tashqari ushbu dasturni qadamma- qadam bo’yicha, hamda lozim bo’lganda esa ayrim qismini bajarishni amalga oshiradi.

Bu buyruq berilganda dastur ishga tushadi, agar dasturda xatoliklar mavjud bo’lmasa, o’zgaruvchilarga sonli qiymatni berishni talab etuvchi muloqat oyna hosil bo’ladi. Qiymatlar tartib bilan berilgandan keyin **Yenter** tugmachasi bosilib, dastur ishga tushiriladi. Dastur natijasini ko’rish **Alt +F5** tugmachalarini bosish orqali amalga oshiriladi.

5. Compili – kompilyasiya “**menyu**”si bo’lib, ishchi sohada joylashgan dastur-ni kompilyasiya qilish uchun ishlatiladi. Bu buyruq berilganda dastur kompilyasiya qilinadi. Agar kompilyasiya vaqtida dasturda xatolik mavjud bo’lsa, u holda bu haqda kerakli xabar kompyuter ekraniga chiqariladi. Xatoliklar tuzatilgandan keyin dastur yana kompilyasiya qilinadi. Shuning uchun dasturda mavjud xatoliklarni

aniqlash maqsadida, uni vaqti- vaqti bilan kompilyasiya qilib turish kerak bo'ladi. Kompilyasiya qilingan dastur bajariladi.

6.Debug – dasturni otladka qilish amallari **“menyu”**si bo'lib, dasturda mavjud xatoliklarni topishni yengillashtiruvchi buyruqlardan iborat.

7. Tools – instrumental vositalar **“menyu”**si bo'lib, Turbo Paskal' muhitidan chiqmasdan turib, ayrim dasturlarni bajarishni amalga oshiradi.

8. Options- muhit parametrlarini o'zgartirish **“menyu”**si bo'lib, Turbo Paskal' muhitida ishlash jarayonida lozim bo'lgan kompilyator parametr-larini o'rnatish uchun ishlatiladi.

9. Window – oynalarni boshqarish **“menyu”**si bo'lib, oynalar bilan ishlashning barcha asosiy amallarini bajaradi. Bu amallar: ***oynalarni ochish, oynani yopish, oynani kerakli joyga joylashtirish va oyna o'lchovini o'zgartirishdan iborat.***

10. Help – yordam olish **“menyu”**si bo'lib, muhitda ishlash bo'yicha kerakli ma'lumotlarni olish uchun ishlatiladi.

Yuqorida ta'kidlaganimizdek, bosh **“menyu”**ga qarashli bo'lim – **ostki “menyu”** deb yuritiladi. Ostki **“menyu”** buyrug'ini tanlash uchun, uning nomi ustiga kursor ko'rsatkichi joylashtirilib, **Enter** tugmachasi bosiladi. Ostki **“menyu”** dan chiqish **Esc** tugmachasini bosish orqali amalga oshiriladi.

Matn muhariri buyruqlari

Turbo Paskal' muhitida ishlash uchun matn muharriri mavjud bo'lib, unda ishlash eng quyi darajadagi matn muharririga o'xshab ketadi. Turbo Paskal' muhiti matn muharriri Paskal' tilidagi dasturlarni kiritish va ularni muharrirlash uchun ishlatiladi.

Turbo Paskal' muhitida ishlashda ma'lum bir cheklashlar mavjud:

- 1. Faylda umumiy belgilar soni 64535 tadan oshib ketmasligi kerak.**
- 2. Dasturda bitta satrning uzunligi 126 ta belgidan ortiq bo'lmasligi kerak.**
- 3. Satrni 127- chidan 249- chi o'rniga bo'lgan joy izoxlarni joylash uchun ishlatiladi.**

Matn muharririda ishlashda har bir satrning oxirida **Enter** tugmachasi bosilib, keyingi yangi satrga o'tiladi. **Enter** tugmachasini bosish orqali bitta satrni bir nechta satrga ajratish mumkin.

Agar kursor satrning oxirida turgan bo'lsa, **Del** – tugmachasini yoki agar kursor ikkinchi satrning boshida turgan bo'lsa, **BkSp** tugmasini bosish orqali ikkita satr bitta satrga birlashtiriladi.

Turbo Paskal' muhitida matnlar bilan ishlashda bajarilgan amallarni bekor qilish, ya'ni matnni boshlang'ich variantiga qaytish uchun **Edit** bo'limidan **Undo ([alt+ BkSp])** tiklash uchun **Redo** buyrug'i bajariladi.

Matn bo'laklarini ajratib olish uchun quyidagi amallardan birortasi- ni bajarish kerak:

- 1) sichqoncha ko'rsatgichini birinchi belgiga o'rnatib, uning chap tugmachasini bosib turgan holda oxirgi belgiga harakatlantirish;**
- 2) Shift tugmachasini bosib turgan holda matn kursorini siljitish.**

Ajratib olingan matn bo'laklarini ko'chirish, nusxalash va o'chirish uchun **Edit** menyusini buyruqlaridan foydalaniladi.

Turbo Paskal' muhiti matn muharririda ishlashda asosan quyidagi buyruqlardan foydalaniladi:

1) Kursorni joylashtirish:

PgUp- kursorni bir varaq yuqoriga;

PgUp- kursorni bir varaq pastga;

Home- satr boshiga;

End- satr oxiriga;

Ctrl+ PgUp- matn boshiga;

Ctrl+ PgUp-matn oxiriga;

2) Muharrirlash buyruqlari:

Baskpace- kursordan chap tomonda turgan belgini o'chirish.

Del- kursordan o'ng tomonda turgan belgini o'chirish;

Ctrl+Y- kursor joylashgan satrni o'chirish;

Enter- mavjud satrni bo'lish, ya'ni satr o'rnatish;

Ctrl+ QL- joriy satrni tiklash;

3) Bloklar bilan ishlash:

Ctrl+KB- sichqoncha yotdamida bo'lakni ajratish blokini boshiga o'rnatish;

Ctrl+ KK- blok oxiriga o'rnatish.

Ctrl+ KH- ajratish blokini olib tashlash va tiklash.

Ctrl+KY- blokni o'chirish;

Ctrl+del- blokni nusxalash;

Ctrl+Kc

Ctrl+Ins- blokni o'rnatish;

Ctrl+ KV

Ctrl+ KR- diskli fayldan blokni o'qish;

Ctrl+ KP- blokni chop etish;

Sinov savollari:

1. Turbo-Paskal tizimining umumiy tuzilishi qanday?
2. Menuning tarkibiy qismlari qaysi?
3. Bosh menu qanday buyruqlardan iborat?
4. Matn muharriri bilan ishlash tartibini tushuntiring?

Uyga vazifa : Turbo-Paskal tizimida ishlash qoidalarini mukammal o'rganish va ularni amaloyotga tatbiq etish.

11-ma'ruza

Mavzu: Paskal tilining umumiy xarakteristikalar, asosiy tushunchalari va elementlari.

Reja:

1. Paskal tilining alfaviti..
2. Paskal tilida miqdorlar va amallar.
3. Standart funksiyalar. Ifoda.
4. Ma'lumotlarning turlari.

Darsning maqsadi:

1. Talabalarda miqdorlar va amallar to'g'risida umumiy va asosiy bilimlarni hosil qilish.

2. Turbo-Paskal tizimida standart funksiyalar va ifodalar to'g'risida mukammal tasavvurga ega bo'lish.

4. Turbo-Paskal tizimida mustaqil ishlash malakalarini shakllantirish.

Tayanch iboralar: *dasturlash tili, dastur, miqdor, o'zgaruvchi, o'zgarmas, amallar, nom, identifikator, tur, oddiy va murakkab turlar, standart turlar.*

Dars o'tish vositalari: *sinf doskasi, o'quv-uslubiy qo'llanmalar, ma'ruza matnlari, kompyuter, ma'ruza bo'yicha slaydlar, Turbo-Paskal tizimi.*

Dars o'tish usuli: *namoyish dasturiy vositalat ishtirokida, jonli muloqotli ma'ruza.*

Darsning xrono xaritasi – 80 minut.

Tashkiliy qism: *xonaning tozaligi, jihozlanikishi, sanitariya holati. Talabalarining davomati* – 2 minut.

Talabalar bilimini baholash: *o'tilgan mavzuni qisqacha takrorlash, talaba bilan savol-javob o'tkazish* - 10 minut.

Yangi mavzu bayoni - 55 minut.

Mavzuni o'zlashtirish darajasini aniqlash va mustahkamlash – 10 minut.

Sinov savollari – 5 minut.

Uyga vazifa berish – 3 minut.

Ma'ruza bayoni

Paskal tilida dasturlarni yozish uchun ishlatiladigan barcha belgilar shaxsiy kompyuterlarning klaviaturasida mavjud. Mavjud bo'lmagan belgilar esa, mavjud belgilar majmuasi bilan almashtiriladi. **Masalan**, **> belgi** **>=** belgilar majmuasi bilan almashtiriladi.

Paskal tilining alfaviti. Paskal tilining alfavitini qo'llanish maqsadlariga qarab bir necha guruhga bo'lish mumkin. Asosiy belgilar:

1) **lotin va rus alfaviti harflari;**

2) **raqamlar:** 0,1,2,3,4,5,6,7,8,9 ;

3) **arifmetik amal belgilari:** +, -, *, /, div, mod;

4) **mantiqiy amal belgilari :** and, or, not;

5) **munosabat belgilari:** < , > , <= , >= , = , <>;

6) **ta'minlash belgisi :** := ; 7) **ajratuvchilar:** ., ; , , : , ' ;

8) **qavslar :** (,), {, }, [,]; 9) **boshqa belgilar:** !, @, #, %, ?, & ;

Xizmatchi so'zlar: and, array, begin, case, const, div, do, downto, else, end, file, for, function, goto, if, in, label, mod, nil, not, of, or, packed, procedure, program, record, repeat, set, string, then, to, type, until, var, while, with.

Standart identifikatorlar: 1) **o'zgarmaslar:** **false, true, maxint;**

2) **turlar:** **Boolean, integer, char, real, string, text;** 3) **fayllar:** **output, input;** 4) **funksiyalar va proceduralar.**

So'z. So'z dasturlash tilining alohida belgilaridan tashkil topgan bo'lib, ma'lum ma'noni anglatadi. Dastur matnida so'zlar bir-biridan bo'shliq yoki maxsus belgilar bilan ajratiladi. Foydalanilishiga qarab so'zlar xizmatchi so'zlar va nomga ajratiladi.

Nom yoki identifikatorlar o'zgaruvchilarni, o'zgarmaslarni, turlarni, proceduralarni va funksiyalarni nomlarini belgilash uchun ishlatiladi. Paskal tilida standart va ixtiyoriy beriladigan nomlar mavjud. Nom harf va raqamdan iborat bo'lib, doimo harfdan boshlanishi kerak.

Masalan: YY, GAMMA, Hoch, XX22.

Paskal tilida nom sifatida xizmatchi so'zlar va standart nomlardan, ya'ni standart o'zgarmaslar, funksiyalar, fayllar, proceduralar va turlarning nomlarini qo'llash mumkin emas. **Standart nomlar quyidagilardan iborat:** *abs, arctan, boolean, char, chr, cos, dispose, eos, eof, eoln, exp, false, forwar, get, input, integer, ln, maxint, new, odd, ord, output, pack,pagen, pred, put, read, readln, real, reset, rewrite, rounds, sin, sqr, sqrt, succ, text, true, string, trunc, unpack, write, writeln.*

Son. Paskal tilida sonlar o'nlikda tasvirlanadi. Sonlar butun va haqiqiy bo'lishi mumkin. Bunda (+) musbat belgisi tashlab yozilishi xam mumkin.

Haqiqiy son qo'zg'aluvchan vergulli va qo'zg'almas vergulli formalarda tasvirlanishi mumkin. **Misol: butun sonlar: 5, 05, +73,-72678; haqiqiy sonlar: 0.65,**

-5Ye-02, 150.0, 1.7Ye2, -5.0Ye2

Bundan tashqari Paskal tilida belgi va satrlardan ham foydalaniladi. Satrlar apostrof ichiga olingan belgilar ketma-ketligidan yoki bitta belgidan iborat bo'ladi: **'Hikmat' 'fakultet', 'summa ', 'q', '>'.**

Arifmetik amallar. Paskal tilida ham sonli ma'lumotlar ustida bir nechta amallarni bajarish mumkin. Quyida Paskal tilida mavjud amallarni keltiramiz.

Paskal tilida arifmetik amallar			
Amallar	Belgilash	Operandlar turi	Natija turi
Ko'paytirish	$A * B$	<i>REAL yoki INTEGER</i>	<i>REAL yoki INTEGER</i>
Bo'lish	A / B	---- " " ----	<i>REAL</i>
Qo'shish	$A + B$	---- " " ----	<i>REAL yoki INTEGER</i>
Ayirish	$A - B$	---- " " ----	---- " " ----
Butunga bo'lish	$A \text{ DIV } B$	<i>INTEGER</i>	<i>INTEGER</i>
Bo'linma qoldig'i	$A \text{ MOD } B$	<i>INTEGER</i>	<i>INTEGER</i>

Bajarilayotgan amallarda operandlardan hyech bo'lmaganda bittasi xaqiqiy turga mansub bo'lsa, u holda hosil bo'ladigan natija ham haqiqiy turga tegishli bo'ladi. Bo'linma natijasi har doim haqiqiy son bo'ladi.

Paskal tilida darajaga ko'tarish amali yo'q. Shuning uchun sonlarni butun darajaga ko'tarish ularni ko'p marta bir biriga ko'paytirish yo'li bilan amalga oshiriladi. Haqiqiy darajaga ko'tarish logarifmlash orqali bajariladi: $x = e$ **yoki** $x = 10$.

Paskal tilida butun sonli berilmalar uchun **DIV va MOD** amallari mavjud bo'lib, butunga bo'lish va bo'linma qoldig'i amallarini anglatadi. **Masalan: 11 DIV 3 = 3 ; 11 MOD 3 = 2.**

Munosabat va mantiqiy amallar. Paskal tilida ikkita mantiqiy konstanta true(chin) va false(yolg'on) ishlatiladi. Bular miqdorlarni taqqos-lashda va ba'zi

shartlarni tekshirishda qo'llaniladi. Ma'lumotlarni taq-qoslash uchun quyidagi munosabat amallaridan foydalaniladi: $<$ (**kichik**), $\leq$ (**kichik yoki teng**), $=$ (**teng**), $>$ (**katta**), $\geq$ (**katta yoki teng**).

Munosabat amallari sonli ma'lumotlar uchun qo'llanilganida mantiqiy qiymatlar hosil bo'ladi. **Masalan:** $96 > 73$ (*chin*), $-8 > -3$ (*yolg'on*), $4 = 7$ (*yolg'on*), $16 < 22$ (*chin*).

Mantiqiy ma'lumotlar uchun quyidagi mantiqiy amallardan foydalanish mumkin: **OR**- qo'shish (**VA**); **AND**- ko'paytirish (**YoKI**); **NOT** - inkor (**YO'Q**). **OR** va **AND** mantiqiy amallar ikkita kattalik bilan, **NOT** amali esa bitta kattalik bilan bajariladi.

Mantiqiy qiymatlar ustida amallar natijasi quyidagicha:

A	V	not A	A OR B	A AND B
<i>true</i>	<i>true</i>	<i>False</i>	<i>true</i>	<i>true</i>
<i>true</i>	<i>false</i>	<i>False</i>	<i>true</i>	<i>false</i>
<i>False</i>	<i>true</i>	<i>true</i>	<i>true</i>	<i>false</i>
<i>false</i>	<i>false</i>	<i>true</i>	<i>false</i>	<i>false</i>

Misol. Agar $a = 25$ va $b = 20$ bo'lsa:

a) $(a > 10)$ or $(b < 12)$ (*true*) b) $(a > 10)$ and $(b < 3)$ (*false*)

v) $(a > 10)$ or $(b > 12)$ (*true*) g) $(a > 10)$ and $(b > 12)$ (*true*)

Standart funksiyalar. Paskal tilida quyidagi standart funksiyalar mavjud.

Paskal tilida standart funksiyalar			
Funksiya	Belgilash	Argument turi	Natija turi
Absolyut qiymat	ABS(X)	REAL yoki INTEGER	REAL yoki INTEGER
Argtangens	ARCTAN(X)	----	REAL
Kosinus	COS(X)	----	----
Sinus	SIN(X)	----	----
Natural logarifm	LN(X)	----	----
Kvadrat ildiz	SQRT(X)	----	----
EkspONENTA	EXP(X)	----	----
Kvadratga ko'tarish	SQR(X)	----	INTEGER yoki REAL
Yaxlitlash	ROUND(X)	REAL	INTEGER
Argument butun qismi	TRUNC(X)	REAL	INTEGER
Juftlikni tekshirish	ODD(X)	INTEGER	BOOLEAN
Oldingi qiymat	PRED(X)	INTEGER	INTEGER
Navbatdagi qiymat	SUCC(X)	INTEGER	INTEGER

ODD(X) funksiyasining argumenti toq bo'lsa **TRUE**, aks holda **FALSE** natijani beradi. **PRED(I)** va **SUCC(I)** funksiyalar oldingi (**I-1**) va navbatdagi (**I+1**) qiymatlarni beradi.

Masalan. $PRED(7)=6$; $SUCC(23)=24$

Ifoda. Paskal tilida ifodalar o'zgarmlardan, o'zgaruvchilardan, funksiyalardan, amallar ishorasidan va qavslardan tuzilgan bo'ladi. Ifodalarda qatnashayotgan o'zgarmlar, o'zgaruvchilar va funksiyalar dasturda e'lon qilingan bo'lishi yoki standart nomiga ega bo'lishi kerak. Sonli qiymatlarni hisoblash uchun arifmetik ifoda qo'llaniladi. Arifmetik ifodada bo'lish va ko'paytirish amallari qo'shish va ayirish amallaridan oldin bajariladi. **Masalan:** $3 + 2*5 - 3$ ifodaning bajarilishidan 10 hosil bo'ladi.

Bir xil o'rinli amallar ishtirok etgan ifodalarda ular chapdan o'ngga qarab bajariladi.

Masalan. $4*6/3$ ifoda natijasi 8 ga teng.

Qavs ichiga olingan ifodalar birinchi navbatda bajariladi.

Masalan. $(7-2)*(5+4)$ ifoda natijasi 45 ga teng

Arifmetik ifodada haqiqiy va butun turdagi ma'lumotlar qatnashishi mumkin va hosil bo'lgan natija ham shunga mos ravishda haqiqiy yoki butun bo'ladi. Ifodalarni Paskal tilida yozilishiga misol keltiramiz.

Oddiy yozuvda	Paskal tilida
$\frac{a + 12,686}{c_2 - 1,87 \cdot 10^3} + \gamma$	$(A+12.686)/(S2-1.87E3)+GAMMA$
$\frac{ 2 \sin 4x + x }{3x}$	$EXP(ABS(2*SIN(4*X)+X)/(3*X)))$
$\frac{2^2 + \ln 1,13}{tg 2x}$	$(2*2*2+LN(1.13))/(SIN(2*X)/COS(2*X))$
$\frac{a^2 + 2^{1,17}}{\sqrt{x+y}}$	$(SQR(A)+EXP(1.17*LN(2)))/SQRT(X+Y)$

Mantiqiy ifoda. Mantiqiy ifodalar mantiqiy qiymatlar, mantiqiy va munosabat amallari orqali tuziladi. Munosabat amallarida arifmetik va mantiqiy ifodalar hamda belgili qiymatlar qatnashishi mumkin. Mantiqiy ifodaning natijasi **true** va **false** qiymat bo'lishi mumkin. Mantiqiy ifodada amallar quyidagi tartibda bajariladi:

1. **not** (yo'q)
2. ***** (ko'paytirish), **/** (bo'lish), **div**, **mod**, **and** (va)
3. **+** (qo'shish), **-** (ayirish), **or** (yoki);
4. **<** (kichik), **<=** (kichik yoki teng), **=** (teng), **<>** (teng emas), **>=** (katta yoki teng), **>** (katta);

Bitta satrda berilgan amallar bir xil bajarilish o'rniga ega. Mantiqiy ifodalarda faqat oddiy qavslar yoziladi.

Masalan. 1) $(A>3)$ and $(B=A+6)$ or not $(c=4)$

2) $(A<=x)$ and $(x<=B)$ or $(c<=x)$ and $(x<=d)$

Ma'lumotlar deganda dasturni tashkil etgan operatorlarning ishlashi uchun unda qatnashayotgan parametrlarga beriladigan yoki ular qabul qiladigan

qiymatlar tushuniladi. Dasturda **sonli, mantiqiy va harfiy qiymatlarni** aniqlaydigan ma'lumotlar ustida amallar bajariladi.

Dastur yozishda, uning hajmini qisqartirish maqsadida, ma'lumotlarning qiymatlari emas, balki ularning nomlari qo'llaniladi.

Hamma ma'lumotlar oddiy tuzilishga ega bo'lib, **skalyar (o'zgaruvchi) va murakkab** qismlarga bo'linadi. Paskal tilida skalyar ma'lumotlarning **4 turi** qo'llaniladi: butun (**INTEGER**), **haqiqiy (REAL)**, **mantiqiy (BOOLEAN)**, **belgili (CHAR) va satrli (STRING) (1-rasm)**.

Ma'lumotlar turi o'zgaruvchilarni, o'zgarmaslarni, funksiyalarni, ifodalarni qaysi turga tegishli ekanligini, ularni mashinada tasvirlanishini va ular ustida bajariladigan amallar natijalarining turini aniqlaydi.

Biz quyida **standart turlar** bilan tanishamiz.

Integer turi. **Integer turi** butun sonlar to'plamidagi qiymatlarni qabul qiladigan ma'lumotlarni aniqlaydi. **Maxint** standart identifikatori orqali mashinada tasvirlanadigan eng maksimal butun son beriladi.

Butun operandalar ustida quyidagi arifmetik amallarni bajarish mumkin, ya'ni bularning bajarilishidan hosil bo'lgan natija ham butun buladi: *** - ko'paytirish, div - butun hosil qilish uchun bo'lish; mod - qoldiqni hosil qilish amali; + - qo'shish, - - ayirish amali.**

div va mod amallarini ko'rib chiqaylik.

a) a div b.. Bu amal natijasi butun bo'ladi, ya'ni *a ni b ga bo'lishda hosil bo'lgan sonning butun qismi ajratiladi*

Masalan. $17 \text{ div } 2 = 8$; $19 \text{ div } 2 = 9$; $7 \text{ div } 17 = 0$.

b) a mod b. Bu amal *a va b ni bo'lishdan hosil bo'lgan qoldiq natijasini beradi.*

Masalan. $17 \text{ mod } 2 = 1$; $7 \text{ mod } 15 = 7$; $19 \text{ mod } 2 = 1$.

Munosabat amallarini ham butun operandalar uchun qo'llash mumkin, faqat ularning natijalari mantiqiy turga tegishli bo'lib, **chin(true) va yolg'on (false)** bo'lishi mumkin. Butun argumentli standart funksiyalarni bajarilishida butun yoki haqiqiy turga mansub natijalar bo'lishi mumkin.

Real turi. Haqiqiy sonlar to'plamidagi qiymatlarni qabul qiladigan ma'lumotlarni **real** turi aniqlaydi. Haqiqiy operandalar ustida quyidagi amallarni bajarish mumkin, ya'ni ularning bajarilishidan hosil bo'ladigan natija ham haqiqiy bo'ladi: **ko'paytirish, bo'lish, qo'shish, ayirish.** Bu amallarning bajarilishida qatnashayotgan operandalardan bittasi butun bo'lishi ham mumkin. Bo'lish amali ikkita butun son uchun ham o'rinli bo'ladi va hosil bo'lgan natija esa doim haqiqiy turga tegishli bo'ladi.

Haqiqiy natijani hosil qiluvchi funksiyalarda haqiqiy argumentlarni qo'llash mumkin: **abs(x), sqr(x), sin(x), cos(x), arctan(x), ln(x), exp(x), sqrt(x).**

Haqiqiy operandalar uchun qo'llaniladigan munosabat amallari mantiqiy natijani beradi: **chin yoki yolg'on.**

Boolean turi. Bu tur mantiqiy **chin(true) yoki yolg'on(false)** qiymatlarni qabul qiladigan ma'lumotlarni aniqlaydi. Mantiqiy operandalar uchun quyidagi amallarni qo'llash mumkin: **not - inkor, and - kon'yunksiya ("va"), or -**

diz'yunksiya ("yoki"). Paskal tilida mantiqiy tur quyidagicha aniqlangan: *false < true.*

Bu esa **integer** va **real** turdagi ma'lumotlar uchun munosabat amallarini qo'llash imkonini beradi.

Char va String turi. Bu tur Paskal tilining belgilarini va tartiblangan belgilar ketma-ketligini aniqlaydi. Bu belgilar soniga **bo'shliq ()** ham kiradi. Harfli o'zgaruvchilarning yoki o'zgarmaslarning qiy-mati har doim apostrof ichiga olingan bo'lishi kerak.

Masalan: *'aaa', 'xa-xa', 'IBM'-satr o'zgarmaslar, 'K', 'B'* - harfli o'zgaruvchilar qiymati. Agar harfli o'zgaruvchilarning qiymati bittadan ortiq belgidan iborat bo'lsa, u holda ular **string** turiga mansub bo'ladi va ular o'zgaruvchilarni tavsiflash bo'limida tavsiflanishi kerak bo'ladi. Satr o'zgaruvchilarining qiymati **255 tagacha** belgidan iborat bo'lishi mumkin. **String** funksiyasi orqali tavsiflanayotgan harfli o'zgaruvchining qiymati nechta belgidan iborat ekanligini, ya'ni mashina xotirasidan shu harfli o'zgaruvchi uchun nechta joy ajratish lozimligini ko'rsatish ham mumkin. Masalan:

1) var f: string [40]; dw, fd:string;

Bu yerda f harfli o'zgaruvchi 40 tagacha belgidan, dw va fd lar esa 255 tagacha belgidan iborat bo'lishi mumkinligi ko'rsatilgan.

2) const ter="Informatika"; tw="Matematika";

var sd, gf:string; pt:string[20];

Bu yerda ter, tw lar satr o'zgarmaslari, sd,gf va pt lar satr o'zgaruvchilaridir.

Sinov savollar:

1. Paskal tili qanday belgilardan iborat?
2. Identifikator nima va ular qanday hosil qilinadi?
3. Sonlar qanday tasvirlanadi?
4. Paskal tilida qanday amallar mavjud?
5. Qanday standart funksiyalarni bilasiz?
6. Ma'lumotlarning nechta standart turi ishlatiladi?
7. Paskal tilida dastur qanday tuzilgan?

Uyga vazifa : *Turbo-Paskal tizimida ifodalar va standart funksiyalar bilan ishlash qoidalarini mukammal o'rganish va ularni amaliyotga tatbiq etish.*

Mavzu: Paskal tilida dasturning umumiy tuzilishi

Reja:

1. Dastur va operator tushunchalari.
2. Paskal tilida dastur va uning umumiy tuzilishi.
3. Paskal tilida oddiy operatorlar.
4. Paskal tilida ta'minlash, ma'lumotlarni kiritish- chiqarish va tarkibiy operatorlar.

Darsning maqsadi:

1. Talabalarda dasturlash, uning turlari to'g'risida umumiy va asosiy bilimlarni hosil qilish.

2. Turbo-Paskal tizimi va uning umumiy tuzilishi to'g'risida mukammal tasavvurga ega bo'lish.

5. Turbo-Paskal tizimida mustaqil ishlash malakalarini shakllantirish.

Tayanch iboralar: *dastur, dastur sarlavhasi, dastur tanasi, operator, buyruq, oddiy va murakkab operatorlar, ta'minlash, kiritish va chiqarish operatorlari, chiziqli dastur.*

Dars o'tish vositalari: *sinf doskasi, o'quv-uslubiy qo'llanmalar, ma'ruza matnlari, komputer, ma'ruza bo'yicha slaydlar, Turbo-Paskal tizimi.*

Dars o'tish usuli: *namoyish dasturiy vositalat ishtirokida, jonli muloqotli ma'ruza.*

Darsning xrono xaritasi – 80 minut.

Tashkiliy qism: *xonaning tozaligi, jihozlanikishi, sanitariya holati. Talabalarning davomati*– 2 minut.

Talabalar bilimni baholash: *o'tilgan mavzuni qisqacha takrorlash, talaba bilan savol javob o'tkazish* - 10 minut.

Yangi mavzu bayoni - 55 minut.

Mavzuni o'zlashtirish darajasini aniqlash va mustahkamlash – 10 minut.

Sinov savollari – 5 minut.

Uyga vazifa berish – 3 minut.

Ma'ruza bayoni

Paskal tilida **dastur** bitta blokni tashkil qiluvchi **dastur sarlavhasi va dastur tanasidan** iborat bo'ladi (2- rasm). Dastur sarlavhasida **program** xizmatchi so'zidan keyin dasturning nomi beriladi. Paskal tilida dastur sarlavhasi quyidagicha umumiy ko'rinishga ega: **<dastur sarlavhasi> := program <dastur nomi>;**

Dastur tarkibiga uning ayrim qismlarini bajarilish tartibini tushuntiruvchi **izohlashlarni** ham kiritish mumkin. Izohlashlar shaklli qavslar ichiga olingan bo'lib, dasturning ixtiyoriy joyida berilishi mumkin.

Dastur tanasi (bloki). Paskal tilida tuzilgan dasturning asosiy qismi uning **tanasi- bloki** hisoblanadi. Umumiy holda dastur bloki berilishi tartibi qat'iy saqlangan **oltita bo'limdan** iborat:

< blok (dastur tanasi)>:= <belgilarni aniqlash bo'limi>

<o'zgarmaslarni aniqlash bo'limi>

<turlarni aniqlash bo'limi>

<o'zgaruvchilarni tavsiflash>

<procedura va funksiyalar>

<operatorlar bo'limi>

Dasturga tegishli hamma ma'lumotlar Paskal tilida mavjud ma'lumotlar turidan hech bo'lmaganda bittasiga tegishli bo'lishi kerak.

Dasturning blokini tashkil etgan bo'limlar bilan tanishib chiqamiz.

