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

GULISTON DAVLAT UNIVERSITETI

AXBOROT TEXNOLOGIYALARI KAFEDRASI

DASTURLASH TILLARI

fanidan
oquv uslubiy majmua

Bilim sohasi: **100000 - Gumanitar**

Ta’lim sohasi: **110000 - Pedagogika**

Ta’lim yo’nalishi: **5110700 - Informatika o‘qitish metodikasi**

Guliston – 2020

O’quv–uslubiy majmua Oliy va o’rta maxsus ta’lim vazirligi tomonidan 2018 yil 27.03. dagi 274 - sonli buyrug’ining 2 - ilovasi bilan tasdiqlangan Dasturlash tillari fani dasturi (№ BD -5112100 2.02 2018 – yil 03.03) talablari asosida tayyorlangan.

Tuzuvchi: D.B.Abdurahimov - GulDU “Axborot texnologiyalari” kafedrasi dotsenti, pedagogika fanlari nomzodi

Taqrizchi: D.E.Toshtemirov - GulDU “Axborot texnologiyalari” kafedrasi dotsenti, pedagogika fanlari nomzodi

Ushbu o’quv–uslubiy majmua Guliston davlat universiteti O’quv – metodik Kengashi tomonidan (29.08.2020 y. dagi, № 1 - sonli bayonnoma) nashrga tavsiya etilgan.

Mundarija

Bet

I. Kirish

O‘quv fanining dolzarbligi va oliy kasbiy ta’limdagi o‘rni

Mustaqil Resrublikamizda yuz berayotgan siyosiy, iqtisodiy, ilmiy-texnikaviy va madaniy o‘zgarishlar Oliy ta’lim tizimida ham o‘z aksini topmoqda. O‘zbekistonda uzluksiz ta’lim-tarbiya tizimini yaratish, shu asosida ta’lim sifatini jahon andozalari darajasiga etkazish ta’lim sistemasining eng dolzarb vazifasiga aylandi. Bu esa barcha mutaxassisliklar qatori Informatika va dasturlash bo‘yicha kadrlar tayyorlash sifatini oshirishni ham taqozo etadi. Bu maqsad vazifalar ushbu fan dasturi mazmunini ham belgilaydi. Algoritm konsepsiyasining vujudga kelishi bilan algebra, sonlar nazariyasi, geometriya va matematikaning boshqa sohalariga tegishli bir qator muammolarning echimli yoki echimli emasligini aniqlashtirish imkonini berdi. Algoritmlar nazariyasi faoliyat sohasi EHMlar vujudga kelishi bilan yanada kengaydi. Yuqoridagi fikrlar “Dasturlash tillari” fanining asosiy mazmunini belgilashga yordam beradi.

“Dasturlash tillari” fani umumkasbiy fanlar blokiga kiritilgan kurs hisoblanib, 2- va 3-kurslarda o‘qitilishi maqsadga muvofiq. “Dasturlash tillari” fani “Informatika o’qitish metodikasi” ta’lim yo‘nalishida o‘qitiladi. Mazkur fan Algoritmlar fanining nazariy va uslubiy asosini tashkil qilib, o‘z rivojida aniq va tabiiy fanlar uchun zamin bo‘lib xizmat qiladi.

O‘quv fanining maqsadi va vazifasi

“Dasturlash tillari” fanini o‘qitishdan maqsad – talabalarga dasturlashning ilmiy-nazariy asoslarini, informatika o‘qituvchisining kasbiy sohasida egallashi lozim bo‘lgan bilimlar, amalda qo‘llash uchun ko‘nikma va makalalarni shaklantirish hamda rivojlantirishdan iborat.

Ushbu maqsadga erishish uchun fan talabalarni ob’ektga yo‘naltirilgan dasturlash tillarida ishlash, amaliy masalalarga dasturlar tuzishga oid nazariy bilimlar, amaliy ko‘nikma va malakalarini shakllantirish vazifalarini bajaradi.

Fan bo‘yicha talabalarning bilim, ko‘nikma va malakalariga quyidagi talablar qo‘yiladi. ***Talaba:***

–ob‘yektga yo‘naltirilgan dasturlash tillarining nazariy asoslari, ob‘yektlarni loyihalash, matematik va interfeys ob‘yektlari, voqealar va xabarlar, ob’yektga yo‘naltirilgan muhitlarda xabarlarni uzatish, ularga ishlov berish mexanizmlari, ob’yektlar iyerarxiyasi asosida dasturlarni loyihalash, muayyan ob’yektga yo‘naltirilgan dasturlash tillari to‘g‘risida ***tasavvurga ega bo‘lishi;***

–ob’yektga yo‘naltirilgan dasturlash tillarida chiziqli, tarmoqlanuvchi va takrorlanuvchi va modulli dasturlar tuza olishni, dasturlashning ob’yektga yo‘naltirilgan paradigmasini, ob‘yektga yo‘naltirilgan muhitlarda dasturlarni loyihalashni ***bilishi va ulardan foydalana olishi;***

–ob’yektga yo‘naltirilgan dasturlash tillari muhitida ishlash, masalalarni tahlil qila olish, muayyan dasturlash tillari yordamida masalalarning dasturini tuzish va natijalarni taqqoslay olish ***ko‘nikmalariga ega bo‘lishi lozim.***

2. MA’RUZA MASHG’ULOTLARI

1-MAVZU: C++ TILINING LEKSIK ASOSLARI

Reja:

1. C++ dasturlash tilining kelib chiqishi.
2. C++ dasturlash tili alifbosi.
3. C++ dasturlash tilida dastur tuzilishi.
4. C++ tilida o_zgaruvchilar va o_zgarmaslar.
5. C++ tilida ma‘lumot toifalari.
6. C++ tilida arifmetik operatorlar.

***Dars maqsadi*:** Talabalarda C++ dasturlash tilining leksik asoslaridan foydalanish ko’nikmalarini hosil qilish.

Tayanch iboralar: Dastur, dastur kodi, strukturaviy dasturlash, ob’ekt, ob’ektga mo’ljallangan dasturlash, kompilyatsiya, inkapsulyatsiya, merosxo’rlik, polimorfizm.

C++ dasturlash tili C tiliga asoslangan. C esa o'z navbatida B va BCPL tillaridan kelib chiqqan. BCPL 1967 yilda Martin Richards tomonidan tuzilgan va operatsion sistemalarni yozish uchun mo'ljallangan edi. Ken Thompson o'zining B tilida BCPL ning ko'p hossalarini kiritgan va B da UNIX operatsion sistemasining birinchi versiyalarini yozgan. BCPL ham, B ham tipsiz til bo'lgan. Yani o'garuvchilarning ma'lum bir tipi bo'lmagan - har bir o'zgaruvchi kompyuter hotirasida faqat bir bayt yer egallagan. O'zgaruvchini qanday sifatda ishlatish esa, yani butun sonmi, kasrli sonmi yoki harfdekmi, dasturchi vazifasi bo'lgan.