Belgilarni (nishonlarni) aniqlash bo'limi. Ixtiyoriy operatorni boshqa operatorlar orasida ajratib ko'rsatish mumkin. Buning uchun bu operatorning oldi-

ga ikki nuqta bilan **belgi (nishon)** yoziladi. Bunday operatorlar **belgilangan (nishonlangan)** operatorlar deyiladi. Operator faqat bitta **belgi (nishon)** bilan belgilanadi va ularning nishonlari har xil bo'lishi kerak. Nishon belgilangan operatorlarning nomi vazifasini bajaradi. Paskal tilida nishon sifatida manfiy bo'lmagan, [0,9999] oraliqdagi butun sonlardan foydalaniladi. **Masalan: 67 : Z:= A + B**

Dasturda ishlatiladigan har qanday nishon nishonlarni aniqlash bo'limida ilgari e'lon qilingan bo'lishi kerak. Bu bo'lim **label (nishon)** xizmatchi so'zidan boshlanadi. Shundan keyin dasturda nishon sifatida ishlatiladigan butun sonlar beriladi. Bo'sh bo'lmagan nishonlar qismi nuqta vergul bilan tugaydi:

<nishonlar bo'limi>:= label<nishon> {<nishon>;

Masalan: label 157,96, 1654;

O'zgaraslarni aniqlash bo'limi. Bu bo'lim **const** xizmatchi so'zidan boshlanib, undan keyin dasturda foydalaniladigan **o'zgaraslar (konstantalar)** nomi va qiymati keltiriladi. O'zgaras nomi bilan uning qiymati (=) belgisi bilan bog'lanadi. Bir o'zgaras boshqa bir o'zgarasdan (;) belgisi bilan ajratiladi. Hamma o'zgaraslarni tavsiflash Paskal- dasturining o'zgaraslar bo'limida berilishi kerak:

<o'zgaraslar bo'limi>:= const <o'zgaraslarni tavsiflash>;

Paskal tilida o'zgaras **butun, haqiqiy va satrli** bo'lishi mumkin. **Masalan: const pi=3.14; wt= 5.7654; text='Qiymat x=';**

Turlarni aniqlash bo'limi. Yuqorida ta'kidlaganimizdek, Paskal tilida qiymatlarni quyidagicha **standart turlari** mavjud: **integer, real, boolean va char.** Dasturlash jarayonida bu standart turlardan tashqari, boshqa qiymatlar turini ham ishlatish mumkin. Bu turlar dasturda ochiq aniqlangan bo'lishi kerak. Foydalanish uchun kiritilayotgan turlarga nom berilishi kerak. Standart turlar esa o'z nomiga ega:

<turlarni aniqlash>:= <turning nomi> = <tur>;

<tur> := <turning nomi>/<turning berilishi>;

Hamma turlarni tavsiflash turlar bo'limida berilishi kerak. Bu bo'lim **type (tur)** xizmatchi so'zi bilan boshlanadi. Shundan keyin bir-biridan **nuqta - vergul (;)** bilan ajratilgan turlarni tavsiflash beriladi: **<turlar bo'limi>:=type<turlarni tavsiflash>;<turlarni tavsiflash>;**

Masalan:

type

mantiq := boolean; hafta:=(dush,sesh,chor,pay,juma,shan,jak);

ishkuni := dush .. shan;

O'zgaruvchilarni aniqlash bo'limi. Paskal tilida tuzilgan har qanday dasturda albatta o'zgaruvchilar qatnashadi. **O'zgaruvchi deganda** qiymat qabul qilish imkoniyatiga ega bo'lgan obyektlar tushuniladi. O'zgaruvchilarning qiymatlari dastur bajarilishigacha ma'lum bo'ladi va dastur bajarilish jarayonida ularga ta'min etiladi.

Dastur tuzish jarayonida har bir o'zgaruvchi o'z nomiga va qabul qila-digan qiymatiga ega bo'ladi. Dasturda qo'llaniladigan o'zgaruvchilar e'lon qilinishi va

ularga tavsif berilishi kerak. Bu o'zgaruvchilarni aniqlash bo'limida amalga oshiriladi:

<o'zgaruvchilarni tavsiflash> := <o'zgaruvchi nomi>:<tur>

Masalan: *x, y, z, h:real;*

Bu tavsiflash *x,y,z,h* nomli o'zgaruvchilarni haqiqiy ekanini aniq-laydi.

Barcha o'zgaruvchilarni tavsiflash bo'limi **var** (*Variable* - *o'zgaruvchi*) xizmatchi so'zi bilan boshlanadi va shundan keyin o'zgaruvchilar nomi hamda turi ko'rsatiladi:

<o'zgaruvchilar bo'limi> := var<o'zgaruvchilarni tavsiflash>;

Masalan: *var*

i,j,k:integer; a,b,c:real;

r,q : Boolean; m,t,z:char;

day,tay:hafta;

Dasturda ishlatiladigan har bir o'zgaruvchiga bir marta tavsif berilishi kerak. Ularni kayta tavsiflanishi dasturda xatolikka olib keladi. {*Procedura va funksiyalar bo'limi bilan keyingi paragraflarda tuliq tanishamiz*}.

Operatorlar bo'limi. Dasturning asosiy maqsadi-ma'lumotlarni qayta ishlash uchun mashina bajarishi lozim bo'lgan amallar ketma-ketligini ko'rsatishdan iborat. Bunday amallar **operatorlar** yordamida bajariladi. Shuning uchun dasturda operatorlar bo'limi eng asosiy bo'limlardan biri hisoblanadi va ular har bir dasturda qatnashishi shart. Chunki shu bulimda dasturda bajarilishi kerak bo'lgan operatorlar ketma-ketligi keltiriladi. *Operatorlar bo'limi* quyidagi ko'rinishda aniqlanadi:

<operatorlar bo'limi> := begin <operator yoki operatorlar> end.

Dasturning bajarilishi operatorlar bo'limining bajarilishiga, ya'ni **begin** va **end** xizmatchi so'zlari orasidagi operatorlar ketma-ketli-gining bajarilishiga olib keladi.

Program

<nom> (<fayl nomi>,...,<fayl nomi>);{dastur sarlavhasi}

label

<nishon>,...<nishon>; {nishonlar bo'limi}

const

<o'zgaruvchi> = <o'zgarmas>; { o'zgarmaslar bo'limi}

type

<tur nomi> = <tur>; {turlar bo'limi}

var

<o'zgaruvchi nomi> ,.. ,<o'z. nomi>:<tur>; {o'zgaruvchi bo'limi}

procedure

<procedura sarlavhasi>; <procedura bloki>; {procedura bo'limi}

function

<funksiya sarlavhasi>; <funksiya bloki>; {funksiya bo'limi}

begin

<operator>; <operator>; {operatorlar bo'limi}

end.

Sinov uchun savollar:

1. Paskal tilida dastur va uning tuzilishi qanday?
2. Dastur sarlovxasi nimadan iborat?
3. Dastur tanasi qanday tuzilgan?
4. Turlar va o'zgaruvchilar bo'limini izoxlab bering?
5. O'zgarmaslar bo'limini tushuntirib bering?
6. Operatorlar bo'limi nima uchun kerak?

Uyga vazifa : Turbo-Paskal tizimida dasturning umumiy tuzilishini mukammal o'rganish va ularni amaloyotga tatbiq etish.

13-ma'ruza

Mavzu: Paskal tilida oddiy operatorlar.

Reja:

1. Operator tushunchasi va ularning turlari.
2. Ta'minlash, ma'lumotlarni kiritish va chiqarish operatorlari.
3. Tarkibiy operatorlar.
4. Chiziqli dastur tuzish.

Darsning maqsadi:

1. Talabalarda operatorlar to'g'risida umumiy va asosiy bilimlarni hosil qilish.
2. Turbo-Paskal tizimida ta'minlash, ma'lumotlarni kiritish va chiqarish operatorlari to'g'risida amaliy ko'nikmaga ega bo'lish.
3. Chiziqli dastur tuzish malakalarini shakllantirish.
4. Turbo-Paskal tizimida mustaqil ishlash malakalarini shakllantirish.

Tayanch iboralar: dastur, dastur sarlavhasi, dastur tanasi, operator, buyruq, oddiy va murakkab operatorlar, ta'minlash, kiritish va chiqarish operatorlari, chiziqli dastur.

Dars o'tish vositalari: sinf doskasi, o'quv-uslubiy qo'llanmalar, ma'ruza matnlari, komputer, ma'ruza bo'yicha slaydlar, Turbo-Paskal tizimi .

Dars o'tish usuli: namoyish dasturiy vositalat ishtirokida, jonli muloqotli ma'ruza .

Darsning xrono xaritasi – 80 minut.

Tashkiliy qism: xonaning tozaligi, jihozlanikishi, sanitariya holati. Talabalarining davomati– 2 minut.

Talabalar bilimni baholash: o'tilgan mavzuni qisqacha takrorlash, talaba bilan savol javob o'tkazish - 10 minut.

Yangi mavzu bayoni - 55 minut.

Mavzuni o'zlashtirish darajasini aniqlash va mustahkamlash – 10 minut.

Sinov savollari – 5 minut.

Uyga vazifa berish – 3 minut.

Ma'ruza bayoni

Paskal tilida **operatorlar oddiy va murakkab** bo'ladi. **Oddiy operatorlar** ichida boshqa operatorlar qatnashmaydi. **Murakkab operatorlar** bir nechta oddiy operatorlardan tashkil topgan bo'ladi. Paskal tilida oddiy operatorlarga **ta'minlash operatori, o'tish operatori, bo'sh operatori va procedura operatorlari** kiradi. Murakkab operatorlar **shartli operator, sikl operatorlari, variant**

(tanlash) operatori va birlashtirish operatorlaridan iborat. Paskal tilining asosiy operatorlari bilan tanishib chiqamiz.

Ta'minlash operatori. Hisoblash natijasining ba'zi bir qiymatlarini hisoblash jarayonining keyingi bosqichlarida qo'llash va ularni eslab qolish kerak bo'ladi. Bu esa aniq yoki hisoblangan qiymatni ma'lum bir o'zgaruvchiga ta'minlash orqali bajariladi. Bu amalni bajarish uchun asosiy operatorlardan biri, **ta'minlash operatori** qo'llaniladi. Ta'minlash operatorining umumiy ko'rinishi quyidagicha:

$\langle ta'minlash\ operatori \rangle ::= \langle o'zgaruvchi \rangle := \langle ifoda \rangle$

Bu yerda "==" belgisi ta'minlash amalini bildiradi. Bu belgini taqqoslash amali "=" bilan chalkashtirmaslik kerak.

Ta'minlash operatori bajarilganda, avval o'ng tomonda turgan ifodaning qiymati hisoblanadi, so'ngra hosil bo'lgan qiymat chap tomonda turgan o'zgaruvchi tomonidan o'zlashtiriladi, ya'ni bu operatorning bajarilishi natijasida ma'lum bir o'zgaruvchi yangi qiymatni qabul qiladi.

Ta'minlash operatorining umumiy ko'rinishi quyidagicha: $V:=A$;

Bu yerda **V - o'zgaruvchi**, **A**- arifmetik yoki mantiqiy yoki satrli ifoda bo'lishi mumkin. Ta'minlash operatorida **ifodaning turi** bilan **o'zgaruvchining turi** doim mos kelishi kerak. Ta'minlash operatorining o'ng tomonida turgan **ifodaning turiga** qarab, bu operator **arifmetik, mantiqiy va satrli ta'minlash operatorlariga** bo'linadi.

Arifmetik ta'minlash operatori. Arifmetik ta'minlash operatori arifmetik turdagi, ya'ni **real va integer** turdagi o'zgaruvchiga qiymatni ta'minlash uchun xizmat qiladi. Shu munosabat bilan bu operatorning o'ng tomonidagi ifoda - arifmetik ifodadan iborat bo'lishi kerak.

Masalan: $x := x+2$;

$s:=a*a + b*b$; $i:=i+1$;

$t:=sqrt(a*a + b*b)$;

Mantiqiy ta'minlash operatori. Agar ta'minlash operatorida ifoda mantiqiy turga, ya'ni **boolean** turiga mansub bo'lsa, u holda bu operator **mantiqiy ta'minlash operatori** hisoblanadi.

Mantiqiy ifoda munosabat (**<, >, =, <=, >=**) amallari orqali yoziladi. Ifodada berilgan munosabat bajarilsa, ifoda qiymati **chin, ya'ni true** qiymatga, aks holda esa **yolg'on - false** qiymatga ega bo'ladi.

Masalan: $d:=true$;

$b:=(z>z1) and (x<o)$;

$c:=d or and not (odd(k) and b)$;

$y:=(k div 2*(c+d)>=x-3) or not(sqrt(a+b)>=3)$;

Satrli (belgili) ta'minlash operatori. Agar o'zlashtirish operatorida o'zgaruvchi satrli (belgili), ya'ni **char yoki string** turiga mansub bo'lsa, u holda o'ng tomonda turgan ifoda ham satrli va **char** turiga mos bo'ladi. Bu ifodaning bajarilishidan hosil bo'lgan natija ham **char yoki string** turiga tegishli bo'ladi. Shuning uchun satrli ifoda shu turga tegishli bo'lgan **o'zgarmas, o'zgaruvchi va funksiyadan** iborat bo'lishi mumkin.

Masalan: $sym := 'samarkand'$; $beta:='SamDU'$;

Alpha := sym + betta ;

Paskal tilida yozilgan dasturlarni kompyuterda bajarish uchun unda qatnashayotgan va qiymati hozircha noma'lum o'zgaruvchilarga aniq qiymat-larni kiritishga to'g'ri keladi. Buning uchun kiritish operatori qo'lla-niladi. Kiritish op-eratorlarining umumiy ko'rinishi quyidagicha:

read (b1,b2,...,bn);
readln (b1,b2,...,bn);
readln;

bu yerda ***b1, b2, ..., bn*** lar qiymati kiritilishi talab etilayotgan o'zgaruvchilarning nomi.

Read (b1,b2,...,bn); operatori ma'lumotlarni kiritishni ta'minlaydi, natijada ***b1, b2, ..., bn*** o'zgaruvchilar mos qiymatlarni oladilar. Kiritilayotgan o'zgaruvchilar turi berilayotgan qiymatlar turi bilan mos kelishi kerak.

Readln (b1,b2,...,bn); operatori ma'lumotlarni ***b1, b1, ..., bn*** o'zgaruvchilarga ta'minlash uchun ishlatiladi va boshqaruvni (*kursorni*) yangi satr boshiga o'tkazishni amalga oshiradi.

Readln; operatori bitta bo'sh satrni o'tkazib yuborishni va yangi satr bosh-iga o'tishni ta'minlaydi.

Masalan: ***read (i,j);***
readln (k,l);
read (m,n);

Qiymatlar kiritish jarayonida haqiqiy o'zgaruvchiga harfiy qiymat yoki har-
fli o'zgaruvchiga haqiqiy qiymat mos keltirilsa, u holda kompyuter xatolik ro'y
berganligi tug'risida (***Type mismatch error***) ma'lumot beradi.

Dasturning bajarilish natijasida hosil bo'lgan ma'lumotlarni kompyuter
ekraniga yoki chop etish qurilmasiga chiqarish uchun ***chiqarish operatorlari***
qo'llaniladi. Bu operatorlar orqali dastur bajarilishidan hosil bo'lgan
o'zgaruvchilarning qiymatlari, natijalar va ixtiyoriy matnlar displey ekraniga yoki
chop etish qurilmasi orqali qog'ozga chiqariladi.

Chiqarish operatorlarining umumiy ko'rinishi quyidagicha:

write (a1,a2,...,an);
writeln (a2,a2,...,an);
writeln;

bu yerda ***a1, a2, ..., an*** lar qiymati chiqarilishi kerak bo'lgan o'zgaruvchilar nomi.

Write (a1,a2,...,an); operatori ***a1,a2,...,an*** o'zgaruvchilarga mos qiymatlarni
chiqarish vazifasini bajaradi. Chiqarilayotgan qiymatlar bitta satrga joylashtiri-
ladi.

Writeln (a1,a2,...,an); operatori ***a1,a2,...,an*** o'zgaruvchilarga mos
qiymatlarni chiqaradi. Oxirgi qiymat chiqarilib bo'lgandan keyin yangi satriga
o'tishni amalga oshiradi.

Writeln; operatori bitta bo'sh satrni o'tkazib yuborishni va keyingi satr
boshiga o'tishni ta'minlaydi.

Masalan: ***write (i,j);***
writeln(k,r) ;
writeln (r1,t1);

Chiqarish operatori orqali o'zgaraslarni, o'zgaruvchilarning qiymatlarini, ifodalarni va matnlarni chiqarish mumkin.

Dasturning bajarilish natijasida hosil bo'lgan ma'lumotlarni ma'lum bir tartibda, satrlarni aniq o'rnida yoki jadval ko'rinishida chiqarish va ularni tahrirlash mumkin.

Chiqarilayotgan ma'lumotlar uchun ajratiladigan joy chiqarish operatorida quyidagicha tartibda ko'rsatiladi:

write (a:m);

writeln (a:m);

bu yerda **a** - chiqarilayotgan o'zgaruvchi nomi; **m** - butun o'zgarma yoki ifoda bo'lib, chiqarilayotgan qiymat uchun ajratilgan joyni bildiradi.

Masalan: write (ad:4, at:6);

Bu yerda **ad** va **at** o'zgaruvchilarning qiymati bitta satrga joylashadi va mos ravishda **4 ta** va **6 ta** joyni egallaydi. Har bir qiymat ajratilgan o'rinda joylashadi va o'ng tomondagi chetki o'rinlarni egallaydi. Bunda to'lmagan o'rinlar bo'sh qoladi va bo'shliqlarni hosil qiladi. Agar o'zgaruvchilarning qiymati ajratilgan joyga sig'may qolsa, u holda o'rinlar soni avtomatik ravishda ko'paytiriladi.

Qo'zg'aluvchan vergulli haqiqiy turli qiymatlarni chiqarishda, chiqarilayotgan qiymatning hammasi va sonning kasr qismi uchun ajratilgan o'rinlar ko'rsatiladi.

Bunda chiqarish operatorining umumiy ko'rinishi quyidagicha bo'ladi:

write (a:m:n);

writeln (a:m:n);

bu yerda **a** - chiqarilayotgan o'zgaruvchi nomi; **m** - chiqarilayotgan qiymat uchun ajratilgan o'rinlar soni; **n** - sonning kasr qismi uchun ajratilgan o'rinlar soni.

Masalan : write (AD:8:3); Bu holda AD qiymat uchun **8 ta** o'rin, shundan **3 tasi** sonning **kasr qismi** uchun ajratiladi.

Agar haqiqiy qiymatni chiqarishda uning kasr uchun ajratilgan o'rin ko'rsatilmasa, u holda natija o'nli tartibdagi normal ko'rinishda hosil bo'ladi.

Bir nechta qiymatlarni bir satrga chiqarishda, ko'rgazmaliroq bo'lishi uchun, bu qiymatlar orasiga bo'shliqlar sonini berish mumkin. Buning uchun chiqarish operatorida quyidagicha yozish kerak:

write (' ' : n); bu yerda **n** - butun tipli o'zgarma bo'lib, bo'shliqlar sonini ko'rsatadi.

Masalan: write (A,":4,B); Bu holda A va B qiymatlari orasida **4 ta bo'shliq** hosil qilinadi.

Quyidagi misol ma'lumotlarni chiqarishda ularni turli xil ko'rinishda joylashtirilishi mumkinligini ko'rsatadi.

program misol;

const pi=3.141592; t=401;

w = true; sim ='dad';

begin

writeln(pi:8:6); writeln(t:6,":5,w:4,":6,sim:3)

end.

Ma'lumotlar ikki satrda quyidagi ko'rinishda chiqariladi:

3.141592

401_____true_____dad

Chiqarish operatorida chiqarilayotgan ma'lumotlarni izohlash uchun o'zgaruvchilar ro'yxatiga belgilar satrini kiritish mumkin. Bunday holda belgilar satri chiqarish ro'yhatining kerakli joyiga to'liq qo'yiladi va chop etiladi. Belgilar satridan jadvallar va matnlarni sarlavhasini, chiqari-ladigan ma'lumotlarga izohlar berishni hosil qilish mumkin, ya'ni natijalar o'z nomi va sarlavhasi yoki izohlari bilan chiqarilishi mumkin.

program ISOHLI;

const

A = 372; X = 3284.323;

Y = 73.21E2; G = true;

C = 'BAB' ; M =10; N= 3;

begin

writeln ('***O'zgarmaslar*****');**

writeln ('A=',A:8);

writeln ('X=',X:M:N);

writeln ('Y=',Y:M);

writeln ('G=',G:5,'belgi',C:3)

yend.

Chiqarilayotgan ma'lumotlar quyidagicha ko'rinishda bo'ladi:

*******O'zgarmaslar*******

A= 372

X=-3284.323

Y=7.321E+03

G=true belgi BAB

Tarkibiy operatorlar. Paskal tilida tarkibiy operatorlar **begin** xizmatchi so'zidan boshlanib, **end** xizmatchi so'zi bilan tugaydigan operatorlar ketma - ketligidan iborat bo'ladi. **Begin va end** xizmatchi so'zlari tarkibiy operatorlarning boshlanishi va tugallanishini chegaralovchi qavslar vazifasini bajaradi. Paskal tilida operatorlar ketma - ketligi bir -biridan **nuqta vergul (;)** bilan ajratiladi. Operatorlar ketma- ketligining oxirida **end** xizmatchi so'zidan oldin **nuqta vergul (;)** qo'yilmasligi ham mumkin.

Tarkibiy operatorlarning umumiy ko'rinishi qo'yidagicha:

begin <operator> ; <operator> ;...; <operator> end.

Tarkibiy operatorning bajarilishi shundan iboratki, **begin va end** operatorlari ichidagi operatorlar qanday ketma - ketlikda yozilgan bo'lsa, shunday tartibda, bir - biridan keyin, operatorlarning hammasi tugaguncha bajariladi.

Dasturning oxirida **end** operatoridan keyin nuqta qo'yiladi.

Misol. $S = \sqrt{p(p-a)(p-b)(p-c)}$ Geron formulasidan foydalanib uchburchak yuzini hisoblash dasturining tuzing, bu yerda $p=(a+b+c)/2$.

Hisoblashda **a, b, c, p va S** haqiqiy turdagi miqdorlar qatnashayapdi. O'zgaruvchilar nomi o'zgaruvchilarni tavsiflash bo'limida tavsiflanadi. Operator-

lar bo'limida eng avval **a,b,c** qiymatlarni kiritish operatori, so'ngra **p** va **S** larni hisoblash operatorlari va **S** natijani chiqarish operatori yoziladi.

Dasturning umumiy ko'rinishi quyidagicha bo'ladi:

```
program SSS;  
  var a,b,c,p,S:real;  
  begin  
    read (a,b,c); p:=(a + b + c)/2;  
    S:=sqrt(p*(p-a)*(p-b)*(p-c));  
    writeln (' S = ',S)  
  end.
```

Sinov savollari

1. Dasturda izohlashlar qanday beriladi?
2. Ta'minlash operatori qanday vazifani bajaradi?
3. Kiritish operatori qanday ko'rinishga ega? Uning ta'minlash operatoridan farqi nimadan iborat?
4. Dasturlashda chiqarish operatoridan nima uchun va qanday foydalaniladi?
5. Tarkibiy operatorlarning asosiy vazifasi nimadan iborat?

Uyga vazifa : Turbo-Paskal chiqli dastur tuzish qoidalarini mukammal o'rganish va ularga doir masalalar tuzish va ularni yechish.

14-ma'ruza

Mavzu: Paskal tilida tarmoqlanuvchi jarayonlarni dasturlash

Reja:

1. Tarmoqlanuvchi jarayonlar.
2. Shartsiz va shartli o'tish operatori.
3. Tanlash operatori.
4. Tarmoqlanuvchi jarayonlar uchun dastur.

Darsning maqsadi:

1. Talabalarda tarmoqlanuvchi jarayonlar dasturlash, uning turlari to'g'risida umumiy va amaliy bilimlarni hosil qilish.
2. Turbo-Paskal tizimida o'tish va shartli operatorlar, tanlash operatori va ularning uning umumiy tuzilishi to'g'risida mukammal tasavvurga ega bo'lish.
3. Turbo-Paskal tizimida tarmoqlanuvchi dastur tuzish bo'yicha mustaqil ishlash malakalarini shakllantirish.

Tayanch iboralar: tarmoqlanuvchi jarayon, o'tish va shartli operatorlar, tanlash operatori, tarmoqlanuvchi dastur.

Dars o'tish vositalari: sinf doskasi, o'quv-uslubiy qo'llanmalar, ma'ruza matnlari, komputer, ma'ruza bo'yicha slaydlar, Turbo-Paskal tizimi .

Dars o'tish usuli: namoyish dasturiy vositalat ishtirokida, jonli muloqotli ma'ruza .

Darsning xrono xaritasi – 80 minut.

Tashkiliy qism: xonaning tozaligi, jihozlanikishi, sanitariya holati. Talabalarining davomati– 2 minut.

Talabalar bilimini baholash: o'tilgan mavzuni qisqacha takrorlash, talabalar bilan savol-javob o'tkazish - 10 minut.

Yangi mavzu bayoni - 55 minut.

Mavzuni o'zlashtirish darajasini aniqlash va mustahkamlash – 10 minut.

Sinov savollari – 5 minut.

Uyga vazifa berish – 3 minut.

Ma'ruza bayoni

Tarmoqlanuvchi hisoblash jarayonlarida operatorlarning tabiiy ketma-ket bajarilish tartibi buziladi va qo'yilgan shartning bajarilishiga qarab bir nechta tarmoqqa bo'linadi. Hisoblash jarayonida ana shu tarmoqlardan faqat bittasi bajariladi. Bunday jarayonlar uchun Paskal tilida dastur tuzishda **shartli o'tish, shartsiz o'tish va tanlash operatorlaridan** foydalaniladi.

O'tish operatori. Bu operator hyech imkon bo'lmay qolgan holda dasturning tabiiy bajarilishini buzib, boshqaruvni dasturning boshqa joyiga uzatish uchun xizmat qiladi.

O'tish operatori qo'yidagicha beriladi:

o'tish operatori := **goto** <**nishon**>;

bu yerda **goto**(..**ga o'tish**) - operator nomi, **nishon** - to'rtta raqamdan oshmagan musbat butun son.

Masalan: **goto 25 ;**

goto 674 ;

Operatorlarda joylashtirilgan har bir **nishon** nishonlar bo'limida e'lon qilinishi kerak.

Masalan: **label 25, 0, 674;**

Paskal tilida o'tish operatoridan foydalanish ma'lum miqdorda chegaralangan: *bu operator yordamida hosil qilingan ixtiyoriy operator ichiga o'tish, hamda tanlovchi operatorlarda bir alternativadan boshqasiga o'tish ta'qiqlanadi.*

Masalan: **program X;**
label 2,98,100,14;
const A = 74;
var y, z, q: real; l:boolean;
begin
readln(z,q);
.....
.....
goto 98;
100 :y:=a + z/q
.....
.....
14: readln(z,q);
.....
2: y:=sqrt(y);
.....
98:l:=false ;
end.

Bu dasturning bajarilishi natijasida **goto 98** operatori boshqaruvni **98 nishon** bilan turgan operatorga uzatadi, oraliqdagi qolgan operatorlar esa bajarilmasdan qoladi.

Bo'sh operator. *Bo'sh operator* hych qanday amalni bajarmaydi va dasturning o'rtasidan yoki tarkibiy operatorlardan chiqish uchun ishlatiladi. Bo'sh operator nishon bilan yoziladi. **Masalan,** *dasturning oxiriga chiqish uchun bo'sh operatori quyidagicha yoziladi:* **40: end.**

Shartli operator. *Shartli operator* Paskal tilida tuzilgan dasturda qo'yilgan shartning bajarilishi yoki bajarilmasligiga ko'ra, boshqaruvni berilgan operatorlardan bittasiga bajarish uchun o'tkazadi. Bu operator asosan bizga ma'lum bo'lgan tarmoqlanuvchi algoritmlarni dasturlashda qo'llaniladi, ya'ni bu jarayonda hisoblash jarayonining ketma-ket tartib bilan bajarilishi buziladi va bir nechta tarmoqqa bo'linadi.

Paskal tilida shartli operatorning ikkita ko'rinishi mavjud: **to'liq va qisqartirilgan.**

To'liq shartli operatorning umumiy ko'rinishi quyidagicha: **<to'liq shartli operator>:=if<mantiqiy ifoda> then <operator> else <operator>;**

Bu yerda **if** xizmatchi so'z bo'lib, "**agar**" degan ma'noni bildiradi; **then** - xizmatchi so'z bo'lib "**u holda**" degan ma'noni bildiradi; **else** - hizmatchi so'z "**aks holda**" degan ma'noni anglatadi.

Shunday qilib to'liq shartli operatorning ko'rinishini quyidagicha tasvirlash mumkin: **if B then S1 else S2;** Bu yerda **B** mantiqiy ifoda; **S1 va S2 lar** operatorlar.

Shartli operator quyidagicha tartibda bajariladi: *agar if operatoridan keyin turgan B mantiqiy ifoda chin qiymatni qabul qilsa, u holda S1 operator bajariladi, aks holda ya'ni B ifoda yolg'on qiymatni qabul qilsa, S2 operator bajariladi.*

Hisoblash jarayonining bajarilishi bir nechta shartga bog'liq bo'lsa, u holda bu shartlarning hammasini **IF** operatorida yozish uchun **mantiqiy amallar (and, or, not)dan** foydalaniladi.

Masalan. a) if $x < 0$ then $i := i + 1$ else $k := k + 1$;

b) if $(x < y)$ and d then $S := S + i$ else $S := S * i$;

v) if $a \geq 0$ then $y := \text{sqrt}(a + 1)$ else $y := \text{sqr}(a)$;

Ayrim hisoblash algoritmlarini dasturlashda shunday holat mavjud bo'ladiki, bunda qo'yilgan shartning bajarilishida ma'lum operatorlar ijro etiladi, aks holda esa hych qanday amal bajarilmaydi. Bunday holatda shartli operatorning qisqartirilgan ko'rinishidan foydalaniladi:

<qisqa shartli operator>::=if<mantiqiy ifoda>then<operator> yoki buni operatorlar yordamida quyidagicha yozish mumkin: **if B then S ;.**

Bu yerda operator quyidagi tartibda bajariladi: *agar mantiqiy ifodaning qiymati chin(true) bo'lsa, u holda S operator bajariladi, aks holda hych qanday amal bajarilmaydi.*

Masalan: if $x < y$ then begin $x := y$; $y := x$ end;

Misol. *Ixtiyoriy berilgan x va y sonlaridan eng kattasini topish dasturini tuzing.*

```
program max;  
var x,y,z:real;  
begin read(x,y);
```

```

begin
  if  $x \geq y$  then  $z := x$  else  $z := y$ ;
  write (z) end;
end.