C tilini Dennis Ritchie B dan keltirib chiqardi va uni 1972 yili ilk bor Bell Laboratoriyasida, DEC PDP-11 kompyuterida qo'lladi. C o'zidan oldingi B va BCPL tillarining juda ko'p muhim tomonlarini o'z ichiga olish bilan bir qatorda o'zgaruvchilarni tiplashtirdi va bir qator boshqa yangiliklarni kiritdi. Boshlanishda C asosan UNIX sistemalarida keng tarqaldi. Hozirda operatsion sistemalarning asosiy qismi C/C++ da yozilmoqda. C mashina arhitekturasiga bog'langan tildir. Lekin yahshi rejalashtirish orqali dasturlarni turli kompyuter platformalarida ishlaydigan qilsa bo'ladi. 1983 yilda, C tili keng tarqalganligi sababli, uni standartlash harakati boshlandi. Buning uchun Amerika Milliy Standartlar Komiteti (ANSI) qoshida X3J11 tehnik komitet tuzildi. Va 1989 yilda ushbu standart qabul qilindi. Standartni dunyo bo'yicha keng tarqatish maqsadida 1990 yilda ANSI va Dunyo Standartlar Tashkiloti (ISO) hamkorlikda C ning ANSI/ISO 9899:1990 standartini qabul qilishdi. Shu sababli C da yozilgan dasturlar kam miqdordagi o'zgarishlar yoki umuman o'zgarishlarsiz juda ko'p kompyuter platformalarida ishlaydi.

C++ 1980 yillar boshida Bjarne Stroustrup tomonidan C ga asoslangan tarzda tuzildi. C++ juda ko'p qo'shimchalarni o'z ichiga olgan, lekin eng asosiysi u ob'ektlar bilan dasturlashga imkon beradi. Dasturlarni tez va sifatli yozish hozirgi kunda katta ahamiyat kasb etmoda. Buni ta'minlash uchun ob'ektli dasturlash g'oyasi ilgari surildi. Huddi 70-chi yillar boshida strukturali dasturlash kabi, dasturlarni hayotdagi jismlarni modellashtiruvchi ob'ektlat orqali tuzish dasturlash sohasida inqilob qildi. C++ dan tashqari boshqa ko'p ob'ektli dasturlshga yo'naltirilgan tillar paydo bo'ldi. Shulardan eng ko'zga tashlanadigani Xerox ning Palo Altoda joylashgan ilmiy-qidiruv markazida (PARC) tuzilgan Smalltalk dasturlash tilidir. Smalltalk da hamma narsa ob'ektlarga asoslangan. C++ esa gibrid tildir. Unda C ga o'hshab strukturali dasturlash yoki yangicha, ob'ektlar bilan dasturlash mumkin. Yangicha deyishimiz ham nisbiydir. Ob'ektli dasturlash falsafasi paydo bo'lganiga ham yigirma yildan oshayapti. C++ funksiya va ob'ektlarning juda boy kutubxonasiga ega. Yani C++ da dasturlashni o'rganish ikki qismga bo'linadi. Birinchisi bu C++ ni o'zini o'rganish, ikkinchisi esa C++ ning standart kutubxonasidagi tayyor ob'ekt/funksiyalarni qo'llashni o'rganishdir.

Alfavit, identifikator, xizmatchi so‘zlar.

Alfavit. C++ alfavitiga quyidagi simvollar kiradi.

- Katta va kichik lotin alfaviti xarflari (A,B,..,Z,a,b,…,z)
- Raqamlar: 0,1,2,3,4,5,6,7,8,9
- Maxsus simvollar: ― , {} | [] () + - / % \ ; _ . : ? < = > _ ! & * # ~ ^
- Ko‘rinmaydigan simvollar (―umumlashgan bushliq simvollari‖). Leksemalarni uzaro ajratish uchun ishlatiladigan simvollar (misol uchun bushlik, tabulyatsiya, yangi qatorga o‘tish belgilari).

Izohlarda, satrlarda va simvolli konstantalarda boshqa literalar, masalan rus xarflarini ishlatilishi mumkin.

C++ tilida olti khil turdagi leksemalar ishlatiladi: ehrkin tanlanadigan va ishlatiladigan identifikatorlar, khizmatchi suzlar, konstantalar(konstanta satrlar), amallar(amallar belgilari), azhratuvchi belgilar.

Identifikator.Identifikatorlar lotin xarflari,ostki chiziq belgisi va sonlar ketma ketligidan iborat bo‘ladi. Identifikator lotin xarfidan yoki ostki chizish belgisidan boshlanishi lozim. Misol uchun:

A1, _MAX, adress_01, RIM, rim

Katta va kichik xarflar farklanadi, shuning uchun ohirgi ikki identifikator bir biridan farq

qiladi.

Borland kompilyatorlaridan foydalanilganda nomning birinchi 32 xarfi ,ba‘zi kompilyatorlarda 8 ta xarfi inobatga olinadi. Bu holda NUMBER_OF_TEST va NUMBER_OF_ROOM identifikatorlari bir biridan farq qilmaydi.

Xizmatchi so‘zlar. Tilda ishlatiluvchi ya‘ni dasturchi tomonidan uzgaruvchilar nomlari sifatida ishlatish mumkin bulmagan identifikatorlar xizmatchi so‘zlar deyiladi. C ++ tilida quyidagi xizmachi so‘zlar mavjud:

int extern else char register for float typedef do
double static while struct goto switch union return
case long sizeof default short break entry unsigned
continue auto if

Dastur tuzilishi

Sodda dastur tuzilishi. Dastur preprocessor komandalari va bir necha funksiyalardan iborat bo‘lishi mumkin. Bu funksiyalar orasida main nomli asosiy funksiya bo‘lishi shart. Agar asosiy funksiyadan boshqa funksiyalar ishlatilmasa dastur quyidagi ko‘rinishda tuziladi:

```
Preprocessor_komandalari
Void main()
{  Dastur tanasi.  }
```

Preprocessor direktivalari kompilyatsiya jarayonidan oldin preprocessor tomonidan bajariladi. Natijada dastur matni preprocessor direktivalari asosida o‘zgartiriladi. Preprocessor komandalaridan ikkitasini ko‘rib chiqamiz.

include <fayl_nomi> Bu direktiva standart bibliotekalardagi funksiyalarni dasturga joyjlash uchun foydalaniladi.

#define <almashtiruvchi ifoda> <almashinuvchi ifoda>

Bu direktiva bajarilganda dastur matnidagi almashtiruvchi ifodalar almashinuvchi ifodalarga almashtiriladi.

Misol tariqasida C ++ tilida tuzilgan birinchi dasturni keltiramiz:

```
#include <iostream.h>
void main()
{
   Cout << —\n Salom, Dunyo!
\n‖; }
```

Bu dastur ehkranga Salom, Dunyo! Jumlasini chiqaradi.

Define direktivasi yordamida bu dasturni quyidagicha yozish mumkin:

```
#include <iostream.h>
#define pr      Cout << —\n Salom, Dunyo! \n‖
#define  begin
{ #define end   }
void main()
    begin
pr;   end
```

Define direktivasidan nomlangan konstantalar kiritish uchun foydalanish mumkindir.