```

Shartli operatorlar **ichma- ich joylashgan** ko'rinishda ham bo'lishi mumkin: **if B1 then S1 else if B2 then S2 else S3**. Bu yerda **B1, B2** - mantiqiy ifodalari, **S1, S2, S3** - operatorlar. Bu operatorlarda **if B2 then S1 else S2** ichki operator **B1** mantiqiy ifodaning qiymati *yolg'on(false)* bo'lsa bajariladi.

Ichma- ich joylashgan shartli operatorning boshqacha ko'rinishi quyidagicha: **if B1 then if B2 then S1 else S2 else S3**;

Ichki joylashgan **if B2 then S1 else S2** operator **B1** ifoda *chin(true)* qiymatga ega bo'lsa bajariladi va buni Paskalda quyidagicha yozish maqsadga muvofiq: **if B1 then begin if B2 then S2 else S3 end**;

Masalan: **if d or b then begin if $x < 0$ then $x := 2 + x$ end
 else begin if $x < y$ then $x := x + 0.5$ else $y := y + 0.5$; end**

Misol. Talabalarni test yakunlari bo'yicha baholash dasturini tuzing. (eng yuqori bal $x=100$).

```

program test;
var x:real;y:string[10];
begin read(x); if  $x \geq 85$  then  $y := 'a'lo'$  else
  begin if ( $x < 85$ ) and ( $x \geq 75$ ) then  $y := 'yaxshi'$ 
    else begin if ( $x < 75$ ) and ( $x \geq 55$ ) then  $y := 'qoniqarli'$ 
      else  $y := 'qoniqarsiz'$  end; end;
  write(y)
end.

```

Tanlash (variant) operatori. Algoritmning xarakterli xususiyati shundan iboratki, ular uchun berilgan hisoblash jarayonlari ko'p tarmoqli bo'ladi. Bunday hisoblash jarayonlarini Paskal tilida yozishning usullaridan biri bizga ma'lum bo'lgan operatoridir. Agar hisoblash jarayonida qo'yilgan shartlar soni ikkitadan ortiq bo'lsa, u holda shartli operatoridan foydalanish ancha qiyinchilik tug'diradi. Ana shunday hollarda Paskal tilida tanlash (variant) operatori qo'llaniladi.

Tanlash operatori Paskal tilida hosil qilinadigan operatorlardan hisoblanib, shartli operatorning umumlashmasidan iboratdir.

Tanlash operatori selektorning qiymatiga qarab, dasturda berilgan amallardan birini tanlaydi.

Tanlash operatorining umumiy ko'rinishi quyidagicha:

<tanlash operatori> ::= case <selektor operatori> of <tanlash ro'yxati elementlari> end.

Masalan: **case c of
 n1:p1;
 n2:p2 ;

 nn:pn;
 end**

bu yerda **c** - selektor; **n1,n2...,nn** - operatorlar nishoni; **p1,p2,...pn** – bajariladigan operatorlar.

Masalan: 1) case i mod 3 of

0: m:=0;

1: m:=-1;

2: m:= 1;

3: m:=1;

end

2) case kun of

dush,zesh,chor,pay,juma:writeln('ish kuni');

shanba,jak:writeln ('dam olish kuni');

end

if B then S1 else S2 shartli operator quyidagi tanlash operatoriga ekvivalent:

case B then

true: S1;

false:S2;

end ;

Masalan: case x<=0 then

true: y:=sqrt(sqr(x)-x+2);

false: y:=ln(x+3);

end

Selektor tanlash nishonlaridan bitta qiymatini qabul qiladi. Qabul qilingan qiymatga qarab nishonlardan bittasiga o'tish amalga oshiriladi va operatorlardan bittasi bajariladi. Agar selektor qiymat tanlash nishonlaridan birontasiga ham mos kelmasa, u holda case operatori aniqlanmaydi, ya'ni dasturning bajarilishida xatolik ro'y beradi.

Masalan: case k of

2: y:= g+1;

4: y:=g*x;

6: y:=g*sqr(x)+h;

8: y:=g*sqr(x)+h*x+1;

10: y:=g*sqr(x+h+1)

end

Misol. Hafta kunlarining tartib nomeri, ularning o'zbekcha, ruscha va inglizcha nomlarini hosil qilish dasturini tuzing.

program hafta;

var i,n: integer;

begin n:=7;

for i:=1 to n **do**

case i of

1: writeln (I, '- kun-Dushanba- Ponedelnik – Monday');

2: writeln (I, '- kun Seshanba – Vtornik – Tuesday);

3: writeln (I, '- kun Chorshanba –Sreda – Wenesday');

4: writeln (I, '- kun Payshanba –Chetverg – Thuisday);

5: writeln (I, '- kun Juma- Pyatnisa – Friday');

6: writeln (I, '- kun Shanba – Subbota – Saturday');

7: writeln (I, '- kun Yakshanba – Voskresenye – Sunday');

end.

Sinov savollari

1. O'tish va bo'sh operatorlari nima uchun ishlatiladi?
2. Shartli operatorning umumiy ko'rinishi qanday?
3. Ichma-ich joylashgan shartli operatorlar qanday yoziladi?
4. Tanlash operatorining vazifasi nimadan iborat va uning shartli operatoridan farqi?

Uyga vazifa : Turbo-Paskal tizimida tarmoqlanuvchi dastur tuzish qoidalarini mukammal o'rganish va ularga doir misollar keltirish.

15-ma'ruza

Mavzu: Paskal tilida takrorlanuvchi jarayonlarni dasturlash

Reja:

1. Takrorlanuvchi jarayonlarni shartli operator orqali dasturlash.
2. Sikl operatorlari. Parametrli sikl operatori.
3. Sharti oldin va keyin qo'yiladigan sikl operatori.
4. Ichma- ich joylashgan sikl operatorlari.

Darsning maqsadi:

1. Talabalarda sikl operatorlari va ularning turlari to'g'risida umumiy va asosiy bilimlarni hosil qilish.
2. Turbo-Paskal tizimida parametrli, sharti oldin va keyin qo'yiladigan sikl operatori va ularning umumiy tuzilishi to'g'risida ko'nikmaga ega bo'lish.
3. Turbo-Paskal tizimida takrorlanuvchi jarayonlarni dasturlashga doir mustaqil ishlash malakalarini shakllantirish.

Tayanch iboralar: takrorlanuvchi jarayon, sikllar, sikl operatorlari, parametrli, sharti oldin va keyin qo'yilgan sikl operatorlari.

Dars o'tish usuli: namoyish dasturiy vositalat ishtirokida, jonli muloqotli ma'ruza .

Darsning xrono xaritasi – 80 minut.

Tashkiliy qism: xonaning tozaligi, jihozlanikishi, sanitariya holati. Talabalarining davomati– 2 minut.

Talabalar bilimni baholash: o'tilgan mavzuni qisqacha takrorlash, talaba bilan savol javob o'tkazish - 10 minut.

Yangi mavzu bayoni - 55 minut.

Mavzuni o'zlashtirish darajasini aniqlash va mustahkamlash – 10 minut.

Sinov savollari – 5 minut.

Uyga vazifa berish – 3 minut.

Ma'ruza bayoni

Takrorlanuvchi hisoblash jarayonlarini dasturlash.

Shunday hisoblash jarayonlari mavjud bo'ladiki, bunda bu jarayonning ayrim bo'laklarini bir necha marta takroran hisoblashga to'g'ri keladi. Bunday jarayonlar **takrorlanuvchi yoki siklli** jarayonlar deyiladi. Siklli jarayonlarni dasturlashda sikl operatorlaridan foydalaniladi, ya'ni dasturda bitta yoki bir necha parametrlarning bir vaqtda o'zgarishida bir gurux operatorlarni hisoblash

ko'p marotaba takrorlanadi. Masalaning qo'yilishiga qarab sikllar **takrorlanish soni ma'lum bo'lgan** va **takrorlanish soni noma'lum** bo'lgan sikllarga bo'linadi.

Sikllik jarayonlarni shartli operatorlar orqali dasturlash. Bunday holda sikl parametrining boshlang'ich qiymati berilgan bo'lishi kerak. Hisoblash shu parametr bilan bajariladi va qadam qiymati bilan o'zgartiriladi, hamda siklning tamom bo'lish sharti tekshiriladi. Agar sikl tugagan bo'lmasa, hisoblash davom ettiriladi.

Misol. $S = \sum_{i=1}^n (\frac{i}{2} + 1)$ ni hisoblash dasturini tuzing.

```
program SUMMA;  
label 1;  
var S, i : integer;  
begin  
  S:=0; i:=0;  
  1: i:=i+1;  
  S:=S+(i /2 + 1);  
  if i < 20 then goto 1 ;  
  writeln('Yig'indi S=',S)  
end.
```

Bundan tashqari, Paskal tilida sikllarni tashkil qilish bir nechta sikl operatorlaridan foydalaniladi. Bu sikl operatorlarining 3 turi mavjud: **parametrli, sharti oldin berilgan va sharti keyin berilgan sikl operatorlar**. Ularni tartib bilan ko'rib chiqamiz.

Parametrli sikl operatori. Agar sikllik jarayonlarda takrorlanish soni oldindan ma'lum bo'lsa, ko'p hollarda parametrli sikl operatoridan foydalaniladi.

Parametrli sikl operatorining umumiy ko'rinishi quyidagicha: **for V:=Xo to Xn do S;**

Bu yerda **for (uchun), to (gacha o'sib boradi)** va **do (bajar)**- xizmatchi so'zlar; **V- sikl parametri** deb ataluvchi o'zgaruvchi; **Xo,Xn** - sikl parametrining mos ravishda **boshlang'ich va oxirgi qiymati**; **S - sikl tanasi** deb ataluvchi operatorlar ketma- ketligi.

Masalan: for n:=1 to 5 do x:=2*x ;

Bu yerda sikl parametri **n** besh marta takrorlanadi va o'zgaruvchi **x** ketma - ket **2*x , 4*x , 8*x , 16*x , 32*x** qiymatlarni qabul qiladi, **n=6** bo'lganda hisoblash to'xtatiladi.

Parametrli sikl operatorida sikl parametri **V boshlang'ich qiymat Xo dan oxirgi qiymat Xn gacha** bo'lgan qiymatlarni **1 qadam** bilan o'zgarishda qabul qiladi. Uning xar bir qiymatida **S operatorlar** ketma-ketligi bajariladi.

Masalan. 1) for i:=1 to 10 do x:=sqr(i+1); S:=S+1;

2) for n:=1 to a+b do S:=S+sqrt(sqr(n)+1);

Paskal tilida parametrli sikl operatorini kamayib boruvchi shaklida ham ifodalash mumkin:

for V := Xn downto Xo do S;

Bu yerda **V - sikl parametri, Xn, Xo-** sikl parametrining **oxirgi va boshlang'ich qiymatlari; S - sikl tanasi; for (uchun), downto** (gacha kamayadigan) va **do(bajar)**

xizmatchi so'zlar. Bu operator quyidagicha bajariladi: sikl operatori **V** eng avval oxirgi **Xn** qiymatni qabul qilib oladi. Qolgan qiymatlari **-1** qadam bilan o'zgarib borgan **Xo** gacha bo'lgan qiymat-lardan iborat bo'ladi. Sikl parametri **V** ning har bir qiymatida **S** operatorlar ketma- ketligi bajariladi.

Masalan. 1) *for n := 5 downto 1 do x;*

Bu yerda sikl parametri **n** besh marta takrorlanadi va **x** o'zgaruvchi ketma- ket $32*x, 16*x, 8*x, 4*x, 2*x$ qiymatlarni qabul qiladi.

2) *y := 0; for i := n downto 1 do y := y + 2/i;*

Misol. $S = \sum_{i=1}^n (2i+1)yig'indini$ hisoblash dasturini tuzing.

1) 1- ko'rinish .

```
program summa;
var S : real;
    N, i : integer;
begin readln(n); S:=0;
      for i:=1 to n do S:=S+(2*i+ 1);
      write('S=';S)
end.
```

2) 2- ko'rinish

```
program summa;
var S: real;    N, i : integer;
begin read(n); S:=0;
      for i := n downto 1 do S:=S+ (2*i+ 1);
      write('S=',S)
end.
```

Sharti oldin qo'yilgan sikl operatori. Bu operator siklik jarayonda takrorlanishlar soni qo'yilgan shartga bog'liq bo'lgan holda qo'lla-niladi.

Sharti oldin qo'yilgan sikl operatorining umumiy ko'rinishi quyidagicha: **while B do S;**

Bu yerda **while(hozircha), do(qilmoq, bajarmoq)**- xizmatchi so'zlar, **B**- mantiqiy ifoda, **S**- operator. Bu yerda **S** operator nol marta yoki bir martadan ortiq bajarilishi mumkin. Chunki **S** operatorining har bir bajarilishidan oldin **B** ifodaning qiymati hisoblanadi. Agar **B** ifoda **chin(true)** qiymat qabul qilsa **S** operator bajariladi, aks holda ya'ni **B** ifoda **yolg'on(false)** qiymat qabul qilsa sikl operatorining bajarilishi tugaydi.

Agar **B** ifoda birinchi hisoblanishidayoq *yolg'on qiymatni* qabul qil-sa, **S operator** bir marta ham bajarilmaydi. Ko'rinib turibdiki, bu yerda siklik jarayonning tugallanish sharti **S operator** bajarilishidan oldin tekshiriladi. Shuning uchun bunday ko'rinishdagi sikl operatori - *sharti oldin qo'yilgan sikl operatori* deb yuritiladi.

Masalan.

1) *y := 1; k:=0;*
 *while y<=M do begin y:=y*3;k:=k+1 end*

2) *y:=1; v:=1;*
 while abs(v) >= eps do v:=(x/y-y)/2; y:=y+v end

Misol. $S = \sum_{i=1}^n (2i+1)$ yig'indini hisoblash dasturini tuzing.

```
program summa;  
  var S:real; n,i:integer;  
  begin read(n); S:=0; i:=1;  
    while i<=n do  
      begin S:=S + (2*i+1); i:=i+1 end;  
      write('S=',S)  
    end.
```

Bu misolda qaysi operatoridan foydalanish - **for** yoki **while** danmi farq qilmaydi. Agar takrorlanish soni ma'lum bo'lsa, **for** operatori ishlatiladi, takrorlanish soni ma'lum bo'lmasa **while** operatori qo'llaniladi.

1- misol. $S = \sum_{i=1}^{\infty} \frac{1}{2i}$ yig'indini aniqlikda hisoblash dasturini tuzing.

```
program summa;  
  var S, eps : real; i : integer;  
  begin  
    readln(eps); S:=0; i:=1;  
    while 1/(2*i) > eps do  
      begin  
        S:=S + 1/(2*i); i:=i+1;  
      end;  
      write('S=',S)  
    end.
```

Sharti keyin qo'yilgan sikl operatori. Bu operator takrorlanuvchi hisoblash jarayonlarida takrorlanishlar soni noma'lum bo'lgan holatlarda ishlatiladi. Operatorning umumiy ko'rinishi quyidagicha:

repeat S1,S2,...,Sn until B;

Bu yerda **repeat** (takrorlamoq), **until(gacha)** – xizmatchi so'zlar, **S1,S2,S3,...,Sn** operatorlar ketma ketligi, **B** - mantiqiy ifoda.

Operatorning bajarilishida **repeat va until** operatorlari orasida joylashgan operatorlar ketma- ketligi bir marta va undan ortiq bajarilishi mumkin. Bu jarayon qo'yilgan mantiqiy ifoda **yolg'on qiymat** qabul qilgunga qadar davom etadi.

Repeat operatorining **while** operatoridan farqi shundaki, bunda eng avval operatorning qiymati hisoblanib, so'ngra qo'yilgan shart tekshiriladi. Bu esa bunda berilgan operatorning hych bo'lmaganda bir marta bajarilishiga imkon yarata-di.

Masalan.

```
y:=1;  
repeat v:=(2*x/y-3*y)/2; y:=y+2*v  
until abs(v)<eps;
```

Misol. $S = \sum_{i=1}^{\infty} \frac{1}{i+2}$ yig'indini berilgan aniqlikda hisoblash dasturini tuzing.

```
program summa;
```

```

var S,eps:real; i:integer;
begin read(eps); S:=0; i:=1;
repeat S:=S + 1/(i+1); i:=i+1
until 1/(i+1) <=eps;
write('S=',S)
end.

```

Sinov savollari:

1. Siklik jarayon nima? Uni dasturlashda qaysi operatorlardan foydalanish mumkin?
2. Parametrli sikl operatori va uning umumiy ko'rinishi?
3. Sharti oldin va sharti keyin qo'yilgan operatorlar va ularning bir- biridan farqi?
4. Tanlash operatori va undan foydalanish xususiyatlari ayting?
5. Siklik jarayonlarga misollar keltiring.

Uyga vazifa : Turbo-Paskal tizimida takrorlanuvchi jarayonlarni dasturlash qoidalarini mukammal o'rganish va masalalar yechishda ularni amaliyotga tatbiq etish.

16-ma'ruza

Mavzu: Murakkab hisoblash jarayonlarini dasturlash.

Reja:

1. Iterasion sikllar.
2. Ichma-ich joylashgan sikllar.

Darsning maqsadi:

1. Iterasion sikllar to'g'risida umumiy va asosiy bilimlarni hosil qilish.
2. Murakkab jarayonlarni dasturlash bo'yicha ko'nikmalarni shakllantirish.
3. Turbo-Paskal tizimida ichma-ich joylashgan sikllarni dasturlash bo'yicha mustaqil ishlash.

Tayanch iboralar: murakkab jarayonlar, ichma-ich va iteratsion sikllar.

Dars o'tish vositalari: sinf doskasi, o'quv-uslubiy qo'llanmalar, ma'ruza matnlari, kompyuter, ma'ruza bo'yicha slaydlar, Turbo-Paskal tizimi .

Dars o'tish usuli: namoyish dasturiy vositalat ishtirokida, jonli muloqotli ma'ruza .

Darsning xrono xaritasi – 80 minut.

Tashkiliy qism: xonaning tozaligi, jihozlanikishi, sanitariya holati. Talabalarining davomati– 2 minut.

Talabalar bilimini baholash: o'tilgan mavzuni qisqacha takrorlash, talabalar bilan savol javob o'tkazish - 10 minut.

Yangi mavzu bayoni - 55 minut.

Mavzuni o'zlashtirish darajasini aniqlash va mustahkamlash – 10 minut.

Sinov savollari – 5 minut.

Uyga vazifa berish – 3 minut.

Ma'ruza bayoni

Iterasion sikllar. Takrorlanishlar soni noma'lum bo'lgan sikllar-da sikl tanasining takrorlanish jarayonida $y_1, y_2, \dots, y_n$ qiymatlar ketma- ketliklarini hosil qiladigan sikllar asosiy o'rinlardan birini egal-laydi. Bunday ketma-ketliklar a

limitga yaqinlashuvchi bo'ladi : $\lim_{n \rightarrow \infty} y_n = a$.

Bu ketma-ketligda y_n ning har bir yangi qiymati oldingi y_{n-1} ni hisobga olgan holda aniqlanadi va unga nisbatan izlanayotgan natija **a** ga yaqinroq aniq qiymat sifatida qaraladi. Yaqinlashishlarni bunday ketma-ketliklarini hosil qiluvchi **sikllar iterasion sikllar** deb yuritiladi.

Iterasion sikllarda **siklning davom etishi (tugash)** sharti **n** ning o'sishida y_n qiymatlarini **a** limitga cheksiz yaqinlashuvchi xossasiga asoslangan

Iterasional sikl tugaydi (natija y ga nisbatan taxminan olinadi, $y_n \approx a$ agar ba'zi bir n qiymatlar uchun $|y_n - y_{n-1}| < \varepsilon$ shart bajarilsa. Bu yerda ε juda kichik son bo'lib, hisoblash natijasining xatoliklaridir.

Iterasion sikl jarayonining tipik misoli sifatida cheksiz qatorlar yig'indisini hisoblash masalasini keltirish mumkin. $t_0, t_1, \dots, t_n, \dots$ qiymatlarning cheksiz qatori **yaqinlashuvchi** deyiladi, agar $s_n = t_0 + t_1 + \dots + t_n$ yig'indining **(n+1)** hadi **n** ning chekli o'sishida **S** ga yaqinlashuvchi bo'lsa.

S qator yig'indisi deb yuritiladi: $\lim S_n = S$; $\lim_{n \rightarrow \infty} \sum_{i=0}^n t_i = S$

Bunda yaqinlashuvchi qatorning umumiy hadi t_n nolga intiladi ya'ni

$$\lim_{n \rightarrow \infty} t_n = o \lim_{n \rightarrow \infty} (S_n - S_{n-1}) = 0,$$

Bundan ko'rinib turibdiki, $S_1, S_2, \dots, S_n, \dots$ ketma-ketlik izlanayotgan qiymatlar ketma-ketligi bo'lib, u yig'indini tugash shartini quyidagicha aniqlaydi :

$$|S_n - S_{n-1}| \leq \varepsilon \quad \text{ëku} \quad |t_n| \leq \varepsilon$$

Misol. Kosinusni qatorga yoyishdan foydalanib $S = \cos x$ fuksiya qiymatini $\varepsilon = 10^{-4}$ xatolik bilan hisoblang:

$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \dots = \sum_{n=0}^{\infty} t_n(x)$$

bu yerda $t_n(x) = (-1)^n \frac{x^{2n}}{(2n)!}$

Bu yig'indini hisoblash uchun, birinchidan, t_n hadning keyingi qiymatini aniqlash, ikkinchidan, iterasion formula $S_n = S_{n-1} + t_n$ bo'yicha yig'indini hisoblash.

Ushbu misolda faktorialni mavjudligini hisobga olganda, t_n ni aniqlash uchun to'g'ridan - to'g'ri umumiy formula bo'yicha hisoblash emas, balki rekurrent munosabat $t_n = t_{n-1} \varphi_n$ dan foydalaniladi.

t_n ni aniqlovchi formulaga **n** ning o'rniga **(n-1)**ni qo'yamiz:

$$t_{n-1} = (-1)^{n-1} \frac{x^{2(n-1)}}{(2(n-1))!} = (-1)^{n-1} \frac{x^{2n-2}}{(2n-2)!}$$

φ_n ni aniqlaymiz:

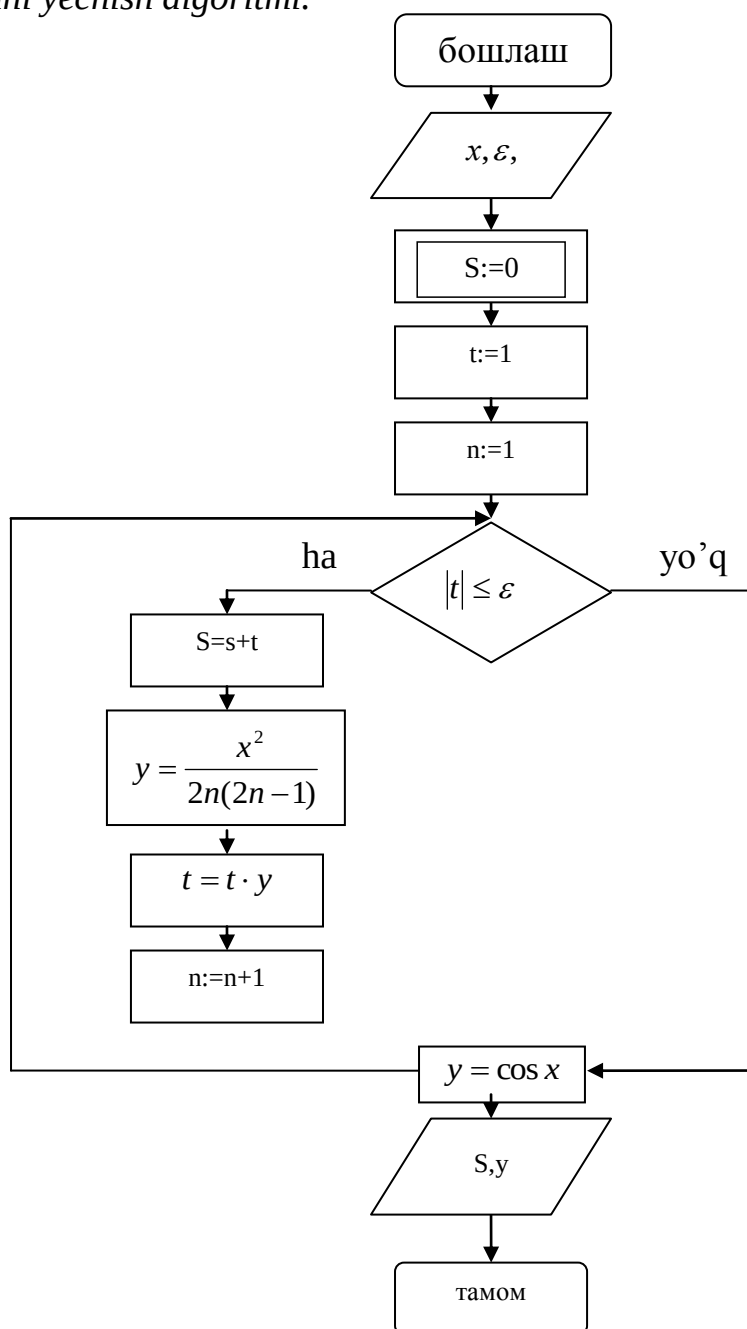
$$\varphi_n = \frac{t_n}{t_{n-1}} = \frac{(-1)^n \frac{x^{2n}}{(2n)!}}{(-1)^{n-1} \frac{x^{2n-2}}{(2n-2)!}} = \frac{(-1) x^{2n-2} (2n-2)!}{(-1)^{n-1} x^{2n-2} (2n)!} = \frac{(-1) x^2 (2n-2)!}{(2n)!}$$

Bizga ma'lumki $m! = 1 \cdot 2 \cdot 3 \cdot \dots \cdot m$. Shuning uchun

$$U_n = \frac{(-1)x^2 \cdot 1 \cdot 2 \dots (2n-2)}{1 \cdot 2 \dots (2n-2)(2n-1)2n} = -\frac{x^2}{(2n+1) \cdot 2n}$$

Formulaga t_n umumiy had uchun $n+0$ ni qo'shish orqali boshlang'ich qiymat $t_0 = 1$ topiladi.

Masalani yechish algoritmi.



Paskal tilida dastur.

```

program kosinus;
  var n: integer; eps, s, t, f, x, y: real;
begin read (x, eps);
    s:=0; t:=1; n:=1;
    while abs (t)>eps do
      begin s:=s+t; f:=-sqr(x)/(2*n*(2+n-1));
        t:=t*f; n:=n+1; end;
  end;
  
```

$y := \cos(x); \text{ wirteln ('s=', 'y=', y)}$
end.

Iterasion sikllarning asosiy xarakterli xususiyatlardan biri ularni algebraik va chiziqlimas tenglamalarni yechishda qo'llanilishidir.

$f(x) = 0$ tenglamani yechish ikki bosqichda amalga oshiriladi.

Birinchi bosqichda, **ildizni ajratish bosqichida** $[\alpha, \beta]$ oraliq aniq-lanadiki, bunda tenglamaning bitta ildizi mavjud bo'ladi. Tenglama ildizini ajratish uchun $f(x)$ funksiya grafigini yasash va shu grafikda $[\alpha, \beta]$ oraliqni aniqlash kerak bo'ladi.

Ikkinchi bosqichda **ildizni aniqlash bosqichida** ma'lum bir iterasion algoritm yordamida berilgan aniqlik darajasi bilan ildizni izlash amalga oshiriladi.

Ildizni aniqlashning eng oddiy metodlaridan biri **iterasiya metodi** hisoblanadi. U quyidagidan iborat.

$f(x) = 0$ berilgan tenglama bu yerda $f(x)$ funksiya $[\alpha, \beta]$ oraliqda uzluksiz.

Tenglamaning unga ekvivalent bo'lgan $x = y(x)$ ko'rinish bilan almashtiramiz. Ildizning boshlang'ich yaqinlashishi $x_0 \in [\alpha, \beta]$ ekanligini bilgan holda uning keyingi har bir qiymati $x_n = \varphi(x_{n-1})$ formula bilan hisoblanadi.

hisoblashlar quyidagi shart bajarilguncha davom ettiriladi:

$$|f(x_n)| \leq \varepsilon, \text{ bu yerda } \varepsilon \text{ -berilgan hisoblash xatosi.}$$

Misol. $x = \operatorname{tg} x = 0$ tenglama ildizini boshlang'ich $x_0 = 4,5$ qiymatda $\varepsilon = 10^{-3}$ aniqlik bilan aniqlang.

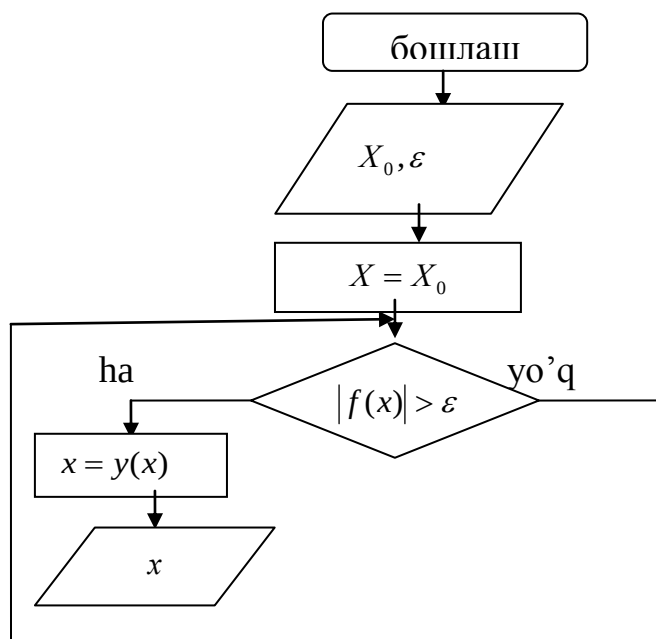
Berilgan tenglamani quyidagi ko'rinishga keltiramiz.

$$x = \operatorname{tg} x, \quad \text{u holda} \quad f(x) = x - \operatorname{tg} x; \quad \varphi(x) = \operatorname{tg} x.$$

X_0 ni aniqlash uchun quyidagicha grafik yasashdan foydalanish qulay:

$y_1 = x; \quad y_2 = y(x) = \operatorname{tg} x$ funksiyalarni grafigini yasash kerak. Bu grafik-larning kesishishi nuqtalari $f(x) = 0$ tenglamaning ildizi bo'ladi. Grafikdan yaqinlashishlar oralig'i $[\alpha, \beta]$ ajratib olinadi. x_0 sifatida shu oraliqdan ixtiyoriy nuqtani olish mumkin.