Misol uchun:

```
#define EULER 2.718282
```

Agar dasturda quyidagi matn mavjud bo‘lsin:
Double mix=EULER
D=alfa*EULER

Preprocessor bu matnda har bir EULER konstantani uning qiymati bilan almashtiradi, va natijada quyidagi matn hosil bo‘ladi. Double mix=2.718282
D=alfa*2.718282

Dastur matni va preprocessor. C ++ tilida matnli fayl shaklida tayyorlangan dastur uchta qayta ishlash bosqichlaridan o‘tadi.

Matnni preprocessor direktivalari asosida o‘zgartilishi. Bu jarayon natijasi Yana matnli fayl bo‘lib preprocessor tomonidan bajariladi.

Kompilyatsiya. Bu jarayon natijasi mashina kodiga o‘tkazilgan obektli fayl bo‘lib, kompilyator tomonidan bajariladi.

Bog‘lash. Bu jarayon natijasi to‘la mashina kodiga o‘tkazilgan bajariluvchi fayl bo‘lib, boglagich(komponovthik) tomonidan bajariladi. Preprocessor vazifasi dastur matnini preprocessor direktivalari asosida o‘zgartirishdir. Define direktivasi dasturda bir jumlani ikkinchi jumla bilan almashtirish uchun ishlatiladi. Bu direktivadan foydalanishning sodda misollarini biz yuqorida ko‘rib chiqdik. Include direktivasi ikki ko‘rinishda ishlatilishi mumkin.

#include fayl nomi direktivasi dasturning shu direktiva urniga qaysi matnli fayllarni qo‘shish kerakligini ko‘rsatadi.

#include <fayl nomi> direktivasi dasturga kompilyator standart bibliotekalariga mos keluvchi sarlavhali fayllar matnlarini qushish uchun muljhallangandir. Bu fayllarda funksiya prototipi, tiplar, o‘zgaruvchilar, konstantalar ta‘riflari yozilgan bo‘ladi. Funksiya prototipi funksiya qaytaruvchi tip, funksiya nomi va funksiyaga uzatiluvchi tiplardan iborat bo‘ladi. Misol uchun cos funkciyasi prototipi quyidagicha yozilishi mumkin: double cos(double). Agar funkciya nomidan oldin void tipi ko‘rsatilgan bo‘lsa bu funksiya hech qanday qiymat qaytarmasligini ko‘rsatadi.Shuni ta‘kidlash lozimki bu direktiva dasturga standart biblioteka qo‘shilishiga olib kelmayjdi. Standart funksiyalarning kodlari bog‘lash ya‘ni aloqalarni tahrirlash bosqichida, kompilyatsiya bosqichidan so‘ng amalga oshiriladi.

Kompilyatsiya bosqichida sintaksis hatolar tekshiriladi va dasturda bunday hatolar mavjud bo‘lmasa, standart funksiyalar kodlarisiz mashina kodiga utkaziladi.

Sarlavhali fayllarni dasturning ihtiyoriy joyida ulash mumkin bo‘lsa ham, bu fayllar odatda dastur boshida qo‘shish lozimdir. Shuning uchun bu fayllarga sarlavhali fayl (header file) nomi berilgandir.

Dasturda kiritish va chiqarish funksiyalaridan masalan Cout<< funksiyasidan foydalanish uchun #include <iostream.h> direktivasidan foydalanish lozimdir Bu direktivada iostream.h sarlavhali fayl nomi quyidagilarni bildiradi: st- standart(standartnij), i- input(vvod), o- output(vihvod), h – head(sarlavha).

O‘zgaruvchilar. (VARIABLES)

O‘zgaruvchilar ob‘ekt sifatida. Ci++ tilining asosiy tushunchalaridan biri nomlangan hotira qismi – ob‘ekt tushunchasidir. Ob‘ektning xususiy holi bu

o‘zgaruvchidir. O‘zgaruvchiga qiymat berilganda unga ajratilgan hotira qismiga shu qiymat kodi yoziladi. O‘zgaruvchi qiymatiga nomi orqali murojaat qilish mumkin, hotira qismiga esa faqat adresi orqali murojaat qilinadi. O‘zgaruvchi nomi bu erkin kiritiladigan identifikatordor. O‘zgaruvchi nomi sifatida xizmatchi so‘zlarni ishlatish mumkin emas.

O‘zgaruvchilar tiplari. O‘zgaruvchilarning quyidagi tiplari mavjuddir: char – bitta simvol; long char – uzun simvol; int – butun son; short yoki short int – qisqa butun son; long yoki long int – uzun butun son;

float - haqiqiy son; long float yoki double – ikkilangan haqiqiy son; long double – uzun ikkilangan haqiqiy son;

Butun sonlar ta‘riflanganda ko‘rilgan tiplar oldiga unsigned (ishorasiz) ta‘rifi kushilishi mumkin. Bu ta‘rif qushilgan butun sonlar ustida amallar mod 2n arifmetikasiga asoslangandir . Bu erda n soni int tipi hotirada egallovchi razryadlar sonidir. Agar ishorasiz k soni uzunligi int soni razryadlar sonidan uzun bulsa, bu son qiyjmati k mod 2n ga teng bo'ladi. Ishorasiz k son uchun ga –k amali 2n – k formula asosida hisoblanadi. Ishorali ya‘ni signed tipidagi sonlarning eng katta razryadi son ishorasini ko‘rsatish uchun ishlatilsa unsigned (ishorasiz) tipdagi sonlarda bu razryad sonni tasvirlash uchun ishlatiladi.

O‘zgaruvchilarni dasturning ihtiyoriy qismida ta‘riflash yoki qayta ta‘riflash mumkin. Misol uchun:

```
Int a, b1, ac; eki
Int a;  int
b1; int ac;
```

O‘zgaruvchilar ta‘riflanganda ularning qiymatlari aniqlanmagan bo‘ladi. Lekin o‘zgaruvchilarni ta‘riflashda initsializatsiya ya‘ni boshlang‘ich qiyjmatlarini ko‘rsatish mumkin. Misol uchun: Int I=0;

```
Char c=‘k‘;
```

Typedef ta‘riflovchisi yangi tiplarni kiritishga imkon beradi. Misol uchun yangi COD tipini kiritish:

```
Typedef unsigned char COD;
COD simbol;
```

KONSTANTALAR. (CONSTANTS)

Konstanta bu o‘zgartirish mumkin bulmagan qiymatdir. C++ tilida besh turdagi konstantalar ishlatilishi mumkin: butun sonlar, haqiqiy sonlar, simvollar, sanovchi konstantalar va nul kursatkich.

Ma‘lumotlarning butun son turi.

Butun sonlar o‘nlik, sakkizlik yoki un oltilik sanoq sistemalarida berilishi mumkin. O‘nlik sanoq sistemasida butun sonlar 0-9 raqamlari ketma ketligidan iborat bo‘lib, birinchi raqami 0 bulishi kerak emas. Sakkizlik sanoq sistemasida butun sonlar 0 bilan boshlanuvchi 0-7 raqamlaridan iborat ketma ketlikdir. O‘n

oltilik sanoq sistemasida butun son 0x eki 0X bilan boshlanuvchi 0-9 raqamlari va a-f yoki A-F xarflaridan iborat ketma ketlikdir. Masalan 15 va 22 o‘nlik sonlari sakkizlikda 017 va 026, un oltilikda 0xF va 0x16 shaklda tasvirlanadi.