Masalani yechish algoritmi:



Paskal tilida dastur:

```

program TI;
  var
    eps, x, x0: real;
  begin
    read (eps, x0);
    writeln ('berilgan qiymatlar');
    writeln ('eps=',eps,'x0=',x0);
    x:=0;
    while abs (x – sin(x) / cos(x) ) > eps do
      begin
        x := sin(x) / cos(x);
        writeln ('x =', x);
      end
    end.

```

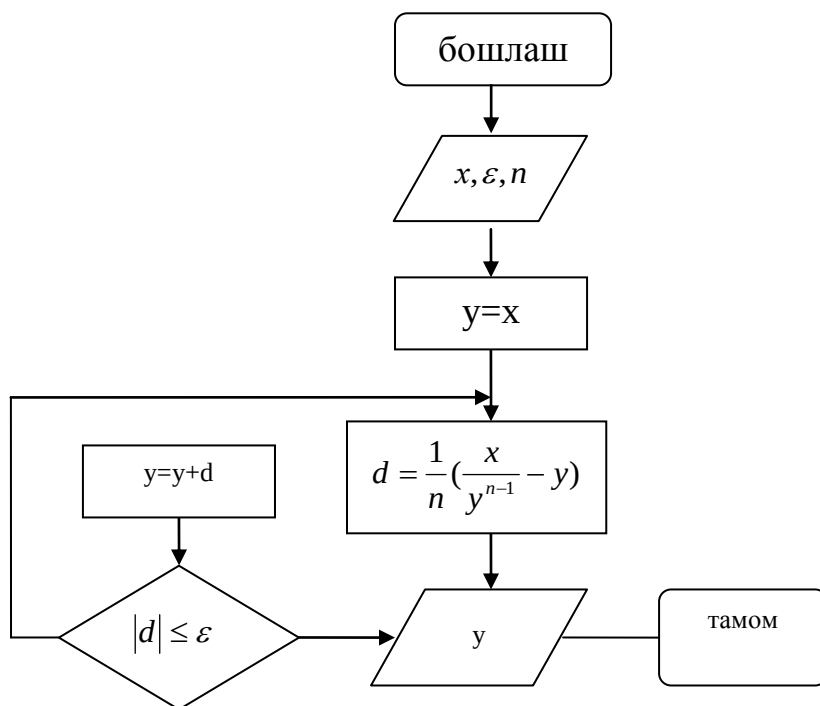
Izoh. $x_n = y(x_{n-1})$ iterasion jarayonni yaqinlashishini amaliy tekshirish uchun X ning barcha oraliq qiymatlari chop etiladi va ularning eng oxirgi- ci tenglamaning berilgan aniqlikdagi ildizi bo'ladi.

Misol. 3. $y = \sqrt[n]{x}$ n –darajali ildizni berilgan ε aniqlikda quyi-dagi rekurent formula bo'yicha hisoblash algoritmi va dasturini tuzing:

$$y_0 = x; \quad y_k = y_{k-1} + \frac{1}{n} \left(\frac{x}{y_{k-1}^{n-1}} - y_{k-1} \right);$$

$$k = 1, 2, 3, \dots \quad |y_k - y_{k-1}| \leq \varepsilon$$

Masalani yechish algoritmi:



Paskal tilida dasturi:

```

program   ITI;
var
     $n : integer; E, d, x, y : real;$ 
begin
    read (N, E);
     $y := x;$ 
    repeat
         $D := (x | \exp((n-1) * \ln(y)) - y) | n;$ 
         $y := y + D;$ 
    until  $\text{abs}(D) \leq E;$ 
    writeln (X, 'darajasi', N, 'ildiz')
    writeln ('teng', y, 'ga');
end.

```

Ichma-ich joylashgan sikllar. Hisoblash jarayonlarida bir- biri ichiga joylashgan sikllardan foydalaniladi. Agar sikl tanasi ham yana sikldan iborat bo'lsa, u holda bunday sikllar **ichma- ich joylashgan yoki murakkab sikllar** deb ataladi. Ichida boshqa sikl joylashgan sikl **tashqi sikl** deyiladi. Boshqa sikl ichida joylashgan sikl esa **ichki sikl** deyiladi.

Paskal tilida ichki va tashqi sikllar sifatida: *parametrli sikl, sharti oldin qo'yilgan sikl va sharti keyin qo'yilgan sikl operatorlaridan* ixtiyoriy bittasi qo'llaninishi mumkin. Tashqi va ichki sikllarni tashkil etish qoidasi oddiy sikllarnikiga o'xshab ketadi. Faqat ichma- ich sikllarni tuzishda qo'yidagi qo'shimcha shartga rioya qilish kerak: **ichki siklning hamma operatorlari to'liq tashqi sikl tanasi ichida joylashgan bo'lishi kerak.**

Murakkab sikllar shartli ravishda ichma- ichlik darajasiga bo'linadi, ya'ni eng tashqi sikl 1- *darajali*, keyingi sikl 2- *darajali* va hokozalar kabi bo'ladi. Har xil darajali sikllarning parametrlari bir vaqtda o'zgarmaydi.

Murakkab siklda avval hamma sikl parametrlari o'zlarining boshlang'ich qiymatiga ega bo'ladi. So'ngra eng ichki sikl to'liq bajariladi. Shundan keyin bunga nisbatan tashqi sikl o'z qiymatini bittaga o'zgartiradi, so'ngra ichki sikl esa yana to'liq bajariladi va hokazo bu jarayon hamma darajadagi sikllar talab etilgan barcha qiymatlarini qabul qilib bo'lguncha davom etadi. Ichma- ich joylashgan sikllar tanasining takrorlanishlarining umumiy soni, barcha sikl parametrlarining qabul qiladigan oxirgi qiymatlarining ko'paytmasiga teng: $N = N_1 * N_2 * \dots * N_k$.

1- misol. $S = \sum_{k=1}^{15} \sum_{n=1}^{10} (2a^k + x^n)$ ifodaning qiymatini hisoblash dasturini tuzing.


```

program summa;
var
    S,a, b, x,:real;
    k,n:integer;
begin
    S:=0; S1:=0; read(a, b, x);
    for k:=1 to 15 do {tashqi sikl boshi}
    begin
        for n:=1 to 10 do {ichki sikl boshi}
        begin
            S:=S + 2* (exp(k*ln(A)) + exp(n*ln(x)));
        end; {ichki sikl oxiri}
            S1:=S1+S;
        end; {tashqi sikl oxiri} S1:=(b+a)*S1;
        write('S1=',S1)
    end.

```

2- misol. $S = \sum_{k=1}^{15} \sum_{n=1}^{10} (i! + j!)$ ifodani hisoblash dasturini tuzing.

```

program summa;
    var i,j : integer; xo,h,xn,x,So,P,P1:real;
begin P:=1;
    begin S:=0; p1:=1;
        for i:=1 to 5 do
            for j:=1 to 10 do
                begin p:=p*i; p1:=p1*j; end;
                S:=S +(p + p1); end;
            writeln('S=',s);
        end.

```

Sinov savollari

1. Murakkab sikllar nima va ulardan qanday foydalaniladi?
2. Iteratsion sikllar nima va ulardan qanday foydalaniladi?
3. Iteratsion sikllarni dasturlash jarayonini tushuntiring.
4. Ichma- ich joylashgan sikllarni tushuntiring?
5. Ichma- ich sikllarni dasturlash jarayonini gapirib bering.

Uyga vazifa : Turbo-Paskal tizimida ichma-ich va iteratsion sikllarni dasturlash qoidalarini mukammal o'rganish va ularni amaloyotga tatbiq etish.

17-ma'ruza

Mavzu: Ma'lumotlarning o'zgaruvchi turlari.

Reja:

1. Sanab o'tiladigan turlar.
2. Chegaralangan turlar.

Darsning maqsadi:

1.Talabalarda o'zgaruvchi turlar to'g'risida umumiy va asosiy bilimlarni hosil qilish.

2. Turbo-Paskal tizimida sanab o'tiladigan turlar tuzilishi to'g'risida mukammal tasavvurga ega bo'lish.

3. Turbo-Paskal tizimida chegaralangan turlar bilan mustaqil ishlash malakalarini shakllantirish.

Tayanch iboralar: *dastur, oddiy va murakkab operatorlar, o'zgaruvchi turlar, sanab o'tiladigan va chegaralangan turlar.*

Dars o'tish vositalari: *sinf doskasi, o'quv-uslubiy qo'llanmalar, ma'ruza matnlari, komputer, ma'ruza bo'yicha slaydlar, Turbo-Paskal tizimi.*

Dars o'tish usuli: *namoyish dasturiy vositalat ishtirokida, jonli muloqotli ma'ruza.*

Darsning xrono xaritasi – 80 minut.

Tashkiliy qism: *xonaning tozaligi, jihozlanikishi, sanitariya holati. Talabalarning davomati*– 2 minut.

Talabalar bilimni baholash: *o'tilgan mavzuni qisqacha takrorlash, talaba bilan savol javob o'tkazish* - 10 minut.

Yangi mavzu bayoni - 55 minut.

Mavzuni o'zlashtirish darajasini aniqlash va mustahkamlash – 10 minut.

Sinov savollari – 5 minut.

Uyga vazifa berish – 3 minut.

Ma'ruza bayoni

Sanab o'tiladigan turlar.

Bu turni tashkil qilgan *skalyar o'zgarmaslar* tartiblangan chekli ketma-ketlikni hosil qiladi. Har bir o'zgarmasning qiymati o'z nomi bilan bir biridan vergul bilan ajratiladi. Berilgan skalyar turni tashkil qilgan komponentalar to'plami qavs ichiga olinadi.

Sanab o'tiladigan turni tashkil qilgan qiymatlar to'plami dastur tuzuvchi tomonidan biror bir belgilari bilan mos keladigan bitta yagona guruhga birlashtiriladi, ya'ni foydalanuvchi o'ziga kerakli turni aniqlaydi va dasturga kiritadi.

Sanab o'tiladigan turlar **type** xizmatchi so'zi bilan boshlanadigan turlarni aniqlash bo'limida yoki to'g'ridan to'g'ri o'zgaruvchilar bo'limida tavsiflanishi mumkin.

Masalan:

a) type

yilfasli=(bahor, yoz, kuz, qish);

hafta=(dush, sesh, chor, pay, juma, shan, yaksh);

meva=(olma, nok, olcha, o'rik, olxo'ri);

var

fasl: jilfasli;

kun: hafta;

dd: meva;

Bu degan so'z, **fasl** o'zgaruvchisi **jilfasli** ga tegishli **bahor, yoz, kuz, qish** qiymatlarini, **kun** esa **hafta** ga tegishli qiymatlarni, **dd** esa **meva** ga tegishli qiymatlarni qabul qilishi mumkin.

b) var fasl: (bahor, yoz, kuz, qish);

kun: (dush, sesh, chor, pay, juma, shan, yaksh);

dd: (olma, nok, olcha, o'rik, olxo'ri);

Bu holda turning nomi qatnashmaydi, o'zgaruvchilar esa berilgan turni tashkil etuvchi qiymatlar to'plami bilan aniqlanadi.

Sanab o'tiladigan turni tashkil qilgan qiymatlarning tartiblangan ketma-ketligi noldan boshlab avtomatik ravishda nomerlanadi va ular orasida esa munosabat o'rnatiladi:

Sanab o'tiladigan turlar ustida munosabat amallarini va *succ, pred, ord* funksiyalar qiymatini hisoblash mumkin: *olma<nok (true) bahor> qish (false); succ (olcha)=o'rik; pred (olcha)=nok;*

Sanab o'tiladigan turdagi o'zgaruvchilar parametrli sikl operatorida qo'llanilishi mumkin, ya'ni sikl operatorini quyidagicha yozish mumkin:

dd:olma;

for i:=dd to olxo'ri do <operator>.

Bu yerda sikl operatori **i** ham **meva** turiga tegishli bo'lishi mumkin.

Chegaralangan turlar.

Standart skalyar, yoki ixtiyoriy skalyar (**real** dan tashqari) turning oralig'i *chegarangan (cheklangan) tur* kabi aniqlanishi mumkin. Oraliq o'zgarmasning kichik va katta qiymatlari bir-biridan nuqtalar ajratilgan holda beriladi.

Chegarangan tur quyidagi ko'rinishda beriladi:

<Chegaralangan tur>=<1- o'zgarmas> .. <2- o'zgarmas>;

Bu yerda **1- o'zgarmas** hych qachon **2 – o'zgarmas**dan katta bo'lmasligi kerak.

Masalan:

1..20 (integer)

dush..yaksh (sanab o'tiladigan tur)

'A'..'Z' (char)

O'zgarmasning eng kichik va eng katta qiymatlari chegaralangan turni aniqlaydigan oraliqning *quyi va yuqori chegaralari* deyiladi.

Chegaralangan turni tavsiflash turlarni aniqlash bo'limida amalga oshirilishi mumkin.

Masalan:

a) type (turlar bo'limi)

m=5 .. 48;

jilfasli=bahor..qish;

hafta=dush..yaksh;

meva=olma..olxo'ri;

var

x,y,z:m;

fasl: jilfasli;

kun: hafta;

dd:meva;

a) var (o'zgaruvchilar to'plami)

x,y,z: 5. .48

fasli= bahor. .qish;

hafta= dush. .yaksh;
meva= olma. .olxo'ri;

Chegaralangan turga tegishli o'zgaruvchilar ustida shu turga mos bo'l-gan barcha amallarni bajarish mumkin.

z:=x+y;
p:=bahor>fasl ;
x:=ord (kun);
y:=succ(dd);

Misol. Hayvonlar ro'yxati berilgan. Shu ro'yxatdagi yo'lbars va quyon dan keyin turgan hayvonning tartib nomerlarni aniqlang.

program tartib ;

type hayvon= (tulki, bo'ri, yo'bars, quyon, ayiq, sher, kiyik);

var t1, t2 : hayvon; h1, h2: integer;

begin

t1:=yo'bars; t2:=succ (quyon);

h1:=ord(t1)+1; h2=odr(t2)+1;

writeln ('yo'barsning tartib nomeri=', h2:2);

writeln ('quyondan keyin turgan hayvonning tartib nomeri=', h2:2);

end. {dasturni bajarishda ro'yxatdagi hayvonlarni nomini lotin alfavitida ifodalash kerak}

Sinov savollari.

1. O'zgaruvchi turlarga qanday ma'lumotlar kiradi?
2. Qanday o'zgaruvchi turlarni bilasiz?
3. Chegaralangan tur nima va u qanday tavsiflanadi?
4. Sanab o'tiladigan tur nima va u qanday tavsiflanadi?
5. Chegaralangan va o'tiladigan turlarga misollar keltiring?
6. type fasl=(bahor, yoz, kuz, qish)

var x,y: fasl;

t:=(issiq, sovuq) tavsiflash berilgan bo'lsa:

x,y,t o'zgaruvchilar qanday qiymatlarni qabul qiladi?

Succ(y); pred(sovuq) nimaga teng?

Read(x); write (kuz); yozuv to'g'rimi?

For x:=bahor to kuz do ko'rinishda yozish mumkinmi?

7. type raqam ='0'..'9';

var a:raqam; k:0..9; n:integer; tavsiflash berilgan?

a). raqam qiymatining turi qanday?

b) k o'zgaruvchi qanday qiymatlarni qabul qiladi?

c) k:=5; k:=-10; k:=ord(0); k:=ord(a)-ord('0'); yozuvlar to'g'rimi?

Uyga vazifa : Turbo-Paskal tizimida chegaralangan va sanab o'tiladigan turlar bilan ishlash qoidalarini mukammal o'rganish va ularni amaloyotga tatbiq etish.

18- Ma'ruza

Mavzu: Paskal tilida massivlar. Bir va ikki o'lchovli massivlar.

Reja:

1. Paskal tilida murakkab turlar.
2. Muntazam turlar. Massiv tushunchasi.
3. Bir o'lchovli massivlar.
4. Ikki o'lchovli massivlar.

Darsning maqsadi:

1. Talabalarda massivlar to'g'risida umumiy va asosiy bilimlarni hosil qilish.
2. Turbo-Paskal tizimida massivlar, ularning berilishi va ulardan foydalanish bo'yicha ko'nikmalarga ega bo'lish.
3. Turbo-Paskal tizimida massiv elementlari bilan mustaqil ishlash malakalarini shakllantirish.

Tayanch iboralar: *tur, murakkab tur, muntazam tur, massiv, massiv o'lchovi, massiv elementlari, bir o'lchovli va ikki o'lchovli massivlar, massivlarni tavsiflash, massiv elementlari bilan ishlash.*

Dars o'tish vositalari: *sinf doskasi, o'quv-uslubiy qo'llanmalar, ma'ruza matnlari, komputer, ma'ruza bo'yicha slaydlar, Turbo-Paskal tizimi.*

Dars o'tish usuli: *namoyish dasturiy vositalat ishtirokida, jonli muloqotli ma'ruza.*

Darsning xrono xaritasi – 80 minut.

Tashkiliy qism: *xonaning tozaligi, jihozlanikishi, sanitariya holati. Talabalarning davomati*– 2 minut.

Talabalar bilimni baholash: *o'tilgan mavzuni qisqacha takrorlash, talabalar bilan savol javob o'tkazish* - 10 minut.

Yangi mavzu bayoni - 55 minut.

Mavzuni o'zlashtirish darajasini aniqlash va mustahkamlash – 10 minut.

Sinov savollari – 5 minut.

Uyga vazifa berish – 3 minut.

Ma'ruza bayoni

Standart bo'lmagan murakkab turlar. Berilmalarning standart bo'lmagan turlaridan yana biri ***murakkab turlar*** hisoblanadi. Murakkab turlar to'rt xil ko'rinishda berilishi mumkin: ***regulyar (muntazam) turlar, ya'ni massivlar; kombinatsiyalangan turlar, ya'ni yozuvlar; faylli turlar, ya'ni fayllar; to'plamli turlar, ya'ni to'plamlar.***

Regulyar (muntazam) tur. Massivlar. Massiv deganda bir xil turga mansub bo'lgan obyektlarning biror xususiyatiga ko'ra mujassamlangan majmuasi yoki to'plami tushuniladi. Massivlarni tavsiflash uchun regulyar (muntazam) turlardan foydalaniladi, ya'ni ular bir xil turdagi ma'lumotlarning chekli to'plamini tavsiflash uchun ishlatiladi. Massivlarni tashkil etgan obyektlar uning elementlari bo'lib, ular tartib bilan nomerlanadi ya'ni o'z indeksiga ega bo'lib, ular ana shu indekslarning nomeri bo'yicha tartiblanadi.

Regulyar turlar, ya'ni bir o'lchovli massivlar quyidagicha ko'rinishda beriladi:

array [< indeks turi>] of < komponenta turi >;

Indekslar turi *real* va *integer* standart turlardan tashqari barcha oddiy turlar bo'lishi mumkin, chunki massivlar cheklangan elementlar sonidan iborat bo'ladi.

Ko'p xollarda indekslar turi sifatida chegaralangan turlardan foydalaniladi. Chunki chegaralangan turlarning qiymatlar to'plami tartiblangan va nomerlangan bo'ladi.

Masalan, 100 ta haqiqiy turdagi elementdan iborat massiv quyidagicha beriladi:

a) array[1..100] of real

Chegaralangan butun tur 1..100 elementlarini sonini va tartibini aniqlaydi.

b) array [1147..2229] of integer

v) array [-109..-1] of integer

Har bir beriladigan regulyar turga, ya'ni massivga turlarni tavsiflash orqali nom berish mumkin.

Masalan.

type

chegara = 1..100;

vektor = array [chegara] of real;

massif = [1147..2279] of integer;

massif2 = array [-109.. -1] of integer;

Regulyar tur o'zgaruvchilaridan foydalanish uchun ular o'zgaruvchilarni tavsiflash bo'limida aniqlanadi.

var

a,b : vektor;

c,d : massif;

Regulyar turlarni to'g'ridan - to'g'ri o'zgaruvchilarni tavsiflashda ham berish mumkin.

var

r,t : array [chegara] of real;

s,g : array [1147..2279] of integer;

q,h : array [-109..-1] of integer;

k,m : array [1..50] of (schar,kub,piramida);

Sanab o'tiladigan turning qiymatlar to'plami ham cheklangan nomerlangan to'plamni tashkil qiladi, ya'ni sanab o'tiladigan tur ham indeks turi sifatida qo'llanilishi mumkin. Chunki sanab o'tiladigan turning qiy-matlari tartiblangan va chekli sondan iborat bo'ladi:

type

oy = (yanvar, febral, mart, aprel, may, iyun, iyul, avgust, sentabr, oktabr,

noyabr, dekabr);

hafta = (dushanba, seshanba, chorshanba, payshanba, juma, shanba);

Regulyar tur o'zgaruvchilari quyidagicha aniqlanadi:

var

t,r: array [oy] of real;

a,b: array [hafta] of real;

Bundan tashqari indeks sifatida **boolean va char** turlari ham qo'llanilishi mumkin:

type

```

    belgi : array [boolean] of integer;
    alfavit : array [char] of integer;
var
    k: belgi; s: alfavit;

```

Masalan.

```

type
    x = array [1..k] of boolean;
    yil = array [yanvar .. dekabr] of integer;
    chiroq = array [1..3] of sfetofor;
var
    z,y : x; oj : yil; m : chiroq;

```

Massivlar bir xil turga tegishli bo'lsa va bir xil sondagi komponentalardan iborat bo'lsa, bunday massivlar uchun ta'minlash operatorini qo'llash mumkin: **X:=Y**; Bu operatorning bajarilishi natijasida **X** massivning elementlari **Y** massivga ham ko'chiriladi.

Indeks turining indeksi o'zgarmas yoki ifoda bo'lishi mumkin. Massivlar matematikada, iqtisodda va informatikada berilmalarning tartiblangan to'plamini tavsiflash uchun keng va samarali qo'llaniladi: **vektorlar, matrisalar, jadvallar va hokozalar.**

Misol. Bir o'lchovli **20 elementdan** iborat bo'lgan **A** massiv elementlarining yigindisini hisoblash dasturini tuzing:

$$S = \sum_{i=1}^{20} A_i$$

```

program summas;
type
    n = 1.. 20;
var
    S: real; A: array [N] of real;
    i : n;
begin
    for i:=1 to 20 do
        read(A[i]);
    readln;
    S:=0;
    for i:=1 to 20 do S:= S + A[i];
    writeln('Massiv yig'indisi =', S);
end.

```

Ko'p o'lchovli massivlar. Bir o'lchovli massivlarda massiv komponentalari qiymatlardan, ya'ni alohida berilmalardan iborat bo'ladi. Paskal tilida bundan tashqari massiv komponentalari massivlardan iborat bo'lishi mumkin. Agar massiv - komponentalarni qiymatlaridan tashkil topgan bo'lsa, u holda ikki o'lchovli massivlar hosil bo'ladi. **Masalan, matrisalar.** Xuddi shunday, massiv komponentalari matrisalardan iborat bo'lsa, uch o'lchovli massiv hosil bo'ladi.

Ikki o'lchovli massivni aniqlovchi tur ham, xuddi bir o'lchovli massiv kabi beriladi:

array [<indeks turi>] of <komponenta turi>;

Bu yerda komponenta tur emas, regulyar bo'lganligi uchun u quyidagicha ko'rinishda bo'ladi:

array [<indeks turi>] of <komponenta turi> of array [<indeksturi>] of < turi>

Masalan, 10 ta satr va 20 ta ustun iborat bo'lgan ikki o'lchovli haqiqiy massiv turini quyidagicha berish mumkin:

array [1..10] of array [1..20] of real;

Bu turga tegishli o'zgaruvchilardan foydalanish doimgidek, ikki xil usulda berilishi mumkin. Birinchi usulda talab qilingan regulyar tur- o'zgaruvchilarni tavsiflash bo'limida beriladi.

Masalan:

var

A: array [1..10] of array [1..20] of real;

B: array [dush .. yaksh] of array [boolean] of -20..20;

Ikkinchi usulda, turlarni tavsiflash bo'limida aniqlanadigan regulyar turga nom beriladi, o'zgaruvchilarni tavsiflashda esa faqatgina shu turning nomi ko'rsatiladi:

type

matrisa = array [1..10] of array [1..20] of real;

yangitur = array [dush..yaksh] of array [boolean] of -20..20;

var A,B: *matrisa*; C: *yangitur*;

Paskal tilida regulyar turlarni yozishni qisqartirilgan ko'rinishi ham qo'llaniladi:

array [<indeks turi>{,<indeks turi>}] of <komponenta turi>;

Masalan:

type

matrisa = array [1..10,1..20] of real;

A = [1..20, 1..20] of integer;

Bu yerda **matrisa [i,j]** - *matrisa elementlari*; **i**- *elementlarning satrlar bo'yicha turgan o'rnini aniqlaydi* va 1..10 oraliqdagi qiymatni qabul qiladi; **j**- *esa elementlarning ustunlar buyicha turgan o'rnini aniqlab*, 1..20 oraliqdagi qiymatlarni qabul qiladi.

Shu narsani eslatish lozimki, indekslar bitta va bir xil turga mansub bo'lishi shart emas.

Misol. Ikki o'lchovli **A(6,6)** massiv elementlari yig'indisini hisoblash dasturini tuzing.

program *summas*;

var A:array[1.. 6,1.. 6] of real; i,j : integer; S:real;

begin

for i :=1 to 6 do

for j :=1 to 6 do

read(A[i,j]); readln;

for i:=1 to 6 do


```

for j := 1 to 6 do
  S := S + A[i,j] ;
  writeln('Yig'indi S=',S);
end.

```

Sinov savollari:

1. Murakkab turlar ma'lumotlarning qanday ko'rinishi?
2. Muntazam turlar nima uchun ishlatiladi?
3. Massiv deganda nimani tushunasiz?
4. Qanday massivlardan foydalaniladi?
5. Bir o'lchovli massivlar qanday tasvirlanadi?
6. Ikki va ko'p o'lchovli massivlardan qanday foydalaniladi?

Uyga vazifa : Turbo-Paskal tizimida massiv elementlari bilan ishlash qoidalarini mukammal o'rganish va ularni amaloyotga tatbiq etish.

19-ma'ruza

Mavzu: Paskal tilida qism dastur. Procedure va Function dan foydalanib dasturlash

Reja:

1. Paskal tilida qism dastur.
2. Procedure ning umumiy tuzilishi va uni tavsiflash.
3. Function ning umumiy tuzilishi va uning tavsifi.
4. Qism dasturga murojaatni amalga oshirish.

Darsning maqsadi:

1. Talabarlarda qism dastur, uning turlari to'g'risida umumiy va asosiy bilimlarni hosil qilish.
2. Turbo-Paskal tizimida qism dasturdan foydalanib dasturlash bo'yicha mukammal ko'nikmalarga ega bo'lish.
3. Turbo-Paskal tizimida qism dasturdan mustaqil foydalanish malakalarini shakllantirish.

Tayanch iboralar: qism dastur, rrocedure, function, qism dastur tuzilishi, qism dasturni tavsiflash, qism dasturga murojaat.

Dars o'tish vositalari: sinf doskasi, o'quv-uslubiy qo'llanmalar, ma'ruza matnlari, komputer, ma'ruza bo'yicha slaydlar, Turbo-Paskal tizimi .

Dars o'tish usuli: namoyish dasturiy vositalat ishtirokida, jonli muloqotli ma'ruza .

Darsning xrono xaritasi – 80 minut.

Tashkiliy qism: xonaning tozaligi, jihozlanikishi, sanitariya holati. Talabarlarning davomati– 2 minut.

Talabalar bilimini baholash: o'tilgan mavzuni qisqacha takrorlash, talabala bilan savol javob o'tkazish - 10 minut.

Yangi mavzu bayoni - 55 minut.

Mavzuni o'zlashtirish darajasini aniqlash va mustahkamlash – 10 minut.

Sinov savollari – 5 minut.

Uyga vazifa berish – 3 minut.

Ma'ruza bayoni

Masalalar yechish jarayonida parametrning har xil qiymatlarida bir xil hisoblashlarni bir necha marta takroran bajarish zaruriyati tug'iladi. Dastur hajmini kichraytirish uchun bunday hisoblashlarni alohida qism dasturga ajratish maqsadga muvofiq bo'ladi. Asosiy dasturda esa zaruriyat paydo bo'lgan paytda qism dasturga murojaat qilinadi. Paskal tilida qism dastur sifatida **funksiya(Function) va procedura(Procedure)dan** foydalaniladi.

Funksional boqliqlikni aniqlash uchun mo'ljallangan procedura **funksiya-procedura** deb yuritiladi. Paskal tilida funktsiyani aniqlash uchun **<funksiya tavsiflash>** tushunchasi ishlatiladi. Shu kabi barcha tavsiflash quyidagicha ko'rinishga ega:

<funksiya tavsiflash>::= <funksiya sarlavhasi>; <blok>;

O'z navbatida funksiya sarlavhasi quyidagicha tavsiflanadi:

<funksiya sarlavhasi>::= function<funksiya nomi>(<formal parametrlar ruyxati>):<tur nomi>;

Bundan ko'rinish turibdiki, funksiya sarlavhasi **function** xizmatchi so'zidan boshlanadi. Foydalaniladigan funktsiyaga identifikator ko'ri-nishida nom beriladi. Funktsiya nomidan keyin esa qavs ichida tavsiflanayotgan funktsiyaning argumenti sifatida formal parametrlar ro'yxati beriladi. Funktsiya sarlavhasi aniqlanayotgan funksiya qiymati turining nomi bilan tugallanadi. Bu yerda faqat tur nomini ko'rsatish mumkin:

function AAA3(n:integer; x: real):real;

Misol. $Y=x^n$ darajani xisoblash dasturini funksiya ko'rinishida yozing.

function daraja(N:integer; x: real) :real;

var

i : integer; y : real;

begin

y:=1;

for i := 1 to n do

y := y*x;

daraja :=y

end.

Bu yerda funksiya sarlavhasida **daraja** nomi bilan kiritiladigan berilmalarni aniqlovchi **N, x** parametrlar sanab o'tilgan. Funktsiyaning bajarilish natijasi **daraja** ga ta'minlanadi.

Procedurani tavsiflash. Proceduralar o'zgaruvchilar bo'limidan keyin dasturning maxsus tavsiflash bo'limida tavsiflanadi. Har qanday procedura, dasturga o'xshab, **procedura sarlavhasi va blokidan** tuzilgan.

Procedura sarlavhasi quyidagicha beriladi:

procedure<nom>(<parametrlar ro'yxati>;

bu yerda **procedure** - xizmatchi so'z; **nom** - identifikatorlarni hosil qilish umumiy qoidalariga mos aniqlanadigan procedura nomi; **parametrlar ro'yxati** - procedura ishi natijalarini va kiritiladigan berilmalarni ularning turlarini belgilash uchun ishlatiladigan nomlar ro'yxati.