Ma‘lumolarning uzun butun son turi.

Oxiriga l eki L harflari quyilgan o‘nlik,sakkizlik yoki o‘n oltilik butun son.

Ma‘lumotlarning ishorasiz (unsigned) butun son turi: Ohiriga u yoki U harflari quyilgan o‘nlik,sakkizlik yoki o‘n oltilik oddiy yoki uzun butun son.

Ma‘lumotlarning haqiqiy son turi:

Olti qismdan iborat bulishi mumkin: butun qism, nuqta, kasr qism, yoki E belgisi, o‘nlik daraja , F eki f suffikslari.

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

Ma‘lumolarning uzun haqiqiy son turi : Ohiriga L eki l suffikslari quyjilgan haqiqiy son. Masalan: 2E+6L;

Simvolli konstanta.

Bittalik qavslarga olingan bitta yoki ikkita simvol. Misol uchun ‗x‘,‘*‘,‘\012‘,‘\0‘,‘\n‘- bitta simvolli konstanta; ‗dd‘,‘\n\t‘,‘\x07\x07‘ ikki simvolli konstantalar. ‗\‘ simvolidan boshlangan simvollar eskeyp simvollar deyjiladi.Simvolli konstanta qiymati simvolning kompyuterda qabul qilingan sonli kodiga tengdir.

Satrli konstanta.

Satrli konstantalar C++ tili konstantalariga kirmaydi, balki leksemalari alohida tipi hisoblanadi. Shuning uchun adabiyotda satrli konstantalar satrli leksemalar deb ham ataladi. Satrli konstanta bu ikkilik qavslarga olingan ihtiyoriy simvollar ketma ketligidir. Misol uchun ― Men satrli konstantaman‖. Satrlar orasiga eskeyp simvollar ham kirishi mumkin. Bu simvollar oldiga \ belgisi quyiladi. Misol uchun :

―\n Bu satr \n uch katorga \n zhoyjlashadi‖.

Satr simvollari hotirada ketma-ket joylashtiriladi va har bir satrli konstanta ohiriga avtomatik ravishda kompilyator tomonidan ‗\0‘ simvoli qo‘shiladi. Shunday satrning hotiradagi hazhmi simvollar soni+1 baytga tengdir. Ketma-ket kelgan va bushlik, tabulyatsiya yoki satr ohiri belgisi bilan ajratilgan satrlar kompilyatsiya davrida bitta satrga aylantiriladi. Misol uchun:

―Salom‖ ―Toshkent ‖ satrlari bitta satr deb qaraladi.

―Salom Toshkent‖

Bu qoidaga bir necha qatorga yozilgan satrlar ham buysinadi. Misol uchun :

―O‘zbekistonga ‖

―bahor ‖

―keldi‖ qatorlari

bitta qatorga mos:

―O‘zbekistonga bahor keldi‖

Agar satrda ‗\‘ belgisi uchrasa va bu belgidan so‘ng to ‗\n‘ satr ohiri belgisigacha bushlik belgisi kelsa bu bushlik belgilari ‗\‘ va ‘\n‘ belgisi bilan birga satrdan uchiriladi. Satrning uzi keyingi satrda kelgan satr bilan qo‘shiladi.

―Ozbekistonga \

— bahor\
— keldi‖ qatorlari
bitta qatorga mos:
—Uzbekistonga bakhor keldi‖

Sanovchi konstanta.

Sanovchi konstantalar enum hizmatchi so‘zi yordamida kiritilib, int tipidagi sonlarga qulay suzlarni mos quyish uchun ishlatiladi.

Misol uchun:

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

Agar son qiymatlari ko‘rsatilmagan bulsa eng chapki so‘zga 0 qiymati berilib qolganlariga tartib buyicha usuvchi sonlar mos quyiladi:

Enum{zero,one,two};

Bu misolda avtomatik ravishda konstantalar quyidagi qiymatlarni qabul qiladi:

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

Konstantalar aralash ko‘rinishda kiritilishi ham mumkin:

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

Bu misolda avtomatik ravishda konstantalar quyidagi qiyjmatlarni qabul qiladi: Zero=0, one=1, for=4;five=5,seeks=6; Yana bir misol:

Enum BOOLEAN {NO, YES};

Konstantalar qiymatlari:

NO=0, YES=1;

Nomlangan konstantalar.

C++ tilida o‘zgaruvchilardan tashqari nomlangan konstantalar kiritilishi mumkin. Bu konstantalar qiymatlarini dasturda o‘zgartirish mumkin ehmas. Konstantalar nomlari dasturchi tomonidan kiritilgan va hizmatchi so‘zlardan farqli bo‘lgan identifikatorlar bulishi mumkin. Odatda nom sifatida katta lotin harflari va ostiga chizish belgilari kombinaciyasidan iborat identifikatorlar ishlatiladi. Nomlangan konstantalar quyidagi shaklda kiritiladi:

Const tip konstanta_nomi=konstanta_qyjmati.

Misol uchun:

Const double EULER=2.718282;

Const long M=99999999;

Const R=765;

Ohirgi misolda konstanta tipi kursatilmagan, bu konstanta int tipiga tegishli deb hisoblanadi.

Nul ko‘rsatkich.

NULL- ko‘rsatkich yagona arifmetik bulmagan konstantadir. Konkret realizatsiyalarda null ko‘rsatkich 0 eki 0L eki nomlangan konstanta NULL orqali tasvirlanishi mumkin. Shuni aytish lozimki bu konstanta qiymati 0 bo‘lishi eki _0‘ simvoli kodiga mos kelishi shart ehmas.

Quyidagi jadvalda konstantalar chegaralari va mos tiplari ko‘rsatilgan:

Ma‘lumotlar turi	Hajm, bit	Qiymatlar chegarasi	Tip vazifasi
Unsigned char	8	0…255	Kichik butun sonlar va simvollar kodlari
Char	8	-128…127	Kichik butun sonlar va ASII kodlar
Enum	16	-32768…32767	Butun sonlar tartiblangan katori
Unsigned int	16	0…65535	Katta butun sonlar
Short int	16	-32768…32767	Kichik butun sonlar, tsikllarni boshqarish
Int	16	-32768…32767	Kichik butun sonlar, tsikllarni boshqarish
Unsigned long	32	0…4294967295	Astronomik masofalar
Long	32	-147483648… …2147483647	Katta sonlar
Float	32	3.4E-32…3.4E+38	Ilmiy hisoblar (7 raqam)
Double	64	1.7E-308…1.7E+308	Ilmiy hisoblar(15 raqam)
Long double	80	3.4E-4932… 1.1E+4932	Moliyaviy hisobalr (19 raqam)

C++ da arifmetik amallar

Ko'p dasturlar ijro davomida arifmetik amallarni bajaradi. C++ dagi amallar quyidagi jadvalda berilgan. Ular ikkita operand bilan ishlatildi.

C++ dagi amal	Arifmetik operator	Algebraik ifoda	C++ dagi ifodasi:
Qo'shish	+	h+19	h+19
Ayirish	-	f-u	f-u
Ko'paytirish	*	sl	s*l
Bo'lish	/	v/d,	v/d
Modul olish	%	k mod 4	k%4

Bularning ba'zi birlarinig hususiyatlarini ko'rib chiqaylik. Butun sonli bo'lishda, yani bo'luvchi ham, bo'linuvchi ham butun son bo'lganda, javob butun son bo'ladi. Javob yahlitlanmaydi, kasr qismi tashlanib yuborilib, butun qismining o'zi qoladi.

Modul operatori (%) butun songa bo'lishdan kelib chiqadigan qoldiqni beradi. x%y ifodasi x ni y ga bo'lgandan keyin chiqadigan qoldiqni beradi. Demak, 7%4 bizga 3 javobini beradi. % operatori faqat butun sonlar bilan ishlaydi. Vergulli (real) sonlar bilan ishlash uchun "math.h" kutubxonasidagi fmod funksiyasini qo‘llash kerak.

C++ da qavslarning ma'nosi huddi algebradagidekdir. Undan tashqari boshqa boshqa algebraik ifodalarning ketma-ketligi ham odatdagidek. Oldin ko'paytirish, bo'lish va modul olish operatorlari ijro ko'radi. Agar bir necha operator ketma-ket kelsa, ular chapdan o'nga qarab ishlanadi. Bu operatorlardan keyin esa qo'shish va ayirish ijro etiladi.

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

Birinchi bo'lib m * 5 hisoblanadi. Keyin 7 % n topiladi va qoldiq (9 + x) ga bo'linadi. Chiqqan javob esa m * 5 ning javobiga qo'shiladi. Qisqasini aytsak, amallar matematikadagi kabi. Lekin biz o'qishni osonlashtirish uchun va hato qilish ehtimolini kamaytirish maqsadida qavslarni kengroq ishlatishimiz mumkin. Yuqoridagi misolimiz quyidagi ko'rinishga ega bo'ladi.

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

Amallar jadvali

Arifmetik amallar	**Razryadli amallar**	**Nisbat amallari**	**Mantiqiy amallar**
+ qo‘shish	& va	= = teng	&& va
- ayirish	| yoki	!= teng emas	|| yoki
* ko‘paytirish	^ inkor	> katta	! inkor
/ bo‘lish	<< chapga surish	>= katta yoki teng	
% modul olish	>> o‘ngga surish	< kichik	
- unar minus	~ inkor	<= kichik yoki teng	
+ unar plyus			
++ oshirish			
-- kamaytirish			

Amallar jadvali (davomi)

Imlo amallar	Qiymat berish va shartli amallar	Tipli amallar	Adresli amallar
() – doirali qavs	= - oddiy qiymar berish	(tip) – tipni o‘zgartirish	& - adresni aniqlash
[] – kavadrat qavs	op= - murakkab qiymat berish	sizeof- hajmni hisoblash	* - adres bo‘yicha qiymat aniqlash yoki joylash
, - vergul	? – shartli amal		

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

Binar amallar additiv ya‘ni + qo‘shuv va – ayirish amallariga , hamda multiplikativ ya‘ni * kupaytirish, / bulish va % modul olish amallariga ajratiladi.

Additiv amallarining ustivorligi multiplikativ amallarining ustivorligidan pastroqdir. Butun sonni butun songa bo‘lganda natija butun songacha yahlitlanadi. Misol uchun 20/3=6; (-20)/3=-6; 20/(-3)=-6.

Modul amali butun sonni butun songa bulishdan hosil bo‘ladigan qoldikka tengdir. Agar modul amali musbat operandlarga qo‘llanilsa, natija ham musbat bo‘ladi, aks holda natija ishorasi kompilyatorga bog‘likdir.

Binar arifmetik amallar bajarilganda tiplarni keltirish quyidagi qoidalar asosida amalga oshiriladi: short va char tiplari int tipiga keltiriladi;

Agar operandlar biri long tipiga tegishli bo‘lsa ikkinchi operand ham long tipiga keltiriladi va natija ham long tipiga tegishli bo‘ladi;

Agar operandlar biri float tipiga tegishli bulsa ikkinchi operand kham float tipiga keltiriladi va natija ham float tipiga tegishli bo‘ladi;

Agar operandlar biri double tipiga tegishli bo‘lsa ikkinchi operand ham double tipiga keltiriladi va natija ham double tipiga tegishli bo‘ladi;

Agar operandlar biri long double tipiga tegishli bo‘lsa ikkinchi operand ham long double tipiga keltiriladi va natija ham long double tipiga tegishli bo‘ladi;

Unar amallarga ishorani o‘zgartiruvchi unar minus – va unar + amallari kiradi. Bundan tashqari ++ va -- amallari ham unar amallarga kiradi.

++ unar amali qiymatni 1 ga oshirishni ko‘rsatadi. Amalni prefiks ya‘ni ++i ko‘rinishda ishlatish oldin o‘zgaruvchi qiymatini oshirib so‘ngra foydalanish lozimligini, postfiks ya‘ni i++ ko‘rinishda ishlatish oldin o‘zgaruvchi qiymatidan foydalanib so‘ngra oshirish kerakligini ko‘rsatadi. Misol uchun i qiymati 2 ga teng bo‘lsin, u holda 3+(++i) ifoda qiymati 6 ga, 3+i++ ifoda qiymati 5 ga teng bo‘ladi. Ikkala holda ham i qiymati 3 ga teng bo‘ladi.

-- unar amali qiymatni 1 ga kamaytirishni ko‘rsatadi. Bu amal ham prefiks va postfiks ko‘rinishda ishlatilishi mumkin. Bu ikki amalni faqat o‘zgaruvchilarga qo‘llash mumkindir. Unar amallarning ustivorligi binar amallardan yuqoridir.

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

6&5 qiyjmati 4 ga ya‘ni 100 ga teng.

6|5 qiyjmati 7 ga ya‘ni 111 ga teng.

6^5 qiymati 3 ga ya‘ni 011 ga teng.

~6 kiyjmati 4 ga yajhni 010 ga teng.

Bu misollarda amallar ustivorligi oshib borishi tartibida berilgandir.

Bu amallardan tashqari M<<N chapga razryadli siljitish va M>>N ungga razryadli siljitish amallari qo‘llaniladi. Siljitish M butun sonning razryadli ko‘rinishiga qo‘llaniladi. N nechta pozitsiyaga siljitish kerakligini ko‘rsatadi.

Chapga N pozitsiyaga surish bu operand qiymatini ikkining N chi daraasiga kupaytirishga mos keladi. Misol uchun 5<<2=20. Bu amalning bitli kurinishi: 101<<2=10100.

Agar operand musbat bulsa N poziciyaga ungga surish chap operandni ikkining N chi darajasiga bo‘lib kasr qismini tashlab yuborishga mosdir. Misol uchun 5>>2=1. Bu amalning bitli kurinishi 101>>2=001=1. Agarda operand qiymati manfiy bulsa ikki variant mavjuddir: arifmetik siljitishda bushatilayotgan razryadlar ishora razryadi qiymati bilan to‘ldiriladi, mantiqiy siljitishda bushatilayotgan razryadlar nullar bilan tuldiriladi.