Ro'yxatda berilgan parametrlar **formal parametrlar** deyiladi :

procedure<nom>;

Proceduraning asosiy qismi **blok** bo'lib, o'zgaruvchilar bo'limida (**belgilar, o'zgarmlar, turlar, o'zgaruvchilar, proceduralar va funksiyalar**) va **begin-end** tarkibiy operatorlarni o'z ichiga olgan operatorlar bo'limidan tuzilgan. Procedura bloki nuqta vergul bilan tugaydi.

Misol. n natural ko'rsatkichli $y=x$ darajani hisoblash dasturini procedura ko'rinishida ifodalang.

Prasedure *daraja*($n:integer;x:real;var y:real$);

```
var
  i : integer;
begin
  y := 1;
  for i := 1 to N do y := y*x;
end.
```

Procedura sarlavhasida **daraja** nomi bilan proceduraning kiritiladigan berilmalarini aniqlovchi **N, X** parametrlar va proceduraning bajarilish natijasi - izlanayotgan daraja qiymati belgilangan **Y** parametr keltirilgan. Barcha formal parametrlar ko'rsatilgan.

Procedura tanasi (bloki) quyidagicha tuzilgan:

1) *tavsiflash qismi, bu yerda lokal o'zgaruvchi hisoblangan, faqat shu procedura ichida kerakli va mazmunga ega bo'lgan i o'zgaruvchi aniqlanadi. (lokal o'zgaruvchi qiymati asosiy dasturda foydalanilmaydi)*

2) *natural ko'rsatkichli haqiqiy son darajasini hisoblash algoritmini bajaruvchi begin-end tarkibiy operatorlardan.*

Misol 2. $y=x^n$ hisoblash algoritmini parametrsiz procedura ko'rinishda ifodalang.

```
program daraja1;
var
  i : integer;
begin
  y:=1;
  for i:=1 to n do y:=y*x;
end.
```

Bu holda **daraja1** procedurada *formal parametrlar* ruyxati berilmaydi va proceduraning blokida **tavsiflangan i** lokal o'zgaruvchi va shu proceduraning tavsifidan iborat bo'lgan dasturda tavsiflangan **x,n,y** o'zgaruvchilar bilan ishlaydi. **x, n, y** o'zgaruvchilar **daraja** proceduraga nisbatan **global o'zgaruvchilar** deyiladi. *Global o'zgaruvchilar* qiymatidan asosiy dasturning ixtiyoriy joyida foydalanish mumkin.

Proceduraga murojaat qilish. Tavsiflash bo'limida joylashgan procedura hyech qanday amalni bajarmaydi. Bu procedurani bajarish uchun dasturning kerakli joyida unga murojaat qilish kerak bo'ladi. Proceduraga murojaat qilish procedurani chaqirish - **maxsus operatori** yordamida yoki **procedura operatori** orqali amalga oshiriladi. U quyidagicha ko'rinishga ega:

<nom > (<argumentlar - ro'yxati >);

bu yerda **nom** - murojaat bo'ladigan proceduraning nomi; **argumentlar ruyxati** - proceduraning bajarilishida uning formal parametrlari o'rniga qo'yiladigan konkret qiymatlar va nomlar ketma-ketligi.

Procedurani chaqirishda uning sarlavhasida ko'rsatilgan formal parametrlar argumentlarga berilish tartibida almashtiriladi: ro'yxatdagi chapdan birinchi parametrga mos ravishda birinchi argument qo'yiladi, ikkinchisiga- ikkinchi va hokazo.

Procedura operatorida berilgan argumentlar **tayinli (faktik)** parametrlar ham deb yuritiladi. **Formal va tayinli (faktik)** parametrlarning soni va turi bir-biriga mos kelishi zarur.

Misol. $Z = a^m$ darajani hisoblash dasturini tuzing, bu yerda **m**- butun son va **a**= 0.

Butun ko'rsatkichli daraja qo'yidagicha aniqlanadi:

1, agar $m = 0$

$a = a$, agar $m > 0$

$1/a$, agar $m < 0$

```
program dastur;
var m:integer; a,z:real;
  procedure daraj a(n : integer; x : real; var y : real);
    var i:integer;
  begin
    y:=1;
    for i:=1 to n do
      y:=y*x
    end;
  begin
    read(a,m);
    if m=0 then z:=1 else
    if m > 0 then daraja(m,a,z) else daraja(-m,1/a,z);
    writeln(a:8:3,'darajasi',m:3,'teng',z)
  end.
```

Ushbu dasturda procedura operatori ikki marta ishlatilgan va har safar dasturning tavsiflash qismidan daraja procedurasi chaqiriladi. Procedura tanasida **x, n, y** formal parametrlarni birinchi marta **m, a, z** tayinli (faktik) parametrlarga va ikkinchi marta **-m, 1/a, z** parametrlarga almashtirish ro'y beradi. So'ngra procedura operatorlari uchun mo'ljallangan amallar ketma-ketligi bajariladi va **z** natija dasturga qaytariladi. **Daraja** dastur operatoriga qaytish, procedurani chaqirish operatoridan keyingi operator orqali amalga oshiriladi.

Formal (rasmiy) parametrlarni **tayinli (faktik)** parametrlar bilan almashish mexanizmini va maqsadini tushunish juda muhimdir. Formal parametrlar - bu tayinli parametrlarni qo'yish o'рни va turini aniqlaydigan va procedurada qatnashadigan **soxta o'zgaruvchilardir**. **Tayinli parametrlar** - bu procedurani chaqirishda uning tanasidan formal parametrlar bilan almashadigan dasturning real obyektlaridir. Ana shu obyektlar ustida procedura tanasida qaralayotgan operatorlar bilan amallar bajariladi.

Agar parametrsiz procedurani chaqirish amalga oshirilsa, u holda procedura operatori shu procedura nomidan iborat bo'ladi.

Misol. *Parametrsiz proceduradan foydalanib, butun ko'rsatkichli $y = x^n$ darajani hisoblash dasturini tuzing.*

```
program dastur2;  
  var m : nteger; a,z : real;  
  procedure daraja;  
    var i : integer;  
  begin  
    y:=1;  
    for i:=1 to n do  
      y:=y*x  
  end;  
begin  
  read(n,x);  
  writeln(x:8:3,'darajasi',n:3);  
  if m=0 then z:=1 else  
    if m >0 then daraja else  
    begin n := -n; x := 1/x; daraja end;  
  writeln('teng',z)  
end.
```

Bu misolda procedura operatori faqat uni chaqirishni amalga oshiradi. Boshlang'ich berilmalar shu proceduraga murojaat qilishdan iborat bo'lgan, dastur tanasida aniqlangan va tavsiflangan n , x o'zgaruvchilar yordamida, ya'ni global o'zgaruvchilar yordamida proceduraga uzatiladi. **Daraja** procedurasining bajarilish natijasi ham xuddi shunday asosiy dasturga uzatiladi.

Global va lokal o'zgaruvchilar tushunchasi Paskal tilida blokli struktura mavjudligidan hosil bo'lgan. Har kanday **dastur, procedura va funksiya** o'zining tavsiflanish sohasi blokidan tashkil topadi va bu blok ichida boshqa procedura va funksiyalarni tavsiflash, hamda ularga murojaat qilish ham qatnashishi mumkin. Dastur va unda tavsiflangan procedura va funksiyalar majmuasi blokli strukturani hosil qiladi. O'zining tavsiflash bo'limi ichida boshqa procedura va funksiyalarni tavsiflash mavjud bo'lgan blok unga nisbatan tashqi blok deyiladi. Ayrim bloklarning tavsiflash bo'limidan ishtirok etgan blok ichki yoki qism blok deyiladi.

Yuqorida aytilganlarni yaxshiroq tushunib olish uchun uchta blokdan iborat strukturani keltiramiz (3- rasm). **I,J,K** o'zgaruvchilar **3-blokka** nisbatan lokal hisoblanadi va **1,2-** bloklar uchun tegishli bo'lmaydi. **X, Y, Z** o'zgaruvchilar **2, 3** bloklar uchun taaluqlidir va **1- blokka** taaluqli emas, 3-blokka nisbatan ular global hisoblanadi. **A, V, S** o'zgaruvchilar dasturga to'liq taaluqlidir va **2, 3-** bloklarda **global** hisoblanadi.

Misol. **program** uzqs;
 var x, y, z, S, P : real;
 procedure uzburhak; {procedurani tavsiflanishi}
 const coef = 0.5;
 var P : real; R : boolean;

```

begin
   $R := (x+y>z) \text{ and } (x+z>y) \text{ and } (y+z>x)$ 
  if  $R$  then begin  $P := \text{coef}*(x+y+z);$ 
   $S := \text{sqr}(P*(P-x)*(P-y)*(P-z));$  end else  $S := -1.0;$ 
end; {proceduraning oxiri}
begin { dasturning asosiy qismi}
  readln( $x,y,z$ ); uzburhak; {procedurani chaqirish}
  if  $S \leq 0$  then writeln('Uchburchak yasash mumkin emas')
  else begin  $PL := \text{sqr}(S);$  writeln('PL=', $PL$ ) end;
end. { dastur oxiri}

```

Bu misolda o'zgaruvchilar **R** va **P** o'zgaruvchilar **uzburchak** procedurasi-ning bloki uchun lokal o'zgaruvchilar hisoblanadi. Ular faqat shu blokning ichidan foydalaniladi va **uzqs** dastur blokiga taaluqli emas. Dasturning **x, y, z, S** o'zgaruvchilari procedura bloki uchun global hisoblanadi va unda qo'llaniladi.

Procedura va funksiyalar mavjud bo'lgan dasturlarni yozishda lokal va global o'zgaruvchilarning nomini shunday tanlash maqsadga muvofiqki bunda ular bir- biri bilan mos tushmasligi kerak. Mos tushgan xolda global obyektlar blok ichida o'z kuchini yo'qotadi, ya'ni bir xil nomli global o'zgaruvchilar aniqlangan joyda ulardan foydalanib bo'lmaydi. Dasturning bajarilishida xatolik paydo bo'ladi. Dastur tuzish jarayonida formal va faktik parametrlarning nomlari ham iloji boricha mos tushmasligi kerak, ya'ni ularning nomlarini har xil tanlash kerak. Bu esa dasturni yanada ko'rgazmaliroq bo'lishiga imkon beradi va procedurani chaqirishda formal parametrlar bilan faktik parametrlarni almashishida tushunmovchilik hosil bo'lmaydi.

Funksiyaga murojaat qilish. Agar proceduraga murojaat qilish uchun procedura operatori xizmat qilsa, procedura - funksiyaga murojaat qilish uchun funksiyani chaqirish xizmat qiladi. Shu narsani ta'kidlash zarurki, "**funksiya**" termini ma'lum bir dasturiy obyektni anglatadi, "**funksiyani chaqirish**" esa - shu dasturiy obyektiga undan foydalanish maqsadida dasturda beriladigan murojaatni bildiradi.

Funksiyaga murojaat standart funksiyalarga o'xshash amalga oshiriladi (**sin, cos, tan va hakoza**) va ifodadagi operandalarni turli xilligi bilan procedurani chaqirishdan farq qiladi. Qayerda ifodani yozish kerak bo'lsa, o'sha joyda funksiya nomi yoziladi, undan keyin qavs ichida faktik parametrlar beriladi. Agar parametrsiz funksiya chaqirilsa, u holda faqatgina funksiyaning nomi ko'rsatiladi.

Daraja funksiyasi faktik parametrlarning har xil qiymatlari uchun to'rt marta chaqiriladi.

Funksiya va procedura parametrlari. Procedura yoki funksiyani tavsiflashda uning sarlavhasida to'rt xil parametrlarni ko'rsatish mumkin:

- 1) **Qiymat parametrlari;**
- 2) **O'zgaruvchi- parametrlar;**
- 3) **Procedura- parametrlari;**
- 4) **Funksiya- parametrlari.**

1. Qiymat parametrlari. Qiymat parametrlari boshlang'ich berilmalarni proceduraga yoki funksiyaga uzatish uchun ishlatiladi, formal parametrlari ro'yxatida ular vergul bilan ajratilgan holda va turi ko'rsatilgan holda beriladi.

Masalan:

a) procedure SAS(i, j:integer; r,z:real);

b) function SAS1(i1, j1:integer; r15:real):real;

Tayinli parametrlar sifatida turi o'xshash ixtiyoriy ifodadan foydalanish mumkin.

Procedura yoki funksiyani chaqirishda tayinli parametrlarning qiymati hisoblanadi va rasmiy parametrlarni boshlang'ich qiymati kabi qo'llaniladi, ya'ni qiymatlarni o'rniga qo'yish amalga oshiriladi. Procedura yoki funksiyani bajarilish jarayonida rasmiy parametrlar o'zgarishi mumkin, lekin bu tayinli parametrlarga hech qanday ta'sir qilmaydi, chunki ular procedura yoki funksiya chaqirishga ega bo'lgan qiymatlarini saqlab qoladi. Shuning uchun qiymat parametrlarini procedura yoki funksiya natijalarini asosiy dasturga uzatishda foydalanish mumkin emas.

Masalan.

```
program AB;  
  var x : integer;  
procedure ADA(y:integer);  
  begin y:=3 ; end;  
begin x:=0; ADA(x);  
  writeln(x)  
end.
```

Bu dasturning bajarilishi natijasida **x=0** qiymat hosil bo'ladi.

2. O'zgaruvchi- parametrlar. O'zgaruvchi parametrlar proceduralar bajarilish natijalarini aniqlash uchun ishlatiladi va rasmiy parametrlar ro'yxatida **var** xizmatchi so'zidan keyin qat'iy turi ko'rsatilgan holda beriladi:

procedure SAS(var k,l:integer; var z:real);

Funksiya sarlavhasida *o'zgaruvchi- parametrlardan* foydalanish tavsiya etilmaydi.

Rasmiy o'zgaruvchi - parametrlarga mos keluvchi tayinli parametrlar faqat o'sha turdagi o'zgaruvchilar bo'lishi mumkin. Proceduraga murojaat qilishda va rasmiy parametrlarni tayinli parametrlar bilan almashtirishda o'zgaruvchilarni qiymatlarini emas, balki joylarini uzatish amalga oshiriladi. Rasmiy parametrlarning hamma o'zgarishi tayinli parametrlarga ham taaluqli bo'ladi.

Paskal tilida rasmiy parametrlar sifatida procedura va funksiya nomlaridan ham foydalanish mumkin.

3. Funksiya - parametrlari. Funksiya parametrlari rasmiy parametrlar ro'yxatida **function** xizmatchi so'zidan keyin funksiya turi ko'rsatilgan holda beriladi.

```
procedure ADA(i,j:integer; var z:integer; function FG:real);  
function SUM(function f(x:real):real;m,k:integer):real;
```

Procedura yoki funksiyani chaqirishda rasmiy procedura parametrlari va funksiya parametrlari o'rniga mos tayinli procedura yoki funksiya nomlarini o'rin almashtirish amalga oshiriladi.

4. Procedura- parametrlari. Procedura parametrlari rasmiy parametrlar ruyxatida procedure xizmatchi so'zidan keyin ko'rsatiladi:

procedure adas(i,j:integer; var z:real; procedure ad22);

Demak **procedura- parametrlari yoki funksiya parametrlari** deb, shunday parametrlarga aytiladiki, bunda shu parametrlar ham o'z navbatida procedura yoki funksiya hisoblanadi. Procedura sarlavhasida procedura-parametrlari uchun faqat rasmiy parametrlarning nomi ko'rsatiladi. Funksiya- parametrlari uchun natija turi ham ko'rsatiladi.

Misol. **program parametr;**
 var a,b,c,d: integer;
 procedure PP(x: nteger; var y: integer);
 var c: integer;
 begin
 c:=1; d:=1; x:=1; y:=1;
 writeln(x,y,c,d);
 end;
 begin
 a:=0; b:=0; c:=1; d:=0;
 PP(a,b); writeln(a,b,c,d);
 end.

Dasturning bajarilishidan quyidagi hosil bo'ladi:

1 1 1 1
0 1 0 1

Bu yerda **x** - qiymat parametri, **y** - o'zgaruvchi parametr, **s** - lokal va **d** - global parametrlar.

Paskal tilida procedura va funksiyadan foydalanishning o'ziga xos xususiyatlari

Rekursiya. Paskal tilida proceduralar va funksiyalar rekursiv bo'lishi mumkin. Procedura yoki funksiya **rekursiv** deyiladi, agar uning o'zida o'ziga murojaat qilish mavjud bo'lsa. Procedura yoki funksiyani bunday chaqirish **rekursiv** tavsiflash natijasida yoki **rekursiv** murojaat qilish natijasida hosil bo'lishi mumkin. Rekursiv tavsiflash quyidagicha tushuniladi: procedura yoki funksiya blokning bajarilish qismida uning o'ziga murojaat qilish qatnashadi. Rekursiv tavsiflashga faktorialni hisoblash funksiyasi misol bo'la oladi:

function fact (n : integer) : integer;
 begin
 if n=1 then fact :=1
 else fact:=n*fact(n-1);
 end.

Rekursiv procedura va funksiyalardan foydalanish dasturni ancha ixcham va ko'rgazmali qiladi, keyin har doim ham yetarlicha samara bermaydi. Bunday masalalarni yechishda iterasion sikllardan foydalanish yaxshi samara beradi. Rekursiyani iterasion usullar mavjud bo'lmagan hollarda qo'llash ma'qul bo'ladi.

Qo'shimcha (zararli) samarasi. *Qo'shimcha (zararli) samara* deb procedura yoki funksiyada berilmalarning ko'zda tutilmagan buzilishiga aytiladi. Paskal tilida procedura va funksiyalarni qo'llashda quyidagicha qo'shimcha samara mavjud bo'lishi mumkin: **procedura yoki funksiyada unga uzatilayotgan boshlang'ich berilmalarning buzilishi; procedura yoki funksiyada global o'zgaruvchilarning buzilishi; o'zgaruvchi- parametrlarni funksiya ichida qo'llash va unga qanday kiymatlarni berish.**

Qo'shimcha samaraning paydo bo'lishi va amallari dastur tuzuvchi tomonidan nazorat qilinishi kerak.

Standart bo'lmagan dasturlash qoidalaridan iloji boricha foydalanmaslikka harakat qilish kerak. Buning uchun procedura yoki funksiyaning hamma parametrlarini qiymat- parametrlariga va o'zgaruvchi parametrlariga bo'lishga, procedura yoki funksiya global o'zgaruvchilardan foydalanmaslikka, funksiya o'zgaruvchi- parametrlarni qo'llamaslikka qattik e'tibor berilishi kerak, agar procedura yoki funksiyaning bajarilish natijasi bir nechta bo'lishi kerak bo'lsa, procedurani qo'llash kerak. Lokal va global o'zgaruvchilar bilan ishlaydigan parametrsiz procedura va funksiya ko'proq foydalanishga e'tibor berish kerak.

O'sib ketuvchi tavsiflash. Agar procedura yoki funksiya bevosita o'z-o'zini chaqirsa, u holda bunday **rekursiya oddiy rekursiya deyiladi.** Butunlay shunaqa holat ham mavjud bo'ladiki, bunda procedura yoki funksiya bitta yoki bir nechta oraliq procedura va funksiyalarni chaqiradi va oxirida birinchi procedura yoki funksiya chaqiradi.

Bunday rekursiya **bilvosita rekursiya** deyiladi. Procedura yoki funksiyalarni bunday murojaat kilishda Paskal tilining zarur koidalarini bir- birida bajarish imkoniyati yuq, chunki xamma procedura va funksiyalar ular chakirishguncha tavsiflangan bo'lishi kerak. Bunday holda procedura va funksiyalarni o'sib boruvchi ko'rinishda tavsiflashdan foydalaniladi. Bu tavsiflash quyidagidan iborat: *procedura yoki funksiya faqat o'zining sarlavhasida tavsifga ega bo'ladi, bundan keyin esa **forward** standart nom qo'yiladi, so'ngra procedura yoki funksiya tavsiflash matni rasmiy parametrlar ruyxatida takrorlanmasdan (procedura va funksiya tavsiflash bulimining ixtiyoriy joyiga) joylashtiriladi.*

Masalan:

```
program misol;
  var a,b:real;
  procedure pap(x:real):forward;
  procedure pas(y:real);
  begin
    .....
    .....
    pap(a);
  end;
  .....
  .....
begin
  .....
```

```

.....
    pas(b);
end;
begin
    .....
    pas(a); pap(b);
end.

```

Sinov savollari

1. Paskal tilida funksiya qanday aniqlanadi? Funksiyaga murojaat qanday amalga oshiriladi?
2. Procedura qanday tavsiflanadi va unga murojaat qanday bajariladi?
3. Lokal va global o'zgaruvchilar nima uchun ishlatiladi?
4. Procedura va funktsiyaning tavsiflashda qanday parametrlar ishlatiladi?
5. Procedura va funktsiyaning o'ziga xos xususiyati nimadan iborat?

Uyga vazifa : Turbo-Paskal tizimida qism dastur bilan mustaqil ishlash qoidalarini mukammal o'rganish va ularni amaliyotga tatbiq etish.

20-Ma'ruza

Mavzu: Paskal tilida murakkab turlar.

Reja:

1. To'plamli turlar.
2. Kombinatsiyalangan turlar.
3. Faylli turlar.

Darsning maqsadi:

1. Talabalarda murakkab turlar va uning ko'rinishlari to'g'risida umumiy va asosiy bilimlarni hosil qilish.
2. Turbo-Paskal tizimida to'plamli, kombinatsiyalangan va faylli turlar bo'yicha mukammal ko'nikmaga ega bo'lish.
3. Turbo-Paskal tizimida murakkab turlar bilan mustaqil ishlash malakalarini shakllantirish.

Tayanch iboralar: tur, murakkab tur, to'plamli, kombinatsiyalangan va faylli turlar, turlarnig sarlavhasi va berilish usullari.

Dars o'tish vositalari: sinf doskasi, o'quv-uslubiy qo'llanmalar, ma'ruza matnlari, komputer, ma'ruza bo'yicha slaydlar, Turbo-Paskal tizimi .

Dars o'tish usuli: namoyish dasturiy vositalat ishtirokida, jonli muloqotli ma'ruza .

Darsning xrono xaritasi – 80 minut.

Tashkiliy qism: xonaning tozaligi, jihozlanilishi, sanitariya holati. Talabalarining davomati– 2 minut.

Talabalar bilimni baholash: o'tilgan mavzuni qisqacha takrorlash, talaba bilan savol javob o'tkazish - 10 minut.

Yangi mavzu bayoni - 55 minut.

Mavzuni o'zlashtirish darajasini aniqlash va mustahkamlash – 10 minut.

Sinov savollari – 5 minut.

Uyga vazifa berish – 3 minut.

Ma'ruza bayoni

To'plamli tur . To'plamlar. Paskal tilida *to'plamli tur* qiymati to'plamdan iborat bo'ladi. To'plam deganda ixtiyoriy obyektlarning tartiblangan majmuasi tushuniladi. To'plam chekli sondagi elementlardan iborat bo'ladi va ularning barchasi bir xil turga tegishli bo'lishi kerak. To'plamning elementlari, **real** turidan tashqari ixtiyoriy skalyar tur qiymatlari bo'lishi mumkin. To'plamning elementlari tegishli bo'lgan tur **bazaviy** deb yuritiladi.

Paskal tilida to'plam ko'pi bilan **256 ta**, kami bilan **0 ta** elementdan iborat bo'lishi mumkin.

To'plam elementlari bir- biridan vergul bilan ajratilgan holda kvadrat qavslar ichida beriladi. Bitta ham elementga ega bo'lmagan **bo'sh to'plam []** ko'rinishda bo'ladi.

Paskal' tilida to'plamlar quyidagicha berilishi mumkin:

[]- bo'sh to'plam ;

[1,3,5,7,9] – elementlari 1,3,5,7,9 butun sonlardan iborat to'plam;

['a','b','c','d','e']- elementlari a,b,c,d,e harflardan iborat to'plam;

[1, k]- elementlari butun sonlar 1 va k dan iborat to'plam;

[1,2..100] – elementlari 1dan 100 gacha butun sonlardan iborat to'plam;

[qizil, oq, sariq, qora] – elementlari sanab o'tiladigan to'plam;

[i .. 2*i]- elementlari i dan 2+i ifodadan iborat to'plam;

[2 .. 2] – bitta elementdan iborat to'plam;

[6 .. 1]- bo'sh to'plam;

[2 .. 2,6 .. 2]- bitta elementdan iborat to'plam.

To'plam elementlarining o'rni va ularning necha marta qatnashishi ahamiyatga ega bo'lmaydi, chunki bitta element necha marta qatnashishidan qat'iy nazar bir marta hisobga olinadi.

Masalan:

[2,3,4,5,4,5,3,2] - to'plam [2,3,4,5] to'plamga teng;

[true, false] to'plam [false,true] to'plamga teng;

To'plamlar umumiy ko'rinishda quyidagicha beriladi:

to'plam nomi = set of komponenta turi;

bu yerda **set (to'plam), of (dan)**- Paskal' tilining xizmatchi so'zlari.

Masalan:

type

belgi = '0' .. '9';

tuplam = set of belgi;

var

b: tuplam;

Bu tavsiflash **b** o'zgaruvchi **tuplam** turiga mansub ekanligini anglatadi va u to'plamdan **['0','5','4'], ['2'], [7','8','0'], []** va hokoza qiymatlarni qabul qilishi mumkin. To'plam o'zgaruvchisini to'g'ridan - to'g'ri o'zgaruvchilar bo'limida ham tavsiflash mumkin:

type

belgi='0' .. '9';

var

tuplam = set of belgi;

a: tuplam;

Masalan:

type

tuplam = set of (3,5,7,11,13);

raqam = set of 1..42;

var

tp: tuplam;

R: raqam;

harf: set of ('a', 'e', 'b', 'c', 'd', 'f');

Ushbu misolda **tp** o'zgaruvchi 3,5,7,11,13 qiymatlarni **R**-esa **1- dan 42 ga-cha** bo'lgan raqamlardan birini qabul qilishi mumkin.

To'plam elementlarining soni **256** tadan oshib ketmasligi, shunga mos ravishda bazaviy tur qiymatlarining nomeri **0..255** oralig'ida bo'lishi kerak.

Paskal' tilida to'plamlar bilan ishlashda munosabat amallari (**'='** *teng*, **'< >'** *teng emas*, **'>='** *katta yoki teng*, **'<='** *kichik yoki teng*) dan, to'plamlarning *birlashmasi, kesishmasi, ayirmasi va in* amallardan foydalanish mumkin. Amallarning bajarilish natijasi ifodaning **chin yoki yolg'on** bo'lishiga bog'liq ravishda **true yoki false** bo'lishi mumkin.

Tenglik ('=')amali . Ikkita **A** va **V** to'plamlar **teng** deyiladi, agar ular bir xil sondagi xuddi shunday elementlardan iborat bo'lsa. Taqqoslanayotgan to'plamlarda elementlarning joylashish o'rni farq qilmaydi.

A to'plam	V to'plam	Ifoda	Natija
[1,2,3,4]	[3,1,2,4]	A= B	true
['a','b','c']	['c','a']	A= B	false
['a' .. 'z']	['z'..'a']	A= B	true

Katta yoki teng (>=)amali. Bu amal to'plam elementlarining tegishlilik xossasi uchun ishlatiladi. **A>=V** ifodaning qiymati **true** bo'ladi, agar **V** to'plamning barcha elementlari **A** to'plamda mavjud bo'lsa. Aks holda ifodaning qiymati **false** bo'ladi.

A to'plam	V to'plam	Ifoda	Natija
[1,2,3,4]	[2,3,4]	A>=B	true
['a' .. 'z']	['b'..'t']	A>=B	true
['z','x','c']	['c','x']	A>=V	false

Kichik yoki teng (<=) amali. Yuqoridagi amal kabi bajariladi, lekin **A<=V** ifoda **true** bo'ladi, agar **A** to'plamning barcha elementlari **V** to'plamga tegishli bo'lsa. Aks holda ifodaning qiymati **false** bo'ladi.

A to'plam	V to'plam	Ifoda	Natija
[1,2,3]	[3,1,2,4]	A< =B	true
['d' .. 'h']	['z'..'a']	A< =B	true
['a','v']	['a','n','v']	A< =B	true

In amali. Biror bir elementni ko'rsatilgan to'plamga tegishli yoki tegishli emasligini aniqlash uchun ishlatiladi. Asosan shartli operatorlarda qo'llaniladi.

A qiymat	Ifoda	Natija
-----------------	--------------	---------------

2	if A in [1,2,3] then ..	true
‘v’	if A in [‘a’..’n’] then..	false
x1	if A in [x0,x1,x2,x3] then..	true

in amali murakkab shartlarni tekshirishni osonlashtirish va ko'rgazmaliroq qilishni bajaradi.

Masalan: *if(a=1) or (a=3) or (a=4) or (a=5) or(a=6) then ..* ifodani, *if a in [1..6] then..* kabi qisqa ko'rinishdagi ifoda ko'rinishda yozishga imkon beradi. Ko'p hollarda bu amalni inkor amali bilan yozishga harakat qiladilar: *X not in M*. Bunday yozuv xato hisoblanadi. U quyidagicha yoziladi: *not (X in M)*.

To'plamlarning birlashmasi (+). Ikkita A va V to'plamlarning birlashmasi deb, shu ikkala to'plamning elementlaridan tashkil topgan uchinchi to'plamga aytiladi.

A to'plam	V to'plam	Ifoda	Natija
[1,3,4]	[1,2,5]	A+B	[1,2,3,4,5]
[‘a’..’d’]	[‘e’..’z’]	A+B	[‘a’..’z’]
[]	[]	A+B	[]

To'plamlarning kesishmasi (*). Ikkita A va V to'plamlarning kesishmasi deb, bir vaqtning o'zida ikkala to'plamni ham tashkil qilgan elementlardan tuzilgan uchinchi to'plamga aytiladi.

A to'plam	V to'plam	Ifoda	Natija
[1,2,3]	[1,4,2,5]	A*B	[1,2]
[‘a’..’z’]	[‘b’..’r’]	A*B	[‘b’..’r’]
[]	[]	A*B	[]

To'plamlarning ayirmasi (-). Ikkita A va V to'plamlarning ayirmasi deb, ikkinchi to'plamda mavjud bo'lmagan birinchi to'plam elementlaridan tuzilgan uchinchi to'plamga aytiladi.

A to'plam	V to'plam	Ifoda	Natija
[1,2,3,4]	[3,4,1]	A-B	[2]
[‘a’..’z’]	[‘d’..’z’]	A-B	[‘a’..’c’]
[x1,x2,x3,x4]	[x4,x1]	A-B	[x2,x3]

Kombinasiyalangan turlar. Yozuvlar.