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

Nisbat amallari. Nisbat amallari qiymatlari 1 ga teng agar nisbat bajarilsa va aksincha 0 ga tengdir. Nisbat amallari arifmetik tipdagi operandlarga yoki ko‘rsatkichlarga qo‘llaniladi. Misollar:

1!=0 qiymati 1 ga teng;

1==0 qiymati 0 ga teng;

3>=3 qiymati 1 ga teng;

3>3 qiymati 0 ga teng;

2<=2 qiymati 1 ga teng;

2<2 qiymati 0 ga teng;

Katta >, kichik <, katta eki teng >=, kichik eki teng <= amallarining ustivorligi bir hildir.

Teng == va teng emas != amallarining ustivorligi uzaro teng va qolgan amallardan pastdir.

Mantiqiy amallar. C ++ tilida mantiqiy tip yukdir. Shuning uchun mantiqiy amallarni butun sonlarga qo‘llanadi. Bu amallarning natijalari quyidagicha aniqlanadi:

x||y amali 1 ga teng agar x>0 eki y>0 bo‘lsa, aksincha 0 ga teng x&&y amali 1 ga teng agar x>0 va y>0 bo‘lsa, aksincha 0 ga teng !x amali 1 ga teng agar x>0 bulsa, aksincha 0 ga teng

Bu misollarda amallar ustivorligi oshib borish tartibida berilgandir.

Inkor ! amali unar kolganlari binar amallardir.

Bu amallardan tashqari quyidagi amallar ham mavjuddir:

Qiymat berish amali. Qiymat berish amali = binar amal bo‘lib chap operandi odatda o‘zgaruvchi ung operandi odatda ifodaga teng bo‘ladi. Misol uchun Z=4.7+3.34 Bu qiymati 8.04 ga teng ifodadir. Bu qiymat Z o‘zgaruvchiga ham beriladi. Bu ifoda ohiriga nuqta vergul ; belgisi quyilganda operatorga aylanadi.

Z=4.7+3.34

Bitta ifodada bir necha qiymat berish amallari qo‘llanilishi mumkin. Misol uchun: C=y=f=4.2+2.8;

Bundan tashqari C ++ tili da murakkab qiymat berish amali mavjud bo‘lib, umumiy ko‘rinishi quyidagichadir:

O‘zgaruvchi_nomi amal= ifoda;

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

Misol uchun:

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

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

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

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

Imlo belgilari amal sifatida. C ++ tilida ba‘zi bir imlo belgilari ham amal sifatida ishlatilishi mumkin. Bu belgilardan oddiy () va kvadrat [] qavslardir. Oddiy qavslar binar amal deb qaralib ifodalarda yoki funksiyaga muroaat qilishda foydalaniladi. Funksiyaga murojaat qilish qo‘yjidagi shaklda amalga oshiriladi:

<funksiya nomi> (<argumentlar ruyhati>). Misol uchun sin(x) eki max(a,b).

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

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

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

d=4,d+2 amali natijasi 8 ga teng.

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

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

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

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

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

(N > 0) ? F : N ifoda N musbat eki manfiyligidan qat‘iy nazar DOUBLE tipiga tegishli bo‘ladi. Shartli ifodada birinchi ifodani qavsga olish shart emas.

Tiplar bilan ishlovchi amallar. Tiplarni o‘zgartirish amali quyidagi ko‘rinishga ega:

(tip_nomi) operand; Bu amal operandlar qiymatini ko‘rsatilgan tipga keltirish uchun ishlatiladi. Operand sifatida kostanta, o‘zgaruvchi yoki qavslarga olinga ifoda kelishi mumkin. Misol uchun (long)6 amali konstanta qiymatini o‘zgartirmagan holda operativ hotirada egallagan baytlar sonini oshiradi. Bu misolda konstanta tipi o‘zgarmagan bo‘lsa, (double) 6 eki (float) 6 amali konstanta ichki ko‘rinishini ham o‘zgartiradi. Katta butun sonlar hakikiy tipga keltirilganda sonning aniqligi yuqolishi mumkin.

sizeof amali operand sifatida ko‘rsatilgan ob‘ektning baytlarda hotiradagi hajmini

hisoblash uchun ishlatiladi. Bu amalning ikki ko‘rinishi mavjud:

sizeof ifoda sizeof (tip) Misol uchun:

Sizeof 3.14=8

Sizeof 3.14f=4

Sizeof 3.14L=10

Sizeof(char)=1

Sizeof(double)=8.

Amallar ustivorligi

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

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

Operatorlar va bloklar.

Har qanday dastur funksiyalar ketma ketligidan iborat bo‘ladi. Funksiyalar sarlavha va funksiya tanasidan iborat bo‘ladi. Funksiya sarlavhasiga void main() ifoda misol bo‘la oladi.

Funksiya tanasi ob‘ektlar ta‘riflari va operatorlardan iborat bo‘ladi.

Har qanday operator nuqta-vergul belgisi bilan tugashi lozim. Quyidagi ifodalar X=0, yoki I++ operatorga aylanadi agar ulardan so‘ng nuqtali vergul kelsa

```
X = 0; I++;
```

Operatorlar bajariluvchi va bajarilmaydigan operatorlarga ajratiladi. Bajarilmaydigan operator bu izoh operatoridir.

Izoh operatori /* belgisi bilan boshlanib */ belgisi bilan tugaydi. Bu ikki simvol orasida ihtiyoriy jumla yozish mumkin. Kompilyator bu jumlani tekshirib o‘tirmayjdi. Izoh operatoridan dasturni tushunarli qilish maqsadida izohlar kiritish uchun foydalaniladi.

Bajariluvchi operatorlar o‘z navbatida ma‘lumotlarni o‘zgartiruvchi va boshqaruvchi operatorlarga ajratiladi.

Ma‘lumotlarni o‘zgartiruvchi operatorlarga qiymat berish operatorlari va nuqta vergul Bilan tugovchi ifodalar kiradi. Misol uchun:

```
I++;
X*=I;
I=x-4*I;
```

Boshqaruvchi operatorlar dasturni boshqaruvchi konstruktsiyalar deb ataladi. Bu operatorlarga quyidagilar kiradi:

Qo‘shma operatorlar; Tanlash operatorlari; Tsikl operatorlari; O‘tish operatorlari;

Qo‘shma operatorlar. Bir necha operatorlar { va } figurali qavslar yordamida qo‘shma operatorlarga yoki bloklarga birlashtirilishi mumkin. Blok eki qo‘shma operator sintaksis jihatdan bitta operatorga ekvivalentdir. Blokning qo‘shma operatordan farqi shundaki blokda obektlar ta‘riflari mavjud bo‘lishi mumkin.

Quyidagi dastur qismi qo‘shma operator:

```
{ n++;
summa+=(float)n; }
```

Bu fragment bo‘lsa blok:

```
{ int n=0; n++;
summa+=(float)n;
}
```

Kiritish chiqarish operatorlari.