Yuqorida ta'kidlaganimizdek regulyar turlar bir- xil turga mansub bo'lgan elementlar to'plamini tavsirlavash uchun ishlatiladi. lekin dasturlash jarayonida shunday xollar mavjud bo'ladiki, unda foydalanilayotgan to'plamning yoki mas-sivning elemenlari turli xil turlarga tegishli bo'ladi. Masalan: avtomobil boshqaruvchisi to'g'risidagi ma'lumot uchun mashina nomeri, markasi, hay-dovchining ismi, familiyasi va adresini bitta blokka birlashtirish lozim bo'ladi. Buning uchun Paskal tilida **kombinasiyalangan turlar yoki yozuvlar** qo'llaniladi. Yozuvlar mantiqan bir- biri bilan bog'langan elementlarni bitta blokda guruxlash uchun ishlatiladi. Yozuv - ma'lumotlarning strukturlashgan turi bo'lib, aniqlangan sondagi turli xil turdagi komponentalardan iborat bo'ladi.

Yozuv turini aniqlash **resord** so'zi bilan boshlanib, **end** so'zi bilan tugaydi. Ular orasiga **maydon** deb ataladigan **komponentalar ro'yxati**, ularning maydoni nomi va har bir maydonning turi bilan birgalikda joylashtiriladi. Yozuvlar umumiy ko'rinishda quyidagicha tavsiflanadi:

ture

<turning nomi>= record

<maydon nomi>:<komponenta turi>;

<maydon nomi>: <komponenta turi>;

end;

var

<nom,>:<tur nomi>;

Masalan:

1) **mash=record**

nomer : integer;

marka : string [20];

fio : string[40];

adres : string [60];

end;

var

m, v: mash;

Bu misolda **mash** yozuvi to'rtta komponentadan iborat: **nomer, mashina markasining nomi, haydovchining familiyasi va adresi**. Yozuv maydoniga kirish yozuv turining o'zgaruvchisi orqali amalga oshiriladi. Bizning misolimizda **m** va **v** o'zgaruvchilar **mash** turining o'zgaruvchilari hisob-lanadi.

2) **type**

oy nomi = (yanvar, fevral,mart, aprel, may,iyun, iyul, avgust,);

vaqt=record

kun : 1..31; oy: oy nomi; yil:1900..2000;

end ;

var

bugun, ertaga, kecha; vaqt;

Bu yerda yozuv uchta komponenta **kun, oy, yil** lardan iborat ;

Maydon qiymatiga murojaat qilish nuqta bilan ajratilgan maydon identifikatori va o'zgaruvchi identifikatori orqali amalga oshiriladi. Bunday kombinasiya murakkab nom deb yuritiladi.

Masalan, mash yozuvi maydoniga kirish uchun M. Nomer, M. Marka, M. FIO, M. Adres ko'rinishida murojaat qilish kerak bo'ladi.

Murakkab nomni maydon turini qo'llash mumkin bo'lgan barcha joylarda ishlatish mumkin. Maydonlarga qiymat berish uchun ta'minlash operatori ishlatiladi.

Masalan: M. Nomer:=1678;

M.Marka:='gaz-31';

M.FIO:= 'Salimov A.G';

M.Adres:= 'Orzu ko'chasi 12';

Murakkab nomlarni, shu jumladan kiritish chiqarish operatorlarida ham qo'llash mumkin:

read (M.Nomer, M.Marka, M.FIO,M.Adres);

write ; (M.nomer: 4, M.Marka: 7, M.FIO:12, M. Adres:25);

Ta'minlash operatorini yozuvlarda ham to'liq qo'llash mumkin, buning uchun ular bir-xil turga mansub bo'lishlari kerak.

Masalan : $V:=M$;

Bu operator bajarilgandan keyin **V** yozuv maydonining qiymati **M** yozuvin-ing mos maydonlarining qiymatlariga teng bo'ladi.

Ko'pgina masalalarni yechishda masalalarni yozuvlardan iborat bo'lgan massivlardan foydalanish juda qulay. Ularni quyidagi ko'rinishda tavsiflash mum-kin.

Masalan:

```
type
  Person = record
    FIO:string[20];
    Age : 0..90;
    Prof: string[30];
end;
var
  list: array [1..50] of Person;
```

5 ta belgidan ortiq identifikatorlardan foydalanishda noqulaylik-lar hosil bo'ladi. Bunday muammoni hal qilishda Paskal tilida with operatori ishlatiladi. Bu operatorning umumiy ko'rinishi qo'yidagicha:

WITH<yozuv turining o'zgaruvchisi>do<operator>;

WITH operatorida yozuv turining o'zgaruvchisi bir marta ko'rsatilib, may-don nomlari bilan oddiy o'zgaruvchilar kabi ishlash mumkin, yani yozuvni aniqlovchi maydon identifikatori oldida o'zgaruvchi nomi ko'rsatilmaydi.

**Masalan : *mash* yozuvi maydoniga *wit h* operatori yordamida qiymat ber-
ish:**

```
with M do
begin Nomer:=1545;
  Marka:='Gaz-31';
  FIO:='O'raqov O.';
  Adres:='Yulduz ko'chasi, 15'; end;
```

Paskal tilida bir-biri ichiga joylashgan yozuvlar ishlatilishi mumkin, yani mos ravishda **with** operatori ham ichma-ich joylashgan bo'lishi mumkin:

```
With RV1 do
With RV2 do
With RV3 do yozuvi
```

With RV1, RV2, RV3 do ... yozuviga ekvivalentdir. Bunda ichma-ichlik dara-jasi **9** tadan oshib ketmasligi kerak.

Paskal tilida dasturlash jarayonida ixtiyoriy sondagi variantdan iborat struk-turaga ega bo'lgan yozuvlar ishlatilishi mumkin. Bunday yozuvlar **variantli yozu-
vlar** deb yuritiladi. Variantli yozuvlar bir-biriga o'xshash yozuvlarni birlashtirish uchun ishlatiladi. Ular unchalik shart bo'lmagan aniqlangan va variant qismlardan iborat bo'ladi. Variant qismi **case** operatori orqali beriladi.

```
type
  zap=record
```

```

case<maydon belgisi>:<tur nomi> of
  <1-tanlash konstantasi>:(maydon,...,tur);
  -----
  <n-tanlash konstantasi>:(maydon,...,tur);
end;

```

Har bir variant komponentasi qavslar ichiga olinadi.

```

Masalan: type
  zap=record
    Numer: byte;
    Articul:integer;
    Case Flag: Boolean of
      True :(Cena1:integer);
      False:(Cena2:real);
  end;
var
  pzap=zap;

```

Faylli turlar. Fayllar.

Fayl- bir xil turdagi va bir uzunlikdagi komponentalar ketma -ketligidan tuzilgan ma'lumotlarning strukturlashgan turidir. Fayl komponentasi sifatida ko'proq yozuvlar ishlatiladi. Fayl turlarining tavsifi **file of** xizmatchi so'zlari bilan boshlanadi va undan keyin fayl komponentasining turi beriladi. So'ngra oldinda aniqlanayotgan fayl turining identifikatori beriladi. Komponentalar soni *fayl uzunligi* deb yuritiladi.

Faylli turni tavsiflash umumiy ko'rinishda quyidagicha beriladi:

```

ture
  <tur nomi > =file of <kompanenta turi >;
var
  <identifikator > : <tur nomi >;

```

Masalan. ture

```

  karta = file of record;
  uzun: byte ;
  ves: real;
  end;

```

```

var
  Kartfile = karta ;

```

Yozuvlardan iborat faylli turlar quyidagicha tavsiflanishi mumkin:

```

ture
  <komponenta turi >= record
    <maydon >:<tur> ;
    .....
    <maydon >:<tur> ;
  end;
var
  FV : file of < komponenta turi >;
  RV : < komponenta turi >;

```


Masalan : **Yozuv = record**
 Nam : integer;
 Fio : string [20];
 Maosh : real ;
 end;
 var
 FtVk : file of Yozuv ;
 RtVk :Yozuv;

Amaliyotdan shu narsa ma'lumki, fayllarning bunday tavsiflanishini ularni eslab qolish va undan foydalanish uchun yaxshi samaralidir.

Faylli tur o'zgaruvchilarini ifodalarda ishlatib bo'lmaydi. Fayl magnit disk-larning ichki xotirasida saqlanadi va lozim bo'lgan paytda ma'lumotlarni qayta ish-lash uchun asosiy xotiraga chaqiriladi.

Fayl komponentalariga kirish fayl ko'rsatkichlari (bufer o'zgaruvchisi) or-qali amalga oshiriladi. Fayllarni o'qishda yoki yozishda bu ko'rsatgich keyingi komponentaga o'zgartiriladi va uni qayta ishlashga tayyor qiladi. Bufer o'zgaruvchisining boshqa o'zgaruvchilardan farqi shundan iboratki, uning ifodada va ta'minlash operatorida qatnashishi mumkin emas.

Fayllar bilan ishlash standart procedurasi va funksiyalari.

Paskal tilida fayllar bilan ishlash uchun bir qator standart procedura va funksiyalari ishlatiladi: Ularni tavsiflash uchun quyidagicha belgilashlar kiritamiz: **FV**-faylli o'zgaruvchi; **STR**-satrli ifoda ; **P1, P2,.....,Pn** o'zgaruvchilar ; **n**-butun sonli ifoda

Mavjud proceduralar fayllarni tashkil qilishi va ularga murojaatni amalga oshirish uchun ishlatiladi.

Ular quyidagidan iborat:

Assign (FV, Str) – *faylga nom berish procedurasi;*

STR – fayl nomi bo'lib, **FV** faylli tur o'zgaruvchisiga ta'minlanadi. Bu proce-duraning bajarilishidan boshlab barcha amallar **STR** qiymat bilan aniqlanadigan fayllar ustida olib boriladi.

Rewrite (FV) – *yangi fayl hosil kilish procedurasi.* Bu procedura diskda yangi fayl ochish uchun xizmat kiladi. **Fayl nomi** oldindan **Assign** procedurasida aniqlangan bo'ladi. Agar diskda shu nom bilan fayl mavjud bo'lsa, u holda u o'chib ketadi. Fayl ko'rsatkichi birinchi pozisiyaga o'rnatiladi.

Reset(FV) – *ko'rsatkichni fayl boshiga o'rnatish procedurasi.* Proceduran-ing bajarilishi fayl ko'rsatkichini birinchi komponentaga o'rnatishni ta'minlaydi. Agar procedura mavjud bo'lmagan faylga qo'llanilsa kiritish chiqarishda xatolik hosil bo'ladi.

READ (FV,P) – *fayllardan ma'lumotlarni o'qish procedurasi.* Qiymatlari $P_1, P_2, \dots, P_n$ bo'lgan **FV** faylli o'zgaruvchisi bilan aniqlangan diskli fayldan ma'lumotlarni o'qishni amalga oshiradi. Procedura bajarilib bo'lgandan keyin ko'rsatkich keyingi komponentaga joylashadi.

Write (FV,P) – ma'lumotlarni faylga yozish procedurasi. **FV** o'zgaruvchisi bilan aniqlanadigan $P_1, P_2, \dots, P_n$ o'zgaruvchilar diskli faylga yoziladi. Procedura bajarilgandan keyin ko'rsatkich keyingi komponentaga o'rnatiladi.

Seek (FV,n) – ko'rsatkichni faylda **n** tartib nomerli komponentaga o'rnatish procedurasi. Komponentalarning nomeri 0 dan boshlanadi.

1- komponenta nomeri 0, ikkinchi komponentaning nomeri 1 va hakoza.

Flush (FV) – sektor buferini tozalash procedurasi. Bu procedura bajarilishida buferda joylashgan matn fayllari o'chib ketadi.

Close (FV) – fayllarni yopish procedurasi. Proceduraning bajarilishida **FV** o'zgaruvchi bilan belgilangan faylning yopilishi ta'minlanadi. Agar fayl ochiq bo'lsa, u holda uni yopmasdan turib, dasturdan chiqib ketish maqsadga muvofiq emas.

Erase (FV) – faylni o'chirish procedurasi. Proceduraning bajarilishida **FV** o'zgaruvchili fayl yo'qotiladi. Agar ochiq fayl yo'qotish kerak bo'lsa, u holda uni eng avval **Close** procedurasi yopish lozim.

Rename (FV,STR) – fayl nomini o'zgartirish. Proceduraning bajarilishida disk katalogida **FV** o'zgaruvchi bilan aniqlangan faylga yangi nom berishni amalga oshiradi. Yangi nom **STR** qiymat bilan aniqlanadi.

Truncate (FV) – faylning barcha komponentalarini yo'qotish, ko'rsatkichning turgan joyidan boshlab faylni yozish uchun tayyorlash.

Funksiyalar foydalanuvchiga mavjud fayllar bilan ishlash imkoniyatini yengillashtiruvchi ko'shimcha amallarni bajaradi:

Bu funksiyalar quydagilardan iborat.

EOF (FV) – "fayl tamom" qismini tekshirish funksiyasi. Funksiya qiymati **true** ga teng bo'ladi, agar fayl ko'rsatkichi birdan oxirgi fayl komponentasidan keyin turgan bo'lsa va ixtiyoriy holatda **False** bo'ladi.

File Pos (FV) – komponentaning joriy nomerini aniqlash. Bu komponenta nomeriga teng bo'lgan butun sonli qiymatni hosil qiladi, ya'ni ana shu nomerli komponentada fayl ko'rsatkichi joylashgan bo'ladi. Komponentani nomerlash 0 dan boshlanadi.

FileSize (FV) – fayl uzunligini aniqlash funksiyasi. Funksiya **FV** o'zgaruvchisiga mos bo'lgan faylning komponentalari sonini aniqlaydi va butun sonli qiymat hosil bo'ladi. Bu funksiya asosan faylda biror ma'lumot bor yoki yo'qligini yoki bo'shligini tekshirish uchun ishlatiladi. Agar **FileSize (FV)=0** bo'lsa, u holda fayl bo'sh, boshqa holda faylda axborat mavjud bo'ladi.

IOResult (FV) – xatolik hosil bo'lganda oxirgi kiritish-chiqarish amallarining bajarilish natijasini tekshirish uchun ishlatiladi. Agar xatolik mavjud bo'lsa, funksiya xatolik nomerini hosil qiladi, xato bo'lmasa, 0 hosil bo'ladi.

Matnli fayllar va ular bilan ishlash .

Matnli fayl – bu komponentalari faqat satrlardan, yani belgilar ketma-ketligidan iborat bo'lgan fayl tushuniladi. Bunda satrlar uzunligi **0 dan 255 tagacha** o'zgarishi mumkin. Matnli tur o'zgaruvchisini tavsiflash uchun **text** standart identifikator ishlatiladi.

Uning ko'rinishi qo'yidagicha:

var<identifikator>:text;

Masalan:

var

Qog'oz, qog'oz1: text;

Hujjat, hujjat1, hujjat2 : text;

Matnli fayllar bilan ishlash uchun qo'yidagi procedura va funksiyalar mo'ljallangan:

Assign(FV,STR) - matnli faylga nom berish;

Rewrite(FV) - yangi matnli fayl hosil qilish uchun faylni ochish;

Reset(FV) - mavjud matnli faylni ochish;

Close(FV) - matnli faylni yopish;

Read(FV,Ch) - FV matnli fayldan Ch o'zgaruvchidagi belgini o'qish;

Write(FV,Ch) - FV matnli faylga Ch o'zgaruvchini kiritish;

Readln (FV,STR) – FV matnli fayldan STR satrini o'qish ;

Writeln(FV,STR) - FV matnli faylga STR satrini yozish.

Matnli fayllar bilan ishlashda **readln** va **writeln** proceduralarining hususiy hollari ham ishlatiladi.

Readln(FV) - bu barcha belgilarni o'tkazib yuborish uchun ishlatiladi, yani sa-trga o'tish amalga oshiriladi. Proceduraning bu ko'rinishidagi yozuvi fayl kompo-nentalarini sonini aniqlash uchun ishlatilishi mumkin:

St:=0;

while not eof(FV) do

begin Readln(FV); St:=St+1; end;

Write('faylda ',St:4,'ta satr joylashgan');

Writeln(FV) - faylga bo'sh satrni yuklash uchun ishlatiladi:

for I :=1 to 5 do writeln(FV);

Append(FV)- faylni ochish va ko'rsatkichni fayl oxiriga o'rnatish.

Matnli fayllarni tashkil qilish.

Matnli faylni tashkil qilish uchun eng avvalo fayl o'zgaruvchisini matnli tur sifatida elon qilish kerak bo'ladi. So'ngra qo'yidagi qadamlar-ni amalga oshirish kerak:

faylga nom berish(**Assign** procedurasi);

yo'zish uchun faylni ochish(**Rewrite** procedurasi);

faylga satrlarni yozish(**Writeln** procedurasi);

faylni yozish(**Close** procedurasi).

Masalan : Biron bir matnli faylni hosil qilish uchun, unga ixtiyoriy nom berish , ham saqlab qo'yish qo'yidagicha amalga oshiriladi.

program DemoTextFiles;

type dlina= string[60];

var stro:text; file name:string[14];

S : dlina; rej : Char;

procedure Sordt;

begin

write ('Hosil qilinayotgan faylga nom berish ');

readln(FileName); assign (stro, FileName);

assign(Stro, FileName); rewrite(stro);

```

while true do
    begin readln (S); if S='ZZZ' then begin
        close (stro); exit end ; writeln (stro, S); end;
    end .

```

Matnli fayllarni tashkil etganlar bilan ishlash uchun ularni komponentlariga murojat qilishni amalga oshirish kerak bo'ladi .

U quyidagicha bajariladi.

```

    faylga nom berish (Assign procedurasi);
    o'qish uchun faylni ochish(Reset procedurasi);
    komponentlarni qayta ishlash;
    faylni yopish (Close procedurasi).

```

Masalan. Yuqorida keltirilgan faylning barcha komponentalarini chop etish quyidagicha bajariladi.

```

procedure werta;
begin repeat
    write('chikarilayotgan fayl nomini bering');
    readln(FileName);
    Assign (Stro, FileName);
    Reset(Stro);
    until(IOResult =0) of (FileName='ZZZ');
    while notSeekEof(Stro) do
        begin readln (Stro, S);
        writeln (S) end, end.

```

Matnli fayllarni muharrirlash

Muxarrirlash matnli fayllarni yangi komponentalar xisobiga uzaytirish, mavjud xatoliklarni tuzatishdan iborat. Buning uchun quyidagilar amalga oshiriladi:

```

    faylga nom berish (Assign procedurasi );
    yangi komponentalar kiritish uchun fayl ochish (Append procedurasi);
    yangi komponentalarni yozish (Writeln procedurasi);
    faylni yopish (Close procedurasi).

```

Masalan . Hosil qilingan faylni yangi komponentalar hisobiga kengaytirish quyidagicha bajariladi:

```

procedure Rasht;
begin
    repeat writelin ('Muxarrirlanayotgan fayl nomi');
    readln(FileName);
    assign (Stro, FileName);
    append(Stro);
    until IO Result = 0 ;
    while true do
        begin writeln ('Satrni kiriting');
        readln (S); if S ='zzz' then begin close ( Stro) ;
        exit end ; writeln (Stro, S); end;end.

```

Yuqorida keltirilgan dastur bo'laklari alohida dastur ko'rinishida yozilsa ulardan foydalanishda malum noqulayliklar hosil bo'ladi. Shu-ning uchun bu dasturlarni bitta dasturga birlashtirish kerak bo'ladi.

```
program textfile ;  
   ture dlina = string [ 60];  
   var stro ; text ; rej ; char; s dlina ;  
   while true do begin clr sc;  
   writeln ( 'tartibni ko'rsating' )  
   writeln ( '1: matnli faylni hosil qilish ');  
   writeln ( '2: matnli faylni chop etish');  
   writeln ( '3: faylni kengaytirish');  
   writeln ( '4: dasturdan chiqish');  
   read (rej); writelin; case rej of  
   '1': sordt;  
   '2': obrt;  
   '3': rasht;  
   '4': halt else writeln ('tartib nomerini takrorlash')  
   end; end; end.
```

Sinov savollari.

1. To'plam nima? To'plamli turchi?
2. To'plamli tur qanday tasvirlanadi?
3. Yozuvlar nima uchun ishlatiladi va qanday tasvirlanadi?
4. Kombinasionalangan turni qanday tushunasiz?
5. Yozuvlarga misollar keltiring?
6. faylli tur nima va qanday ko'rinishda yoziladi?
7. Faylli turlar bilan ishlash asoslarini aytib bering?
8. Faylli turlarga misollar keltiring?

Uyga vazifa : *Turbo-Paskal tizimida to'plamli va faylli turlar bilan ishlash qoidalarini mukammal o'rganish va ularni amaloyotga tatbiq etish.*

21-ma'ruza

Mavzu: Turbo –Paskal tizimining qo'shimcha imkoniyatlari. Paskal tilida satrli ma'lumotlar bilan ishlash.

Reja:

1. Belgili va satrli turlar.
2. Satrli ma'lumotlar bilan ishlash amallari.
3. Satrlar bilan ishlash funktsiya va proceduralari.
4. Satrlar bilan ishlashga doir dasturlar.

Darsning maqsadi:

1. Talabalarda belgili turlar to'g'risida umumiy va asosiy bilimlarni hosil qilish.
2. Turbo-Paskal tizimida satrli ma'lumotlar bilan ishlash ko'nikmalarini shakllantirish.

3. Turbo-Paskal tizimida satrlar bilan ishlash funksiya va proceduralaridan mustaqil foydalanish malakalariga ega bo'lish.

Tayanch iboralar: *tur, belgili va satrli tur, satrlar va belgilar bilan ishlash, satrlar bilan ishlash funksiyalari.*

Dars o'tish vositalari: *sinf doskasi, o'quv-uslubiy qo'llanmalar, ma'ruza matnlari, komputer, ma'ruza bo'yicha slaydlar, Turbo-Paskal tizimi .*

Dars o'tish usuli: *namoyish dasturiy vositalat ishtirokida, jonli muloqotli ma'ruza .*

Darsning xrono xaritasi – 80 minut.

Tashkiliy qism: *xonaning tozaligi, jihozlanikishi, sanitariya holati. Talabalarining davomati– 2 minut.*

Talabalar bilimini baholash: *o'tilgan mavzuni qisqacha takrorlash, talaba bilan savol javob o'tkazish - 10 minut.*

Yangi mavzu bayoni - 55 minut.

Mavzuni o'zlashtirish darajasini aniqlash va mustahkamlash – 10 minut.

Sinov savollari – 5 minut.

Uyga vazifa berish – 3 minut.

Ma'ruza bayoni

Paskal sistemasida berilmalar turi va o'zgaruvchilar. Turbo-Paskal sistemasida butun turda berilmalarni tavsiflash uchun (konkret holatlarga bog'liq holda) **integer, ScortLine, Byte, LongInt va Word** xizmatchi so'zlaridan bittasi ishlatiladi.

Haqiqiy turdagi o'zgaruvchilarni tavsiflash uchun **real, single, double, extended va comp** xizmatchi so'zlar qo'llaniladi. Belgili turdagi ma'lumotlar **Shar** turi orqali tavsiflanadi va faqatgina bitta harfdan yoki raqamdan yoki belgidan iborat bo'lib har doim apostrof ichiga olingan bo'ladi.

Belgilar ketma-ketligidan iborat bo'lgan satrlarni tavsiflash uchun **String** so'zi ishlatiladi. Satrlar **0 dan 255** tagacha belgidan iborat bo'lishi mumkin. Agar satr o'zgaruvchilari qabul qiladigan satr uzunliklarini ko'rsatish lozim bo'lsa, **String** so'zidan keyin kvadrat kavslarda belgilar soni ko'rsatiladi: **String[50]; String[100];** va hakoza.

Mantqikiy tur o'zgaruvchilari **true va false** qiymatlardan biron-tasini qabul qilishi mumkin va ular **boolean** so'zi orqali tavsiflanadi.

Bizga ma'lumki Paskal tilida dasturlash jarayonida belgili **char**, hamda satrli **String** turlaridan foydalaniladi. Bu turlar ustida ma'lum bir amallarni bajarish mumkin. Bulardan biri qo'shish yoki yelimlash amalidir.

Masalan.

var

ter,per:string; {255 tagacha belgidan iborat satr}}

ret,ket:char; {1 ta belgidan iborat}

ter2,per2:string[20]; { 20 tagacha belgidan iborat}

begin

ter:='SAMARKAND'; per2:='GUZAL'; ter2:='ShAXAR'; ret:='';

per:=ter + ret + per + ret + ter2; writeln(per);

end.

Bu dastur lavhasining bajarilish natijasida **SAMARKAND GUZAL ShAX-AR** so'zi hosil bo'ladi.

Ana shu funksiya va proceduralardan ba'zilarini keltiramiz.

1. **Concat funksiyasi.** Bu funksiya satrlar ketma-ketligini birlashtirish (kushish) vazifasini bajaradi. Uning umumiy kurinishi quyidagicha:

concat (S1[,S2,...Sn]:string):string;

Bu funksiyaning bajarilishi natijasida **S1** satr qolgan satrlar bilan birlashtiriladi va ular funksiyaning parametrlari hisoblanadi. Turbo-Paskal sistemasida belgilar va satrlar bilan ishlash funksiya va proceduralari mavjud bo'lib, ular belgilardan satrlarni hosil qilish, satrlardan qism satrlarni ajratib olish, satrlarni tashkil etgan belgilar sonini aniqlash va hakoza shunga o'xshash ko'plab amallarni bajarish imkonini beradi.

Misol.

```
var
  a,b,c: string;
begin a:='infor'; b:='matika';
      c:=concat(a,b); writeln(s);
end.
```

Dastur lavhasining bajarilish natijasida a va b satrlar qo'shilib, **informatika** so'zi hosil bo'ladi.

2. **Copy funksiyasi.** Bu funksiya satrlardan satr bo'laklarini ajratib olish uchun ishlatiladi. Uning umumiy ko'rinishi quyidagicha:

copy(S:string;n,m:integer):string;

Bu yerda **s** - berilgan satr; **n** - esa satr bulagi ajratib olinadigan belgi nomeri; **m** - esa ajratib olinadigan belgilar soni.

Masalan.

```
var a,b,c : string;
begin a:='informatika';
      b:=copy(a,3,5); c:=copy(a,8,3);
      writeln(b,":4,c);
end.
```

Bu dasturning bajarilishi natijasida **'forma'** va **'tik'** so'zlari hosil bo'ladi.

3. **Delete procedurasi.** Bu procedura satrlardan belgilarni yoki suz bo'laklarini o'chirish uchun ishlatiladi. Uning umumiy ko'rinishi quyidagicha:

delete(var S:string; n:integer; m:integer);

Bu yerda **S** berilgan satr; **n** - **S** satrda o'chirish boshlanadigan belgining nomeri; **m** - o'chiriladigan belgilar soni;

Masalan.

```
var a,b : string;
begin a:='informatika';
      delete(a,6,5); writeln(a);
      delete(a,3,3);writeln(a);
end.
```

Bu dasturning bajarilishi natijasida **'infor'** va **'in'** so'zlari hosil bo'ladi.

4. Insert procedurasi. Bu procedura berilgan satrga biror satr bulagini kushish uchun ishlatiladi. Uning umumiy ko'rinishi quyidagicha bo'ladi:

insert(S1:string; var S:string;n:integer);

Bu yerda **S** - berilgan satr; **S1** - berilgan satrga qo'shiladigan satr bo'lagi; **n** -S satrga qo'shish amalga oshiriladigan belgining nomeri.

Masalan.

```
var a,b:string;  
begin a:='infora'; b:='matik';  
      insert(b,a,6); writeln(a);  
end.
```

5. Length funksiyasi. Bu funksiya burilgan satrlarning yoki satr bo'laklarining uzunliklarini aniqlash uchun ishlatiladi. Uning umumiy ko'rinishi quyidagicha yoziladi:

length(S:string):integer;

Bu yerda **S** - berilgan satr. Funksiyaning bajarilish natijasi sonidan iborat bo'ladi.

Masalan.

```
var S:string; T:integer;  
begin S:='INFORMATIKA';  
      T:= Length(S); writeln('T=',T);  
end.
```

Bu dasturning bajarilishi natijasida T=11 hosil bo'ladi, ya'ni berilgan satr 11 ta belgidan iborat ekanligi aniqlanadi.

6. POS funksiyasi. Bu funksiya berilgan satrda ayrim satr bo'laklarini ishlashni amalga oshirish uchun ishlatiladi. Uning umumiy ko'rinishi quyidagicha:

pos(S1, S : string):byte;

Bu yerda **S1** - S satrda izlanayotgan satr bulagi; **S** - berilgan satr; **POS** funksiyasi **S1** satr bulagi **S** satrning nechanchi belgisidan boshlab topilgani to'g'risida axborot beradi. Agar bu satr bo'lagi berilgan satrda mavjud bo'lmasa 0 hosil bo'ladi.

Masalan.

```
var s:string; i,j:integer;  
begin s:='informatika'; i:= pos('mat',s);  
      j:= pos('o',s); writeln(i,',',j);  
end.
```

Bu dasturning bajarilishi natijasida 6 va 4 sonlari hosil bo'ladi.

7. STR procedurasi. Bu procedura berilgan sonli qiymatni satrli qiymatga aylantiradi. Uning umumiy ko'rinishi quyidagicha:

str(x [:w,[m]],var s :string);

Bu yerda **m** va **n** lar bo'lishi shart bo'lmagan parametrlar bo'lib, mos ravishda satr uzunligi va verguldan keyingi belgilar sonini bildiradi;
x - berilgan sonli qiymat; **S**-esa sonli qiymatning satrga aylantirilgan natijasi.

Masalan.

```
var S:string; X:real;  
begin X:=1234.678;  
      Str(X:2:2,S); writeln(S);  
end.
```


8. VAL procedurasi. Bu procedura satrli kiymatni sonli kiymatga aylantiradi. Uning umumiy kurinishi quyidagicha:

val(s:string; var v; var code:integer);

Bu yerda **S** - berilgan satr; **v** - **s** ning sonli tasviri joylashadigan sonli uzgaruvchi; **code** - butun uzgaruvchi. Agar **S** satrni songa aylantirish mumkin bo'lmasa, **val** ning bajarilish natijasida **code** nol bo'lmagan qiymat qabul qiladi.

Masalan.

```
var s:string; x:real; c:integer;  
begin s:='124.452';  
      val(s,x,c); writeln(x:2:2);  
end.
```

Sinov savollari.

1. Turbo Paskal tizimining qanday qo'shimcha imkoniyatlarini bilasiz?
2. Harfli va satrli ma'lumotlarni qanday tushunasiz?
3. Satrli ma'lumotlar dasturda qanday tavsiflanadi?
4. Satrli ma'lumotlar bilan qanday amallar bajariladi?
5. Satrli ma'lumotlar bilan ishlash uchun qanday funksiya va proceduralarni bilasiz? Misollar keltiring?

Uyga vazifa : Turbo-Paskal tizimida satrli ma'lumotlar va ularning funktsiya-proceduralari bilan ishlash qoidalarini mukammal o'rganing, ularni amaliyotga tatbiq etishga doir misollar keltiring.

Adabiyotlar

1. Shafrin Yu. Osnovi kompyuternoy texnologii. Uchebnoi posobie- M: 1997, - 560 b.
2. Raxmanqulova S.I. IBM PC shaxsiy kompyuterda ishlash.- T:, NMK "Sharq"-INSTAR, 1996 y.
3. Figurnov V. E. IBM PC dlya pozovatelya. - M; Finanssi i statistika, 1990 g.
4. Ortiqov A., Mamatqulov A. IBM PC kompyuteridan foydalanish.- Toshkent, "Qonis" 1992 y.
5. Volvachev A.K., Krisevich V. S. Programmirovaniye na yazika Paskal dlya personalnix EC EBM- Minsk, Visshaya shkola, 1989 g.
6. Vasyukova N.D., Tyulyaeva V.V. Praktikum po osnovam programmirovaniya. Yazik Paskal- M; Visshaya shkola, 1991 g.
7. Zuev E.A. Yazik programmiravaniya Turbo Paskal 6.0, 7.0- M; Radioi svyaz, 1993.
8. Faysman A. Profессиональное программирование на языке Paskal- M; Nauka, 1989 g.
9. Aminov I. Paskal dasturlash tili. O'quv qullanma- SamDU, Samarqand, 1996 y.
10. Aminov I.B. Informatika va informatsion texnologiyalar. Ma'ruzalar matni. 2005 yil. SamDU.
11. Aminov I.B., Eshtemirov S., Nomozov F. Informatika va informatsion texnologiyalar fanidan laboratoriya ishlari. Uslubiy qo'llanma. Samarqand, SamDU, 2008 yil.

22 - Ma'ruza

Mavzu: Paskal tilida modullar.

Reja:

1. Modul tushunchasi.
2. Standart modullar.
3. CRT moduli va u bilan ishlash funksiyalari.

Darsning maqsadi:

1. Talabalarda modul va ularning turlari to'g'risida umumiy va asosiy bilimlarni hosil qilish.
2. Turbo-Paskal tizimida standart modullar va ular bilan ishlash bo'yicha mukammal ko'nikmaga ega bo'lish.
3. Turbo-Paskal tizimida Crt moduli va uning funksiyalari bilan mustaqil ishlash malakalarini shakllantirish.

Tayanch iboralar: modul, standart modul, modul turlari, modul ko'rinishi, modul yaratish strukturasi, System, Crt modullari va ularning funksiyalari.

Dars o'tish vositalari: sinf doskasi, o'quv-uslubiy qo'llanmalar, ma'ruza matnlari, komputer, ma'ruza bo'yicha slaydlar, Turbo-Paskal tizimi.

Dars o'tish usuli: namoyish dasturiy vositalat ishtirokida, jonli muloqotli ma'ruza.

Darsning xrono xaritasi – 80 minut.

Tashkiliy qism: xonaning tozaligi, jihozlanikishi, sanitariya holati. Talabalarining davomati – 2 minut.

Talabalar bilimini baholash: o'tilgan mavzuni qisqacha takrorlash, talaba bilan savol javob o'tkazish - 10 minut.

Yangi mavzu bayoni - 55 minut.

Mavzuni o'zlashtirish darajasini aniqlash va mustahkamlash – 10 minut.

Sinov savollari – 5 minut.

Uyga vazifa berish – 3 minut.

Ma'ruza bayoni

Turbo Paskal modullari.

Modullar. Turbo Paskalda modullar yaratib va ularni kompilyasiya qilib masalalarni yechish mumkin. O'z vaqtida Turbo Paskal' yaratuvchilari IBM kompyuterlarining barcha imkoniyatlarini o'z ichiga olgan funksiya va proceduralarni modullar ko'rinishida tavsiflaganlar va yaratganlar.

Bu modullarga quyidagilar kiradi:

System – standart funksiya va proceduralardan iborat. **System** moduli barcha dasturlarda avtomatik qo'llaniladi.

Dos – MS DOS operatsion sistemasi vositalaridan foydalanish imkonini beruvchi funksiya va proceduralardan iborat.

Crt - IBM dinamiklari, klavitura va ekran bilan ishlash uchun proceduralar majmuasidan iborat.

Graph - kompyuter grafik imkoniyatlaridan foydalanishni amalga oshirish uchun ishlatiladigan proceduralar to'plami.

Printer - chop etish qurilmasi bilan ishlash uchun kichik modul.

Graph3 – Turbo Paskalning 3- versiyasi uchun to'liq grafik dasturlardan iborat.

Modullardan foydalanish uchun dastur boshida quyidagi satrlar yoziladi:

Program- dastur nomi;

Uses modul nomi;

Agar dasturda bir nechta modullardan foydalanilsa, quyidagicha yoziladi:

Uses 1- modul nomi, 2- modul nomi, ..., N- modul nomi;

Tovushlarni hosil qilish uchun dastur tuzishda **Crt** moduliga tegishli quyidagi proceduralar ishlatiladi:

Sound (i) - chastota bilan ovozni hosil qilish;

Delay (i)- dasturning bajarilishini **i**- millisekundga ushlab turadi;

Nosound – ovoz hosil qilishni to'xtatish;

Key Pressed- funksiyasi, agar qandaydir klavishaturani bossa, **true** qiymatni hosil qiladi, aks holda **false** qiymat hosil bo'ladi.

Random (i) - funksiyasi **0** dan **1** gacha oraliqda tasodifiy sonlarni hosil qiladi.

program Spring;

uses CRT;

begin repeat sound (1400+ Random (600));

delay (random(10)); nosound;

delay (random(1300)); until keypressed; nosound;

end.

Modullarni yaratish.

Turbo Paskalda har bir foydalanuvchi o'zining shaxsiy modullarini yaratishi mumkin. Modullarning umumiy strukturasi quyidagicha:

unit modul nomi;

interfase

.....

{ochiq taviflash bo'limi }

implementation

{yopiq tavsiflash bo'limi}

begin

{inializasiya qilish bulimi}

.....

end.

Modul **unit** xizmatchi so'zidan boshlanib, undan keyin modul nomi keladi. Undan keyin bo'lim boshlanishini ifodalovchi **interfase** xizmatchi so'zi keladi.

Masalan. Ikki sonning kichigini **Min(x,y)** va kattasini **Max(x,y)** ni topish moduli quyidagicha hosil qilinadi:

Unit ISEK;

```

interface
function Min(x,y:integer):integer;
function Max(x,y:integer):integer;
implementation {yopiq tasvirlash bo'limi}
function Min (x,y: integer):integer;
begin if x<=y then Min=X else Min=Y; end;
function Max(x,y:integer):integer;
begin if x>=y then Max:=x else Max:=Y; end;
end.

```

Bu yozilgan modulni kompilyasiya qilish kerak bo'ladi. Kompilyasiya natijasi **ISEK.TPU** nomli fayldan iborat bo'ladi.

System moduli

System modulining procedura va funksiyalari barcha dasturlar uchun o'rinli. **System** moduli uchun **uses** so'zini yozish shart emas. **System** modulida dastur ishini boshqarish proceduralari quyidagilar:

1. *Exit* procedurasi- joriy blokdan zudlik bilan chiqishni amalga oshiradi.
2. *Halt* procedurasi – dastur bajarilishini to'xtatadi va boshqa-rishni operasion sistemaga o'tkazadi.

Turlarni hosil qilish funksiyalari

1. *Chr(x)*-ASCII jadvalida tartib nomeri berilgan songa teng bo'lgan belgini hosil qiladi.
2. *Ord (x)* - berilgan tur qiymati bo'yicha tartib nomerini hosil qiladi.
3. *Round* - haqiqiy tur qiymatlarini yaxlitlash.

Arifmetik funksiyalar

1. *abs (x)*- argumentning absolyut qiymati;
2. *arctan (x)*- argumentning argtangensi;
3. *cos (x)* – argumentning kosinusi ;
4. *exp (x)*- argument eksponentasi;
5. *frac(x)*- argumentning kasr qiymatini hosil qilish;
6. *Int (x)*- argumentning butun qiymatini hosil qilish;
7. *Ln(x)*- argumentning natural logarifimi;
8. *Pi* – P sonining qiymatini hosil qiladi;
9. *Sin(x)*- argumentning sinusi.
10. *Sqr(x)*- argumentning kvadrati;
11. *SQRT(x)*- argumentning kvadrat ildizi;

Sanab o'tiladigan tur procedura va funksiyalari

1. *Dec* procedurasi- o'zgaruvchi qiymatini kamaytiradi;
2. *Ins* procedura- o'zgaruvchi qiymatini bittaga oshiradi;
3. *Odd* procedurasi – argumentning toqligini tekshiradi;
4. *Pred* procedura – argumentning oldingi qiymatini hosil qiladi;

Bundan tashqari **System** quyidagi funksiyalar ham ishlatiladi:

1. *Lo (x)* funksiyasi – argumentning kichik baytini hosil qiladi.
2. *MOVE* procedurasi berilgan son bayti operativ xotirasining bir joydan boshqa joyga nusxalash.

3. **Random** – tasodifiy sonni hosil qiladi.

```
var i:integer; {100 tasodifiy belgini hosil qilish}
begin for i:=1 to 100 write (chr(Random(256)));
      Readln; end.
```
4. **Randomize** procedurasi–tasodifiy sonlar generatorini hosil qildi:

```
var i: integer;
begin Randomize; for i:=1 to 100 do
      write (Chr(Random(256)));
      readln; end.
```
5. **SWAP** funksiyasi- argumentning katta va kichik baytlari o'rnini almashtiradi.
6. **UpCase** –kichik harflarni bosh harflarga o'zgartiradi.

CRT moduli

CRT moduli kompyuter ekrani bilan ishlash imkoniyatlarini amalga oshirish uchun procedura va funksiyalardan iborat. Kompyuter ekranida matn holatida **25 ta** matn satri va har bir satrda **80 ta** belgi joylashishi mumkin. Ekranda belgilar va belgilar foni rangli, ya'ni **16 ta** rangdan, yoki oq qora rangdan iborat bo'lishi mumkin. Oyna- bu ekranning ayrim to'g'ri burchakli qismi bo'lib, u ham alohida ajratilgan ekran vazifasini o'taydi. Ekranda oynani quyi o'ng va yuqori chap koordinatalari parametrlarini berib **Window** procedurasi orqali hosil qilish mumkin:

CRT moduli quyidagi funksiya va proceduralardan iborat:

1. **AssignCrt**- procedurasi- **CRT** qurilmasi uchun matn faylini belgilaydi.
2. **ClrEol** procedurasi- ekranda kursor turgan joydan satr oxirigacha belgilarni o'chiradi.
3. **ClrScr** procedurasi- ekranni tozalaydi va kursorni yuqori chap burchagiga joylashtiradi.

```
Uses CRT; var I: integer;
begin textbackGround (White);
      ClrScr; for I:=3 to 10 do textbackground (I);
      Window (3*I, I,80-3*I, 22-I); ClrScr; end; end.
```

4. **Delay**- procedurasi – berilgan songa teng millisekund bajarilishini ushlab turadi.

```
Uses CRT;
var I: integer;
begin for I:=1 to 5 do
      begin Sound (100); Delay (200); Nosound;
            Delay (500); Sound (100); Delay (100); Nosound; Delay (500);
            end;end.
```

5. **Delline** procedurasi- kursor turgan satrni olib tashlash;
6. **GOTOXY**- kursorni berilgan satr va ustunga joylashtirish;
7. **HighVideo** –chiqarilayotgan belgilarni yuqori yorug'likda hosil qiladi;
8. **Inline** procedurasi- kursor turgan o'ringa bo'sh satrni hosil qiladi;
9. **KeyPreesed** funksiyasi- biror klavisha bosilsa **true** qiymat hosil qiladi, aks holda **false** hosil bo'ladi.

```

Uses Crt;
begin repeat
  textbackGround (Random(8));
  Window(30, 8, 50,16); ClrScr; Delay (300);
until keyppressed; end.

```

10. **LowVidio** – belgilar quyi yorug'likda hosil qilinadi;
11. **NormVidio** – belgilar normal yorug'likda hosil qilinadi;
12. **Nosound**-ovoz hosil qilish dinamikasini o'chiradi;
13. **RedKey** – klaviaturadan belgini o'qiydi;
14. **Sound** – ovoz hosil qilish dinamikasini ishga tushiradi va berilgan chastota bo'yicha ovoz hosil qiladi.

```

uses Ctr;
begin repeat
  Sound (Random(100)+100));
until KeyPressed; Nosound;
end.

```

15. **textbackground** – fon rangini o'rnatadi ;
16. **textcolor** – matn rangini o'rnatadi;
17. **textmode** – matn tartibini o'rnatadi;
18. **WhereX** –x koordinatani hosil qiladi;
19. **WhereY** –y koordinatani hosil qiladi;
20. **Window** – ekranda matn oynasini hosil qiladi: **Window (x1, y1, x2, y2);**

Bu yerda **X1**- oyna chap yuqori burchagi satri, **U1**- oynani chap yuqori burchagi ustuni, **X2**- oynaning o'ng quyi burchagi satri, **U2**- oynaning o'ng quyi burchagi satri.

Sinov uchun savollar:

1. Modul va uning asosiy imkoniyatlari nimadan iborat?
2. Modulning qanday turlarini bilasiz?
3. Standart modullar nima?
4. Modullar yaratishning umumiy ko'rinishi qanday?
5. Modullardan qanday foydalaniladi?
6. System mmoduli va uning funksiyalarini ayting.
7. Crt moduli , vazifasi va funksiyalari?

Uyga vazifa : Turbo-Paskal tizimida standart modullar bilan ishlash qoidalarini mukammal o'rganing va ularni amaloyotga tatbiq etish bo'yicha mustaqil modullar tuzishga harakat qiling.

23-Ma'ruza

Mavzu: Paskal tilining grafik imkoniyatlari.

Reja:

1. Turbo Paskal tiziminng grafik holati. Graph moduli.
2. Graph modulining procedura va funksiyalari.
3. Obyektlarni harakatlantirish va boshqarish.

4. Funksiyalar grafigini yasash.

Darsning maqsadi:

1. Talabalarda grafik ma'lumotlar va ularning turlari to'g'risida umumiy va asosiy bilimlarni hosil qilish.
2. Turbo-Paskal tizimida grafik holatda ishlash bo'yicha mukammal tasavvurga ega bo'lish.
3. Turbo-Paskal tizimida Graph moduli funksiya –proceduralari bilan mustaqil ishlash malakalarini shakllantirish.

Tayanch iboralar: *modul, Graph moduli, grafik holat, grafik holatga kirish va chiqish, tasvirlar yasash, ob'ektlar bilan ishlash, funksiyalar grafigi.*

Dars o'tish vositalari: *sinf doskasi, o'quv-uslubiy qo'llanmalar, ma'ruza matnlari, kompyuter, ma'ruza bo'yicha slaydlar, Turbo-Paskal tizimi .*

Dars o'tish usuli: *namoyish dasturiy vositalat ishtirokida, jonli muloqotli ma'ruza .*

Darsning xrono xaritasi – 80 minut.

Tashkiliy qism: *xonaning tozaligi, jihozlanikishi, sanitariya holati. Talabalarining davomati– 2 minut.*

Talabalar bilimini baholash: *o'tilgan mavzuni qisqacha takrorlash, talaba bilan savol javob o'tkazish - 10 minut.*

Yangi mavzu bayoni - 55 minut.

Mavzuni o'zlashtirish darajasini aniqlash va mustahkamlash – 10 minut.

Sinov savollari – 5 minut.

Uyga vazifa berish – 3 minut.

Ma'ruza bayoni

Turbo Paskal tizimining grafik holati. Turbo Paskal muhitida standart ish tartiboti- matnli hisoblanadi . Bu tartibotda ekranga faqat kompyuterga ma'lum bo'lgan belgilar chiqariladi, ya'ni matnlar bilan ish olib boriladi.

Grafik tartibotida displey ekrani bir- biriga juda yaqin joylashgan nuqtalar majmuasi sifatida qaraladi. Ekranda mavjud nuqta **piksel** yoki **piksel' (pixel)** deb yuritiladi. Piksellar to'g'ri burchakli jadvalni hosil qiladi. Nuqtalar shu jadval elementlari hisoblanadi va ular ekranning yuqori chap burchagidan boshlab nomerlanadi. Bu nuqtaning koordinatasi **(0,0)** dan iborat bo'ladi.

Matnli ma'lumotlarning belgilarini tasvirlash fiksirlangan sondagi piksellardan iborat bo'lgan matrisadan foydalaniladi. Har bir belgining o'rni to'g'ri burchakli o'lchovdan, *masalan, 8x8, 8x14, 8x16* va hokozolardan iborat bo'ladi. Belgi o'rniga dastur yordamida belgining tasviri joylashtiriladi.

Konkret videoadapterlarning grafik holatlari monitoring imkoniyatlari bilan, ya'ni ekranning piksellarining va hosil qilinishi mumkin bo'lgan ranglarning umumiy soni bilan aniqlanadi.

Zamonaviy monitorlar **800x600 ta, 1024x768 ta** va undan ko'proq nuqtalar bilan ishlash imkoniyatini beradi. Monitoring imkoniyati qancha yuqori bo'lsa, unda hosil qilinayotgan tasvirlar shunchalik sifatli bo'ladi. Ko'plab videoadapterlar bir nechta grafik sahifalar bilan ishlashi mumkin. Grafik sahifa deganda- tezkor xotiraning grafik tasvirlarni hosil qilish uchun ajratilgan sohasi tushuniladi.

Kompyuterda ishlash jarayonida mavjud grafik tartibotlardan birini tanlagan holda, nuqtalar soni va mumkin bo'lgan ranglar sonini o'zgartirish mumkin.

Drayver deganda - kompyuterning biror bir qurilmasi ishini boshqaruvchi dastur tushuniladi. *Grafik drayver* grafik tartibotida displey adapterini boshqaradi. Grafik drayverlar diskda ishchi katalogining **BGI** qism katalogida fayl ko'rinishida joylashgan bo'lib, kengaytmasi **.BGI (Borland Graphic Interfase)** dan iborat bo'ladi.

Turbo Paskal' muhitida **CGA, EGA, VGA, MCGA, PC 3270** kabi vedeoadapterlarning grafik imkoniyatlarini ta'minlovchi standart kutubxona moduli- **GRAPH moduli** ishlatiladi.

Graph kutubxonasi **80** ga yaqin grafik procedura va funksiyalar, shu bilan birga o'nlab o'zgarmaslar va ma'lumotlar turining tavsiflaridan iborat. **Graph** modulining ishga tushirish **uses** xizmatchi so'zi orqali amalga oshiriladi.

Graph moduli- Graph.tpu faylidan iboratdir. Grafik tartibotida ishlash uchun bu fayl kompilyator uchun ishchi fayl bo'lishi kerak. Dastur boshida **Graph.tpu** fayli joylashgan katalogga yo'l ko'rsatilishi kerak.

Graph modulida turli drayverlarni ko'rsatish uchun quyidagi o'zgarmalar ishlatiladi:

Const

Detect = 0; (drayverni avtomatik aniqlash)

CGA= 1;

MCGA= 2;

EGA= 3;

EGA64= 4;

VGA= 9;

Bu turdagi adapterlar ko'p hollarda bir nechta grafik tartibotlarda ishlay oladi. **Graph** modulida bu tartibotlarni ko'rsatish uchun quyidagi o'zgarmaslar ishlatiladi:

const

CGACO= 0; {320x200 nuqta, 4 rang}

CGAC1=1; {320x200 nuqta, 4 rang}

CGAHI= 4; {640x200 nuqta, 4 rang}

EGALO= 0; {640x200 nuqta,16 rang}

EGAHi= 1; {640x350 nuqta, 16 rang}

VGAHi= 2; {640x480 nuqta, 16 rang}

Grafik holatini o'rnatish va undan chiqish tartiblari.

Kompyuter ekranining oddiy ish tartiboti matnli hisoblanadi. Matn tartibotidan grafik tartibotiga o'tish uchun **Graph** modulining **InitGraph** procedurasi ishlatiladi. Uning umumiy ko'rinishi quyidagicha:

InitGraph (D, M, F); - *ekranni grafik holatga o'tkazish*. Bu yerda **D**- drayver nomeri, **M**- tartibot nomeri, **F**- esa kerakli drayver mavjud bo'lgan faylga yo'l. Agar **F** o'zgaruvchi bo'sh (**F=""**) satrdan iborat bo'lsa, drayver joriy katalogdan izlanadi. **D** va **M** lar o'zgaruvchi parametrlardir. **InitGraph** proceduralari ishga tushirayotgan paytda **D** ning qiymati **0** ga teng bo'lsa kerakli drayver va shu

drayver uchun ottimal grafik tartibot avtomatik tarzda aniqlanadi. **Graph** modulida **0** ga teng bo'lgan **Detect** o'zgarishi ishlatiladi.

Grafik holatidan chiqish yoki monitorni boshlang'ich ish holatiga o'tkazish uchun **CloseGraph** procedurasi ishlatiladi.

CloseGraph- grafik holatidan chiqish.

Masalan. Ushbu dastur bir vaqtda grafik holatiga o'tadi va undan chiqadi.

```
uses Graph;  
var D, M; integer;  
begin D:=Detect;  
  initgraph(D, M, 'C:\tp');  
  readln; {“Enter” ni bosilishini kutish}  
  CloseGraph; {grafik holatdan chiqish}  
end.
```

Shunday holatlar ro'y berishi, ya'ni **InitGraph** procedurasi yetarli darajada ishlamasligi mumkin. Masalan, **.BGI** turdagi fayllar turgan joy noto'g'ri ko'rsatilgan bo'lishi yoki grafikani inisializatsiya qilish uchun tezkor xotirada joy yetmasligi mumkin. Bunday holatlarni aniqlash uchun **Graph Result** funksiyasi ishlatiladi. U grafikani inisializatsiya qilish jarayonida quyidagi qiymatlarni qabul qilishi mumkin:

- 0- xato yo'q;
- 2- grafik plati topilmadi;
- 3- drayver fayli topilmadi;
- 4- noto'g'ri drayver;
- 5- grafikni inisializatsiya qilish uchun xotira yetarli emas.

Graph modulining funksiya va proceduralari

Graph moduliga tegishli prosekura va funksiyalarni qarab chiqamiz.

Koordinatalarni o'rnatish procedura va funksiyalari.

Ko'plab grafik procedura va funksiyalarda ekranda joriy o'rinni ko'rsatuvchi **ko'rsatkich**dan foydalaniladi. **Ko'rsatkich**ning matn kursoridan farqi shundaki, u ekranda ko'rinmaydi. Ana shu ko'rsatkichning holati, shu bilan birga umuman grafik holatda har qanday koordinata ekranning yuqori chap burchagi (**0,0**) koordinataga nisbatan beriladi. Shunday qilib, ekranning gorizontaal koordinatasi chapdan o'ngga qarab, vertikal koordinatasi esa yuqoridan pastga qarab o'sib boradi.

SetMaxX va GetMaxY funksiyalari. Joriy ish tartibotida mos ra-vishda maksimal gorizontaal va vertikal koordinatalarini aniqlaydi.

```
Masalan: uses Graph;  
          var a,v: integer;  
          begin a:=Detect; InitGraph (a,v);  
                writeln (GetMaxX, GetMaxY:5); CloseGraph;  
          end.
```

GetX va GetY funksiyalari. Integer turidagi qiymatlar bo'yicha ekran-ning gorizontaal va vertikal koordinatalarini o'rnatadi. Agar oyna o'rnatil-magan bo'lsa, koordinatalar ekranning chap yuqori burchagiga nisbatan olinadi.

SetViewport procedurasi - grafik ekranda to'g'ri burchakli oyna hosil qiladi. Uning umumiy ko'rinishi:

Procedure SetViewport (X1, Y1, X2, Y2: integer; ClipOn: boolean);

Bu yerda X1...Y2 lar oyna burchagining koordinatalari. (X1, Y1)- yuqori chap va (X2, Y2) o'ng quyi burchak koordinatalari, ClipOn – Boolean tipidagi ifoda.

MoveTo procedurasi - ko'rsatkichni yangi joriy holatga o'rnatadi. Uning umumiy ko'rinishi:

Procedure MoveTo(X, Y: integer); Bu yerda X, Y ko'rsatkichning gorizont tal va vertikal yo'nalishlar bo'yicha yangi koordinatalari. Koordinata ekranning yuqori chap burchagiga asosan olinadi.

ClearDevice procedurasi – grafik oynani tozalaydi.

Uning umumiy ko'rinishi:

Procedure ClearDevice; Bu procedura bajarilganda ekran tozalanadi, ko'rsatkich ekranning yuqori chap burchagiga joylashadi va ekran fon rangi bilan bo'yaladi.

Ko'pburchaklar hosil qilish funksiya va proceduralar

Rectangle procedurasi – ko'rsatilgan burchak koordinatalari bo'yicha to'g'ri burchak hosil qilish. Uning umumiy ko'rinishi:

Procedure Rectangle (X1, Y1, X2, Y2: integer); bu yerda X2 – X1... to'g'ri burchak burchaklarining koordinatalari. (X1, Y1) – yuqori chap burchak va (X2, Y1) quyi o'ng burchak koordinatalari. To'g'ri burchak joriy rang va joriy chiziq qalinligi bo'yicha hosil qilinadi.

Masalan. Ichma-ich joylashgan 10 ta to'g'ri burchak hosil qilinadi.

uses Graph, Crt;

var d, r, e, x1, y1, x2, y2, dx, dy: integer;

begin d:=delect; initgraph (d, r, '');

E:= Graphresult; if e<> grok then writeln (graphErrorMsg (e))

else begin dx:=getmaxX div 20; dy:= getmaxY div 20;

for d:=0 to 9 do Rectangle (d*dx, d*dy, CetMaxA - d*dx,

GetMaxY- d*dy); if ReadKey =# 0 then d:= ord (ReadKey);

Slosegraph; end;

end.

DrawPoly procedurasi – egilish nuqtalarining koordinatalari bo'yicha ixtiyoriy siniq chiziqlarni chizadi. Uning umumiy ko'rinishi:

Procedure DrawPoly(N:Word;Var Points); Bu yerda N – egilish nuqtalarining soni; Points – nuqta koordinatalarini tashkil etuvchilar.

type

Point Type = record

x, y: Word;

end;

Chizish jarayonida joriy rang va joriy chiziq qalinligi qo'llaniladi.

Masalan. Sinus funksiyasi grafigini hosil qilish.

uses graph;

const N=100;

var d, r, e: integer;

```

        m: array [0.. N+1] of Pointtype; K: Word;
    begin d:=delect; InitGraph (d, r, '');
    e:=graphresult; if e <> gr OK then writeln (GraphErrorMsg (e))
    else begin for k:=0 to N do with m[k] do
        begin X:=trunc (K*GetMaxX/N)
            y:=trunc (GetMaxY*(-sin (2*Pi*K/N)+1)/2) end;
        m [succ (N)]. x:=m [0]. X;
        m [succ (n)]. y:=m [0]. y;
    DrawPoly (N+2,m); Readln; closeGraph; end end.

```

Yoy, aylana va ellipslar hosil qilish funksiya va procedurasi

Sircle procedurasi – aylana chizadi. Uning umumiy ko'rinishi quyidagicha:

Procedure Circle (X, Y: integer; R: word); bu yerda X, Y – aylana markazi-ning koordinatalari, R – aylana radiusi. Aylana joriy rangda hosil qilinadi. Chiziqning qalinligi joriy holatda beriladi. Chiziqning doimiy ko'rinishi **Solidln** orqali beriladi.

Masalan. Quyidagi dastur ekran markazida tasodifiy aylanalar bilan to'ldirilib boriladigan oyna hosil qiladi:

```

uses graph, crt;
var d, r, e, x, y: integer;
begin d:=detect; InitGraph (d, r, '');
e:=graphResult; if e <> gr OK then
writeln (GraphErrorMsg (e)) else begin
X:=getmaxX div 4 ; y:=getmaxY div 4;
rectangle (X, Y, 3*X, 3*Y);
SetviewPort (x+1, y+1, 3+x-1, 3+y-1, ClipOn);
repeat
Set Color (Succ (Random (White)));
SetLineStyle (0, 0, 2+Random (2)+1);
X:= Random (getmaxX); Y:=random (getMaxY);
circle (x, y, Random (getMaxYdivY)); until KeyPressed;
if readkey =#0 then x:=ord (Read Key);
closegraph; end end.

```

Arc procedurasi – aylana yoyini chizadi. Uning umumiy ko'rinishi quyidagicha:

Procedure Arc (X, Y: integer; BegA, EndA, B:Word); Bu yerda X, Y – markaz koordinatalari. **BegA, EndA** – yoyning boshlang'ich va oxirgi burchaklari. **R**- radius. Burchaklar soat strelkasiga teskari holda o'lchanadi va graduslarda ko'rsatiladi. Nol burchak vektorning chapdan o'ngga gorizontallik yo'nalishiga mos keladi. Agar boshlang'ich burchak **0** va oxirgi burchak **359** bo'lsa, u holda to'liq aylana hosil bo'ladi.

Ellipse procedurasi – ellips yoyini chizadi. Uning umumiy ko'rinishi:

Procedure Ellipse(X,Y:integer,BegA, EndA, RX, RY:Word); bu yerda **X, Y** – markaz koordinatalari; **BegA, EndA** – yoyning mos ravishda boshlang'ich va oxirgi burchaklari; **RX, RY** – ellipsning gorizonta va vertika radiuslari.

Masalan: Quyidagi dasturda turli xil munosabatdagi radiuslarda uchta el-lips hosil qilinadi.

```
Uses Graph, CRT;
var d,r,e: Integer;  xa,ya: Word;
begin
  {Grafikni inisializasiya qilish}
  d := Detect; InitGraph(d, r, '');
  e := GraphResult; if e <> grOK then
  WriteLn(GraphErrorMsg(e))
  else
  begin
    {Birinchi grafik}
    OutTextXY(50,40,'RX = RY'); {Yozuv}
    Line (0,100,160,100); {Os X}
    Line (80,55,80,145); {Os Y}
    Ellipse (80,100,180,90,40,40);
    {Ikkinchi grafik}
    OutTextXY(260,40,'RX = 5*RY');
    Line (190,100,410,100);
    Line (300,55,300,145);
    Ellipse (300,100,0,359,100,20);
    {Uchinchi grafik}
    OutTextXY(465,40,'Aspect Ratio');
    Line (440,100,600,100);
    Line (520,55,520,145);
    GetAspectRatio(xa, ya);
    Ellipse (520,100,0,270,40,round(40*(xa/ya)));
    if ReadKey=#0 then d := ord(ReadKey);
    CloseGraph
  end
end.
```

Rang berish va sohani bo'yash funksiya va proceduralari.