Chiquvchi oqim cout kelishilgan buyicha ekranga mos keladi. Lekin mahsus operatorlar yordamida oqimni printer eki faylga mos quyish mumkin. Misol uchun MS-DOS quyidagi komandasi FIRST.EXE dasturi chiqimshini printerga yunaltiradi: S:\> FIRST > PRN <ENTER>

Quyidagi dastur 1001.SRR 1001 sonini ekranga chiqaradi:

```
#include <iostream.h>
void main(void)
{
  cout << 1001;
}
```

Dastur bajarilishi natijasi : S:\> 1001 <ENTER>

1001

Bir necha qiymatlarni chiqarish:

```
#include <iostream.h>
void main(void)
(
  cout << 1 << 0 << 0 << 1;
}
```

Natija:

S:\> 1001TOO <ENTER>

1001

SAVOLLAR

1. Interpretator va kompilyator orasidagi farq nimadan iborat?
2. Dasturning berilgan kodi qanday kompilyatsiya qilinadi?
3. Strukturaviy dasturlash va ob’ektga mo’ljallangan dasturlashning farqi nimadan iborat?
4. Ob’ektga mo’ljallangan dasturlash printsiplarini tushuntirib bering.
5. .srr kengaytmali dastur kodini izoxlab bering.

2-MAVZU: O‘ZGARUVCHI VA O‘ZGARMAS TIPLI KATTALIKLAR

Reja

1. O’zgaruvchilar va o’zgarmaslar
2. C++ tilidagi dasturlarning tarkibiy qismlari.
3. Standart funktsiyalar. Ifoda va operatorlar.Arifmetik operatsiyalar
4. C++ tilida oddiy dastur.
5. Funktsiyalar. Ishorali va ishorasiz tiplar.

Tayanch iboralar: O‘zgaruvchi, o‘zgaruvchining aniqlanishi, xotirani rezevrlanishi, ishorali va ishorasiz tiplar, kalitli so’zlar belgilar, maxsus belgilar, o’zgarmaslar, literal o’zgarmaslar, belgili o’zgarmaslar, #define direktivasi, sonst kalitli so’zi, sanoqli o’zgarmaslar.

***Dars maqsadi*:** Talabalarda C++ dasturlash tilida o’zgaruvchi va o’zgarmas tipli kattaliklardan foydalanish ko’nikmalarini hosil qilish.

O‘zgaruvchilar va o‘zgarmaslar

Dastur o’zi ishlatadigan ma’lumotlarni saqlash imkoniyatiga ega bo’lishi lozim. Buning uchun o’zgaruvchilar va o’zgarmaslardan foydalaniladi. Dastur bajarilishi jarayonida o’z qiymatini o’zgartirmaydigan kattaliklar **o’zgarmaslar**, ya’ni konstanta deb ataladi. Dastur bajarilishi jarayonida o’z qiymatini o’zgartira oladigan kattaliklar **o’zgaruvchilar** deyiladi. O’zgaruvchi nomlari xarflar yoki xarf va sonlardan iborat bo’lishi mumkin. C++ tilida o’zgaruvchilarni belgilashda katta va kichik xarflarning farqi bor. Masalan: A va a xarflari ikki xil o’zgaruvchini bildiradi. Ushbu mavzuda quyidagilarni bilib olamiz.

- O’zgaruvchi va o’zgarmaslarni qanday aniqlash kerak.
- O’zgaruvchilarga qanday qiymat berish kerak va ularni dasturda qanday ishlatish lozim.
- O’zgaruvchi qiymati qanday ekranga chiqariladi.

O‘zgaruvchi nima? C++ tilida o‘zgaruvchilar ma’lumotni saqlash uchun qo‘llaniladi. O‘zgaruvchining dasturda foydalanish mumkin bo‘lgan qandaydir qiymatlarni saqlaydigan komyuter xotirasidagi yacheyka ko‘rinishda ifodalash mumkin.

Kompyuter xotirasini yacheykalardan iborat qator sifatida qarash mumkin. Barcha yacheykalar ketma – ket nomerlangan. Bu nomerlar yacheykaning adresi deb ataladi. O‘zgaruvchilar biror – bir qiymatni saqlash uchun bir yoki bir nechta yacheykalarni band qiladi.

O‘zgaruvchining nomini (masalan, MyVariable) xotira yacheykasi adresi yozilgan yozuv deb qarash mumkin

Xotirani zahiralash. C++ dasturlash tilida o‘zgaruvchini aniqlash uchun kompyuterga uning tipi (masalan, int, char yoki boshqa) haqida ma’lumot beriladi. Bu axborot asosida kompilyatorga o‘zgaruvchi uchun qancha joy ajratish lozim va bu o‘zgaruvchida qanaqa turdagi qiymat saqlanishi mumkinligi haqida ma’lumot aniq bo‘ladi.

Har bir yacheyka bir bayt o‘lchovga ega. Agar o‘zgaruvchi uchun ko‘rsatilgan tip 4 baytni talab qilsa, uning uchun to‘rtta yacheyka ajratiladi. Aynan o‘zgaruvchini tipiga muvofiq ravishda kompilyator bu o‘zgaruvchi uchun qancha joy ajratish kerakligini aniqlaydi.

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

Butun sonlar o‘lchami. Bir xil tipdagi o‘zgaruvchilar uchun turli kompyuterlarda xotiradan turli hajmdagi joy ajratilishi mumkin. Lekin, bitta kompyuterda bir xil tipdagi ikkita o‘zgaruvchi bir xil miqdorda joy egallaydi.

shar tipli o‘zgaruvchi bir bayt hajmni egallaydi. Ko‘pgina kompyuterlarda short int (qisqa butun) tipi ikki bayt, long int tipi esa 4 bayt joy egallaydi. Butun qiymatlar o‘lchovini kompyuter sistemasi va ishlatiladigan kompilyator aniqlaydi. 32 - razryadli kompyuterlarda butun o‘zgaruvchilar 4 bayt joy egallaydi.

- Butun tip (toifa)li o‘zgarmaslar: ular faqat butun sonlardan iborat bo‘ladi. Masalan: 15, 64, 1964, - 21 va h.k.
- Haqiqiy tipli o‘zgarmaslar: ular butun va kasr qismlardan iborat bo‘ladi. Masalan: 1,5 15,64 va h.k.

Haqiqiy tipli sonlarning bu ko‘rinishi oddiy ko‘rinish deyiladi. Juda katta va judda kichik haqiqiy tipli sonlarni darajali (eksponensial) ko‘rinishda yozish qulay. Masalan: $4,5*10^{-11}$ yoki $6,5*10^{11}$ kabi sonlar 4.5E-11 va 6.5E11.

- Simvolli konstantalar. Ular qatoriga dastur bajarilishi davomida o‘zgarmaydigan barcha simvollarni kiritish mumkin.