SetColor procedurasi- hosil qilinayotgan chiziq va belgilar uchun joriy rangni o'rnatadi. Uning umumiy ko'rinishi:

Procedure Setcolor (Color: word); Bu yerda **color**- joriy rang.

SetbkColor procedurasi- fon rangini o'rnatadi. Uning umumiy ko'ri-nishi:

Procedure SetbkColor (Color: word); Bu yerda **Color**- fon rangi.

Masalan. Ushbu dastur bir- birining ichiga joylashgan o'nta to'rt-burchakni hosil qiladi. Bunda fon rangi ketma- ket o'zgarib turadi. Dasturdan chiqish uchun ixtiyoriy tugmachani bosish yetarli.

```
Uses Graph, CRT;
const  NC: array [0..15] of String [12] =
```

```

        ('Black','Blue','Green','Cyan','Red','Magenta','Brown',
        'LightGray','DarkGray','LightBlue','LightGreen1',
        'LightCyan1','LightRed','LightMagenta','Yellow','White');
    var d, r, e, k, color, dx, dy: Integer;
begin {Grafikni inisializasiya kilish}
    d := Detect; InitGraph(d, r, ' ');
    e := GraphResult; if e <> grOK then
        WriteLn(GraphErrorMsg(e)) else
    begin {Ekran markaziga matnni chiqaramiz}
        OutTextXY(200,GetMaxY div 2,'BACKGROUND COLOR');
        dx := GetMaxX div 30; {Priramyeniye dliny}
        dy := GetMaxY div 25; {Priramyeniye vysoty}
        for k := 0 to 9 do {10 ta to'g'riburchakni hosil qilamiz}
            Rectangle(k*dx,k*dy,GetMaxX-k*dx,GetMaxY-k*dy);
            color := black; {fonning boshlang'ich rangi}
            repeat {fonni almashish sikli}
                SetBkColor(color);
                SetFillStyle(0,Color);
                Bar(345,GetMaxY div 2,440,GetMaxY div 2+8);
                OutTextXY(345,GetMaxY div 2,NC[color]);
                delay(1000); inc(color);
            if color > White then
                color := Black until KeyPressed;
            if ReadKey=#0 then
                k := ord(ReadKey);
            CloseGraph
        end
    end.

```

Setpalette procedurasi- palitra ranglaridan birini yangi rangga almashtiradi. Uning umumiy ko'rinishi:

procedure Setpalette (N: Word; Color: sholtnt); Bu yerda N- palitrada rang nomeri; **color**- qayta o'rnatilayotgan rang nomeri. Bu proceduraga murojat qilingandan so'ng N nomerli rang **Solor** rangini oladi.

Masalan, Setpalette(2, white) operator bajarilsa, boshlang'ich ikki nomerli (**cyan**) bryuza rangi oq rangga almashadi.

Quyidagi dastur ekranga bir nechta to'g'ri chiziqlarni chiqaradi va tasodifiy holda palitra rangini almashtiradi.

```

Uses Graph, CRT;
var d,r,e,N,k,color: Integer;
    Palette : PaletteTyper;
begin {Grafikni inisializasiya kilish}
    d := Detect; InitGraph(d, r, ' ');
    e := GraphResult; if e <> grOK then
        WriteLn(GraphErrorMsg(e)) else
    begin {Uzluksiz yo'g'on chiziqni tanlaymiz}

```

```

        SetLineStyle(SolidLn, 0, ThickWidth);
        GetPalette(Palette) ; {Joriy palitra}
        for Color := 0 to Palette.Size-1 do
        begin
            SetColor(Color);
            Line(GetMaxX div 3, Color*10, 2*GetMaxX div 3, Color*10)
        end; {Palitrani almashtiramiz va foydalanuvchini harakatini
kutamiz}

```

```

        while not KeyPressed do
        for e := 0 to Palette.Size-1 do
        SetPalette(e, Random(Palette.Size));
        if ReadKey=#0 then d := ord(ReadKey);
        CloseGraph
        end end.

```

Setallpalette procedurasi- bir vaqtnig o'zida palitraning bir nechta rangini o'zgartiradi. Uning umumiy ko'rinishi:

Procedure Setallpalette (var Palette);

Palette parametri procedura sarlavhasida turlanmagan parametr sifatida tavsiflanadi.

Masalan: Quyidagi dasturda palitraning barcha ranglari bir vaqtda almashadi.

```

Uses Graph, CRT;
var Palette: array [0..MaxColors] of Shortint;
    d,r,e,k: Integer; begin
    {Grafikni inisializasiya kilish}
    d := Detect; InitGraph(d, r, '');
    e := GraphResult; if e <> grOk then
    WriteLn(GraphErrorMsg(e)) else
begin
    {Uzluksiz yo'g'on chiziq tanlanadi}
    SetLineStyle(SolidLn, 0, ThickWidth);
    {Mumkin bo'lgan barcha ranglarda chiziq chiqariladi}
    for k := 1 to GetMaxColor do
begin
    SetColor(k);
    Line(GetMaxX div 3, k*10, 2*GetMaxX div 3, k*10)
end;
    Palette[0] := MaxColors; {Razmer palitry}
    repeat {Palitrani almashtirish sikli}
    for k := 1 to MaxColors do
    Palette[k] := Random(succ(MaxCoLors));
    SetAllPalette(Palette)
    until KeyPressed;
    if ReadKey=#0 then k := ord(ReadKey);
    CloseGraph

```

end
end.

Setfillstyle procedurasi- *sohani to'ldirish turi va rangni o'rnatadi.* Uning umumiy ko'rinishi quyidagicha:

Procedure Setfillstyle (fill,color: word);

Bu yerda **fill** to'ldirish turi; **color-** to'ldirish rangi.

To'ldirish yordamida tasvirning qandaydir bo'lagini doimiy takrorlanuvchi **uzor-lar** bilan to'ldirish mumkin.

To'ldirish turini ko'rsatish uchun quyidagi aniq o'zgarmaslardan foydalaniladi:

Const

Emptyfill= 0; {fon bilan to'ldirish} interleavrfill= 9; {##}

SoledFill= 1;{to'liq to'ldirish} Widedotfill= 10; {....}

LineFill= 2; {----bilan to'ladi} Clasedotfill= 11; {.:...:}

LtSlashFill= 3; {///// } userfill= 12; { }

SlashFill= 4; {//// }

BkSlashFill= 5; { }

LtbkSlashfill= 6;{\\ \\ }

HatchFill= 7; {++++ }

XHatchfill= 8; {xxxx }

Masalan. *Ushbu dastur to'ldirishning barcha standart turlarini nomoyish qiladi.*

Uses Graph, CRT;

var

d,r,e,k,j,x,y: Integer;

begin

{Grafikni inisializasiya kilish}

d := Detect; InitGraph(d, r, ' ');

e := GraphResult; if e <> grOk then

WriteLn(GraphErrorMsg(e)) else

begin

x := GetMaxX div 6;{Grafikning holati}

u := GetMaxY div 5;{ekranda}

for j := 0 to 2 do{ikki qator}

for k := 0 to 3 do{to'rt kvadrat bo'yicha}

begin

Rectangle((k+1)*x,(j+1)*y,(k+2)*x,(j+2)*y);

SetFillStyle(k+j*4,j+1);

Bar((k+1)*x+1,(j+1)*y+1,(k+2)*x-1,(j+2)*y-1)

end;

if ReadKey=#0 then k := ord(ReadKey);

CloseGraph;

end

end.

FloodFill procedurasi- ixtiyoriy yopiq figurani to'ldirishning joriy stilidan foydalangan holda to'ldiradi. Uning umumiy ko'rinishi quyidagicha:

Procedure FlodFill (X,Y: integer, border: Word);

Bu yerda X,Y- yopiq figura ichidagi ixtiyoriy nuqta koordinatalari; **Border**- chegaraviy chiziq rangi. Agar figura yopiq bo'lmasa, u holda to'ldirish butun ekran bo'yicha yoyiladi.

Masalan. Quyidagi dastur tasodifiy aylanalarni to'ldirishini namoyish qiladi.

Uses Graph, CRT;

var

d, r, ye, x, u, s : integer;

begin

{Grafikni inisializasiya kilish}

d := Detect; InitGraph(d, r, ' ');

e := GraphResult;

if e <> grOk then WriteLn(GraphErrorMsg(e)) else

begin

{to'g'ri burchakli oyna hosil qilamiz}

x := GetMaxX div 4;

u := GetMaxY div 4;

Rectangle(x,u,3*x,3*y);

SetViewPort(x+1,y+1, 3*x-1,3*y-1,ClipOn);

{kichik to'rtburchakni bo'yashni namoyish qiladi}

SetPillStyle(LtSlashFill,GetMaxColor);

Rectangle(0,0,8,20); FloodFill(1,1,GetMaxColor);

OutTextXY(10,25,'Press Enter...');

ReadLn; { Enter ni bosilishini kutadi}

{ixtiyoriy klavisha bosilmaguncha aylana hosil qilinaveradi}

repeat

{tasodifiy bo'yash shakli aniqlanadi}

SetFillStyle(Random(12),Random(GetMaxColor+1));

{aylana markazi koordinatalari va rangi beriladi}

x := Random (GetMaxX div 2);

u := Random (GetMaxY div 2);

s := Random (succ(GetMaxColor));

SetColor(c);

{aylana hositl qilinadi va bo'yaladi}

Circle(x, u, Random(GetMaxY div 5));

FloodFill (x, u, s)

until KeyPressed;

if ReadKey=#0 then x:= ord(ReadKey);

CloseGraph

end

end.

Bar procedurasi- ekranning to'g'ri burchakli sohasini bo'yaydi. Uning umumiy ko'rinishi:

Procedure Bar (x1, y2, x2, y2: integer); Bu yerda **x1, ..., u2** lar bo'yaladigan soha kordinatalari bo'lib, **(x1,u1)** yuqori chap burchak va **(x2,u2)** quyi o'ng burchak koordinatalari hisoblanadi. Procedura sohani **SetfillStyle** procedurasi orqali o'rnatilgan **uzor** va rang bilan bo'yaydi.

Masalan. Quyidagi dastur tasodifiy to'rtburchaklarni bo'yaydi.

```
Uses Graph, CRT;
var
  d, r, e : integer;
begin
  {Grafikni inisializasiya qilish}
  d := Detect; InitGraph(d, r, '');
  e := GraphResult; if e <> grOk then
    WriteLn(GraphErrorMsg(e)) else
begin
  {Ekran markazida oyna hosil qilinadi}
  d := GetMaxX div 4;
  r := GetMaxY div 4; Rectangle(d,r,3*d,3*r);
  SetViewPort(d+1,r+1,3*d-1,3*r-1,ClipOn);
  {tasodifiy ko'pburchaklarni hosil qilish va bo'yash}
  repeat
    SetFillStyle(Random(12),Random(succ(GetMaxColor)));
    Bar(Random(GetMaxX),Random(GetMaxY),
      Random(GetMaxX),Random(GetMaxY));
  until KeyPressed;
  if ReadKey=#0 then d := ord(ReadKey);
  CloseGraph ;
end
end.
```

BarZd procedurasi- parallelepipedning uch o'lchovli tasvirini chizadi va uning oldingi o'qini bo'yaydi. Uning umumiy ko'rinishi quyidagicha:

Procedure BarZd (x1, y2, x2, y2, Depth: integer; top: boolean); Bu yerda **x1, ..., u2** oldingi yoq koordinatalari bo'lib, **(x1,u1)** yuqori chap va **(x2, u2)** quyi o'ng burchak koordinatalari hisoblanadi. **Depth-** uch o'lchovli tasvirning uchunchi o'lchovi ("chuqurligi"); **Tor-** yuqori yoqning tasvirlash usuli. Agar **Tor** parametr **true** qiymatga yega bo'lsa, parallelepipedning yuqori yoqi chiziladi, aks holda- chizilmaydi. Bu parametrning qiymati sifatida **Graph** modulida aniqlangan quyidagi o'zgarmaslar ishlatiladi:

```
const
  TopOn = True;
  TopOff = False;
```

Masalan. Qo'yidagi **Var3d** procedurasidan foydalanishning bir nechta holatlarini ko'rsatadi.

```
Uses Graph,CRT;
```

```

var
  d, r, e: Integer;
begin
  {Grafikni inisializasiya kilish}
  d := Detect;
  InitGraph(d, r, ' ');
  e := GraphResult;
  if e <> grOk then
    WriteLn(GraphErrorMsg(e)) else
begin
  {yuqori qirrali ustun}
  Bar3D (80, 100, 120, 180, 15, TopOn);
  {yuqori qirrasiz ustun:}
  Vag3D (150, 150, 190, 180, 15, TopOff);
  {Ustun turadi:}
  Bar3D (230, 50, 250, 150, 15, TopOn);
  Bar3D (220, 150, 260, 180, 15, TopOn);
  {U etogo stolbika net verxney grani, i poetomu on ne meshayet
  postavlennomu na nego sverxu:}
  Bar3D (300, 150, 340, 180, 15, TopOff);
  SetLineStyle(3,0,1);
  SetColor(Yellow);
  SetFillStyle(LtSlashFill,Yellow);
  Bar3D (300, 50, 340, 150, 15, TopOn);
  if ReadKey=#0 then d := ord(ReadKey);
  CloseGraph;
end
end.

```

FillPoly procedurasi- bo'yalgan ko'pburchak chizadi. Uning umumiy ko'rinishi: **procedure Fillpoli (N:word; vat Cords);** N- yopiq ko'pburchak uchlarining soni; **Cords-Pointtype** tilidagi o'zgaruvchi bo'lib, uchlarining koordinatalaridan iborat bo'ladi. Uchlarining koordinatalari **Integer** turidagi qiymatlarining to'rtligi orqali beriladi: birinchisi gorizonta va ikkinchisi vertikal koordinatalarni aniqlaydi. Ular uchun modulda turni quyidagicha foydalanish mumkin:

```

type
  Pointtype= rekord
  X,y:integer; end;

```

Chiziqning shakli va rangi **SetLineStyle** va **SetColor** proceduralari orqali, bo'yash rangi va turi esa **SetFillStyle** procedurasi bilan beriladi.

Masalan: Quyidagi dasturlarda bo'yalgan ko'pburchaklar hosil qilinadi.

Uses Graph, CRT;

```

var
  d, r, e: Integer;
  p : array [1..6] of PointType; n, k : Word;
begin

```

```

    {Grafikni inisializasiya kilish}
    d := Detect; InitGraph(d, r, ' ');
    e := GraphResult; if e <> grOk then
    WriteLn(GraphErrorMsg(e)) else
begin
    {ekran markazida oyna}
    d := GetMaxX div 4;
    r := GetMaxY div 4;
    Rectangle(d,r,3*d,3*r);
    SetViewPort(d+1,r+1,3*d-1,3*r-1,ClipOn);
    {tasodifiy bo'yalgan ko'pburchaklarni chop etish}
    repeat
    {rang va uzorni tanlash}
    SetFillStyle(Random(12),Random(succ(GetMaxColor)));
    SetColor (Random(succ(GetMaxColor)));
    {tasodifiy koordinatalarni tanlash}
    n := Random (4) + 3 ; for k := 1 to n do with p[k] do
begin
    x := Random (GetMaxX div 2);
    u := Random (GetMaxY div 2)
end;
    FillPoly (n, p) {chop etish va bo'yash}
    until KeyPressed;
    if ReadKey=#0 then k := ord(ReadKey);
    CloseGraph
end
end.

```

FillEllipse procedurasi- bo'yalgan elementni hosil qiladi. Uning umumiy ko'rinishi quyidagicha:

Procedure Fillellipse (X,Y,RX,RY:integer); Bu yerda **X,U**- ellips markazi-ning koordinatalari; **RX, RY**- ellipsning **gorizontal** va **vertikal** radiuslari.

Sector procedurasi- ellips sektorini chizadi va uni bo'yaydi. Uning umumiy ko'rinishi:

Procedure Sector (X,Y: integer; BegA, endA, RX, RY: word);

Bu yerda **begA, endA**- mos ravishda ellips sektorining boshlang'ich va oxirgi burchaklari. Boshqa parametrlar **FillEllipse** procedurasi parametrlari kabi aniqlanadi.

Masalan. Quyidagi dastur ekranga bo'yalgan elips va sektorlarni hosil qiladi. Dasturdan chiqish uchun ixtiyoriy tugmacha bosiladi.

Uses Graph, CRT;

var

d, r, e : Integer;

begin

{Grafikni inisializasiya kilish}

d := Detect; InitGraph(d, r, "');

```

e := GraphResult; if e <> grOk then
  WriteLn(GraphErrorMsg(e)) else
begin
  {ekran markazida oyna hosil qilish}
  d := GetMaxX div 4;
  r := GetMaxY div 4;
  Rectangle(d,r,3*d,3*r);
  SetViewPort(d+1,r+1,3*d-1,3*r-1,ClipOn);
  {chiqarish sikli}
  repeat
    SetFillStyle(Random(12), Random(succ(GetMaxColor)));
    SetColor (Random(succ(GetMaxColor)));
    Sector(Random(GetMaxX div 2),Random(GetMaxY div 2),
    Random(360),Random(360),Random(GetMaxX div 5),
    Random(GetMaxY div 5));
    FillElipse (Random (GetMaxX div 2),
    Random(GetMaxY div 2),Random(GetMaxX div 5),
    Random(GetMaxY div 5))
  until KeyPressed;
  if ReadKey=#0 then d := ord(ReadKey);
  CloseGraph;

  end
end.

```

PieSlice procedurasi- aylana sektorini chizadi va bo'yaydi. Uning umu-miy ko'rinishi:

```

procedure Pieslice (x,y: integer; BegA, endA, R: word);

```

Uning **Sector** procedurasidan farqi, unda faqatgina bitta **R** radius ko'rsatiladi. Proceduradan aylanaviy diagrammalarni yasashda foydalanish ancha qulay hisoblanadi.

Masalan. Quyidagi dastur aylanaviy diagrammani hosil qiladi.

```

Uses Graph, CRT;
var
  d, r, e : Integer;
begin { grafik tartibni inisializasiya qilish}
  d := Detect;
  InitGraph(d, r, '');
  e := GraphResult; if e <> grOk then
    WriteLn(GraphErrorMsg(e)) else
begin
  {kichik sektorni chiqarish}
  SetFillStyle(WideDotFill, White);
  PieSlice(GetMaxX div 2+5,GetMaxY div 2+4,270,360,100);
  {katta sektorni chiqarish}
  SetFillStyle (SolidFill, Red);

```

```

    PieSlice (GetMaxX div 2,GetMaxY div 2, 0,270,100).;
    {yozuvni hosil qilish}
    OutTextXY (GetMaxX div 2+90,GetMaxY div 2+70, '25%');
    OutTextXY(GetMaxX div 2-50,GetMaxY div 2-20, '75%');
    {ixtiyoriy klavishani bosishi kutiladi}
    if ReadKey=#0 then d := ord(ReadKey);
    CloseGraph;
end
end.

```

Grafik holatida matnlarni chiqarish.

Quyida beriladigan funksiya va proceduralardan grafik holatida matn li ma'lumotlarni chiqarish uchun foydalaniladi.

Outtext procedurasi- ko'rsatkichning joriy o'rnidan boshlab, matnli satrni chiqaradi. Uning umumiy ko'rinishi: **procedure outtext (txt: String);** Bu yerda **txt**- chiqariladigan satr.

OuttextXY procedurasi- ko'rsatilagn joydan boshlab satr chiqariladi. Uning umumiy ko'rinishi:

```

Procedure OuttextXY (x, y: integer; txt: String);

```

Bu yerda **x,u**: matn chiqarish boshlanadigan nuqta koordinatalari; **txt**- chiqariladigan satr.

SettextStyle procedurasi- grafik ekranda matn chiqarish shartini o'rnatadi. Uning umumiy ko'rinishi.

```

Procedure Settextstyle (font, Direct, Seze: word);

```

Bu yerda **font**- shrift kodi; **Direct**- yo'nalish kodi, **Seze**- shrift o'lchovi kodi. Shrift o'lchovi kodini ko'rsatish uchun quyidagi o'zgarmaslardan foydalaniladi:

```

Const

```

```

    Defaultfont= 0; {8x8 nuqtali shrift}

```

```

    Triplexfont= 1; {utroyenniy shrift Trip. chr}

```

```

    Smallfont= 2; {umen'shenniy Litt. chr}

```

```

    SansSeriffont= 3; {premoy shrift Sans. chr}

```

```

    Gothicfont= 4; {goticheskiy shrift Goth. chr}

```

Matnni chiqarish yo'nalishini berish uchun quyidagi o'zgarmaslar-dan foydalaniladi:

```

    Sonst

```

```

    Horizdir= 0; {chapdan o'ngga}

```

```

    Vertdir= 1; {pastdan yuqoriga}

```

Har bir shriftni o'n martalab o'lchovini o'zgartirish mumkin. Chiqari-layotgan belgilarning o'lchovi **Size** parametri bilan kodlanadi. Kod **1** dan **10** gacha bo'lgan qiymatlarni qabul qilishi mumkin.

Masalan. Quyidagi dastur turli shriftlarni namoyish qiladi.

```

Uses Graph, CRT;

```

```

const

```

```

FontNames: array [1..10] of String[4] =

```

```

    ('TRIP', 'LITT' SANS ', ' GOTH ', 'SCRI ', ' SIMP ', 'TSCR ',
    ' LOOM ', ' EURO', ' BOLD ');

```

```

    Tab1 = 50;
    Tab2 = 150;
    Tab3 = 220;
var
    d, r, Err, {grafikni inisializatsii qilish o'zgaruvchilari}
    Y, dY, {chiqarish ordinatasi va uning orttirmasi}
    Size, {belgining o'lchovi}
    MaxFont, {shriftning maksimal o'lchovi}
    k: Integer; {shrift nomeri}
    NT, SizeT, SynibT: String; {chiqarish satri}
    c: Char;
    {-----}
Procedure OutTextWithTab ( S1, S2, S3, S4: String);
{ Tab1..TabZ larni hisobga olgan holda S1,S2,S3,S4 satrlarni
  chiqarish}
begin
    MoveTo( (Tab1-TextWidth(S1) ) div 2, Y);
    OutText (S1) ;
    MoveTo(Tab1+(Tab2-Tab1-TextWidth(S2)) div 2, Y);
    OutText (S2) ;
    MoveTo(Tab2+(Tab3-Tab2-TextWidth(S3)) div 2, Y);
    OutText(S3);
    if S4='Symbols' then { Symbols kolonka sarlavhasi}
    MoveTo((Tab3+GetMaxX-TextWidth(S4)) div 2, Y)
    else {boshqa satrlar}
    MoveTo(Tab3+3, Y);
    OutText(S4)
end;
    {-----}
begin
    {Grafikni inisializatsiya qilish}
    InitGraph(d, r, ' ');
    Err := GraphResult; if Err < 0 then
    WriteLn(GraphErrorMsg(Err))
    else
begin
    { shriftlar sonini aniqlash}
    {$IFDEF VER70}
    MaxFont := 10; .
    {$ELSE}
    MaxFont := 4;
    {$ENDIF}
    SetTextStyle(l, 0, 4);
    Y := 0;
    OutTextWithTab('N', 'Name', Size, 'Symbols');

```

```

{chiziq sarlavhasi Yning balandligi aniklanadi}
Y := 4*TextHeight('Z') div 3;
Line(0,Y,GetMaxX,Y) ;
{Opredelyayem nachalo Y tablisy i vysotu dY kajdoy stroki}
Y := 3*TextHeight('Z') div 2;
dY := (GetMaxY-Y) div (MaxFont);
    {belgilar satrini tayyorlash}
    SymbT := '';
For s := 'a' to 'z' do
    SymbT := SymbT+c;
    {satrlar jadvalini chikarish}
    for k := 1 to MaxFont do
begin
    Size := 0;
    {dY ga teng bo'lguncha satr balandligini o'zgartirish}
    repeat
        inc(Size);
        SetTextStyle(k,0,Size+1);
        until (TextHeight('Z')>=dY) or (Size=10)
            or (Textwidth(FontNames[k])>(Tab2-Tab1));
        {Gotovim nomer NT i razmer SizeT shrifta}
        Str(k,NT);
        Str(Size,SizeT);
        {satr jadvalini chiqarish}
        SetTextStyle(k,HorizDir,Size);
        OutTextWithTab(NT,FontNames[k],SizeT,SymbT);
        inc(Y,dY)
    end;
    {ramka chizig'i chiziladi}
    Rectangle(0,0,GetMaxX,GetMaxY);
    Line(Tab1,0,Tab1,GetMaxY);
    Line(Tab2,0,Tab2,GetMaxY);
    Line(Tab3,0,TaZ,GetMaxY);
    {ixtiyoriy tugmachani bosish}
    ReadLn;
    CloseGraph;
end
end.

```

Settextjustife procedurasi- ko'rsatkichning joriy holatiga nisbatan chiqarilayotgan matnlar tekislanadi. Uning umumiy ko'rinishi:

Prosedure Settextjustife (Horiz, Vert: word);

Bu yerda **Horiz-** gorizontal tekislash; **Vert-** vertikal tekislash. Tekislash matn qanday joylashishini aniqlaydi, ya'ni ko'rsatilgan joydan **chapga** yoki **o'ngga**, **yuqoriga**, **quyiga** yoki **markaz** bo'yicha. Bu yerda quyidagi o'zgarishlar ishlatiladi:

Const

Lefttext= 0; {ko'rsatkich matndan chapga}
Contertext= 1; {Silletrik chap va o'ngga, yuqori va pastga}
Righttext= 2; {Ko'rsatkich matndan o'ngga}
Bottomtext= 3; {Ko'rsatkich matndan pastga}
Toptext= 2; {Ko'rsatkich matndan yuqoriga}

Masalan: Quyidagi dastur grafik ekran markaziga nisbatan turli tekislash usullarini namoyish qiladi.

Textwidth funksiyasi- chiqarilayotgan matnli satrning uzunligini aniqlaydi. Uning umumiy ko'rinishi quyidagicha:

Function textwidth (txjt: String): word;

Text Height funksiyasi- shrift balandligini aniqlaydi. Uning umumiy ko'rinishi:

Function TextHeight (txt: String): word;

Sinov savollari.

1. Grafik holati nima? Unga qanday o'tiladi?
2. Grafik holatida ekranning holati qanday bo'ladi?
3. Grafik holatdan qanday chiqiladi?
4. GRAPH modulning asosiy funksiya proceduralari qanday?
5. Figuralarni hosil qilish procedura va funksiyalarining sanang?
6. Grafik holatda matnlar bilan ishlash jarayonini ta'riflang?
7. Funksiyalar grafigi qanday hosil qilinadi?
8. Ob'ektlar qanday harakatlantiriladi?

Uyga vazifa : *Turbo-Paskal tizimida grafik holatida ishlash qoidalarini mukammal o'rganing, tasvirlarni va grafiklarni hosil qiling, a ularni amaloyotga tatbiq eting.*

Adabiyotlar

1. Shafrin Yu. Osnovi kompyuternoy texnologii. Uchebnoi posobie- M: 1997, - 560 b.
2. Raxmanqulova S.I. IBM PC shaxsiy kompyuterda ishlash.- T:, NMK "Sharq"- INSTAR, 1996 y.
3. Figurnov V. E. IBM PC dlya pozovatelya. - M; Finantsi i statistika, 1990 g.
4. Ortiqov A., Mamatqulov A. IBM PC kompyuteridan foydalanish.- Toshkent, "Qomus", 1992 y.
5. Volvachev A.K., Krisevich V. S. Programmirovaniye na yazika Paskal dlya personalnix EC EBM- Minsk, Visshaya shkola, 1989 g.
6. Vasyukova N.D., Tyulyaeva V.V. Praktikum po osnovam programmirovaniya. Yazik Paskal- M; Visshsya shkola, 1991 g.
7. Zuev E.A. Yazik programmiravaniya Turbo Paskal 6.0, 7.0- M; Radioi svyaz, 1993.
8. Faysman A. Professeonalnoe programmirovaniya na yazike Paskal- M; Nauka, 1989 g.
9. Aminov I. Paskal dasturlash tili. O'quv qullanma- SamDU, Samarqand, 1996 y.

10. Aminov I.B. Informatika va informatsion texnologiyalar. Ma'ruzalar matni. 2005 yil. SamDU.
11. Aminov I.B., Eshtemirov S., Nomozov F. Informatika va informatsion texnologiyalar fanidan laboratoriya ishlari. Uslubiy qo'llanma. Samarqand, SamDU, 2008 yil.

Adabiyotlar

1. Shafrin Yu. Osnovi kompyuternoy texnologii. Uchebnoi posobie- M: 1997, - 560 b.
2. Raxmanqulova S.I. IBM PC shaxsiy kompyuterda ishlash.- T:, NMK "Sharq"- INSTAR, 1996 y.
3. Figurnov V. E. IBM PC dlya pozovatelya. - M; Finantsi i statistika, 1990 g.
4. Ortiqov A., Mamatqulov A. IBM PC kompyuteridan foydalanish.- Toshkent, "Qonis" 1992 y.
5. Volvachev A.K., Krisevich V. S. Programmirovaniye na yazika Paskal dlya personalnix EC EBM- Minsk, Visshaya shkola, 1989 g.
6. Vasyukova N.D., Tyulyaeva V.V. Praktikum po osnovam programmirovaniya. Yazik Paskal- M; Visshaya shkola, 1991 g.
7. Zuev E.A. Yazik programmiravaniya Turbo Paskal 6.0, 7.0- M; Radioi svyaz, 1993.
8. Faysman A. Professeionalnoe programmirovaniya na yazike Paskal- M; Nauka, 1989 g.
9. Aminov I. Paskal dasturlash tili. O'quv qullanma- SamDU, Samarqand, 1996 y.
10. Aminov I.B. Informatika va informatsion texnologiyalar. Ma'ruzalar matni. 2005 yil. SamDU.
11. Aminov I.B. S.Eshtemirov, F.Nomozov. Informatika va informatsion texnologiyalar fanidan laboratoriya ishlari. Uslubiy qo'llanma. SamDU, Samarqand, 2008 yil.