C++ tilida o‘zgaruvchi va o‘zgarmaslarni tipini belgilashda quyidagilar ishlatiladi. C++ dasturlash tilida har qanday o‘zgaruvchi ishlatilishidan oldin e’lon qilinishi kerak. E’lon qilish degani ularning tiplarini aniqlab qo‘yish maqsadga muvofiqdir:

- Butun toifali o‘zgaruvchilar: int. Masalan: int a,b,i,j; bu yerda dasturda ishlatilayotgan a,b,i,j o‘zgaruvchilarning toifasi butun ekanligi ko‘rsatildi. Bu toifadagi o‘zgaruvchilar xotiradan 2 bayt joy egallaydi. Ularning o‘zgarish intervali (diapazoni): -32768 dan + 32767 gacha. Bu Windows 98 Windows NT da 32 razryadli butun sonlar foydalaniladi.
- Butun toifali katta (uzun) o‘zgaruvchilar: long. Masalan: long a,b,s,dd2. Bu toifadagi o‘zgaruvchilar xotiradan 4 bayt joy egallaydi. Ularning o‘zgarish intervali (diapazoni): -2147483648 dan + 2147483647 gacha.

- Ishorasiz butun o‘zgaruvchilar: unsigned short. Xotiradan 2 bayt joy egallaydi, o‘zgarish intervali (diapazoni): 0 dan 65535 gacha. unsigned long. Xotiradan 4 bayt joy egallaydi, o‘zgarish intervali (diapazoni): 0 dan 4 294967 295 gacha. Unsigned shar. Xotiradan 1 bayt joy egallaydi, o‘zgarish intervali (diapazoni): 0 dan 255 gacha.
- Haqiqiy toifadagi o‘zgaruvchilar: float. Masalan: float a,v; bu yerda a, v o‘zgaruvchilarning toifasi haqiqiy ekanligi keltirilgan. Bu toifadagi o‘zgaruvchilar xotiradan 4 bayt joy egallaydi, o‘zgarish intervali 10^{-38} dan 10^{+38} gacha. Juda katta yoki juda kichik o‘zgaruvchilarni belgilashda double toifasi ishlatiladi va xotiradan 8 bayt joy oladi.
- Simvolli o‘zgaruvchilar uchun char toifasi ishlatiladi. Xotiradan 1 bayt joy egallaydi, o‘zgarish intervali (diapazoni): -128 dan + 127 gacha. Simvolli o‘zgaruvchilar apostrof (‘) ichiga olinib yoziladi va ASCII kodlariga mos ravishda tanlanadi.
- Qator tipidagi o‘zgaruvchilar uchun xam char toifasi belgilangan. Xotiradan 1 bayt joy egallaydi, o‘zgarish intervali (diapazoni): -0 dan 256 tagacha bo‘lgan simvollar ketma-ketligidan iborat bo‘lishi mumkin. Qator toifasidagi o‘zgaruvchilar qo‘shtirnoq (“) ichida yoziladi. Tekst (ma’lumotlarning char turi)- a,z,? va 2 kabi simvollar ketma-ketligidan iborat bo‘lishi mumkin. Odatda har bir simvol 8 bit yoki bir bayt joy egallaydi.

Demak, yuqoridagilar asosiy tiplar hisoblanar ekan. Bular: Char, short, int, long, float, double. Bu yerda dastlabki 4 ta tur butun, qolgan 2tasi xaqiqiy sonlar uchun ishlatiladi. Agar tiplar hajmini taqqoslab ifodalaydigan bo‘lsak, l=sizeof(char)<=sizeof(short) l=sizeof(int)<= sizeof(long) <=sizeof(float)<=sizeof(double).

Umuman C++ tilida ma’lumotlar asosan quyidagi 8ta turga bo‘linishi mumkin. Masalan: char(matnli ma’lumotlar), int(butun sonlar), float(birlik aniqlikdagi haqiqiy sonlar), double (ikkilik aniqlikdagi haqiqiy sonlar), void(bo‘sh qiymatlar), bool(mantiqiy qiymat) va h.k. Endi har bir turga alohida to‘xtalib o‘tamiz:

C++ dasturlash tilida o‘zgaruvchilarni initsializatsiya qilish degan tushuncha xam mavjud. Initsializatsiya qilish degani o‘zgaruvchini e’lon qilish barobarida unga boshlang‘ich qiymatini ham berish mumkin. Masalan: int a=19, b=2, s=100.

Dasturlash

C++ tilidagi dasturlarning tarkibiy qismlari. C++ tilida tuzilgan dastur ob’ektlar, funksiyalar, o‘zgaruvchilar va boshqa elementlardan tashkil topadi. Ushbu mavzuning asosiy qismi ularning har birini to‘liq tavsiflashga bag‘ishlangan. Lekin bu elementlarni uyg‘unlashgan holda qarash uchun biror bir tugallangan ishchi dasturni qarab chiqish kerak. Ushbu mavzudan qo‘yidagilarni bilib olish mumkin.

* C++ tilida dasturlar qanday qismlardan tuzilgan.
* Bu qismlar bir-biri bilan qanday aloqa qiladi.

* Funksiya nima va uning vazifasi nimadan iborat.

```
# include <iostream.h>
# include <math.h>
main ()
{
tip <o‘zgaurvchilar>;
kiritish operatori ("format", & o‘zgaruvchilar) ;
hisoblash tanasi;
chop qilish qismi
}
```

Ma’lumki barcha amallar tezkor yoki doimiy xotirada bajariladi, ya’ni har qanday o‘zgaruvchilarning qiymatlari xotiraga kiritiladi va barcha operatsiyalar xotiradagi o‘zgaruvchilar ustida bajariladi. Natija xotirada paydo bo‘ladi.

Dasturda o‘zgaruvchilar uchun quyidagi formatlar ishlatiladi:

% s - simvolli o‘zgaruvchi uchun.

% s - qator tipidagi o‘zgaruvchi uchun.

% d - butun tipidagi o‘zgaruvchi uchun.

%ld - butun tipidagi katta o‘zgaruvchilar uchun (long)

%lu - butun tipidagi ishorasiz katta sonlar uchun.

%u - butun tipidagi ishorasiz sonlar uchun (char,int….).

%f - haqiqiy tipidagi o‘zgaruvchilar uchun (float).

%e - eksponensial formadagi haqiqiy o‘zgaruvchilar uchun.

%g - haqiqiy tipdagi natijani yaxlitlab, qisqa ko‘rinishda olish uchun.

%lf - haqiqiy tipdagi katta o‘zgaruvchilar uchun (long, float)

%le - eksponensional formadagi haqiqiy o‘zgaruvchilar (juda katta yoki juda kichik) uchun

%lg - haqiqiy tipdagi katta natijani yaxlitlab, qisqa ko‘rinishda olish uchun.

Bundan tashqari o‘zgaruvchilarni 8 va 16 lik sanoq sistemasidagi ko‘rinishida xam kiritish mumkin. Bular uchun alohida formatlar belgilangan.

% 0 – butun tipdagi 8 s/sistemasidagi o‘zgaruvchilar uchun

% x - butun tipdagi 16 s/sistemasidagi o‘zgaruvchilar uchun

% lo - butun tipdagi katta 8 s/sistemasidagi o‘zgaruvchilar uchun

% lx - butun tipdagi katta 16 s/sistemasidagi o‘zgaruvchilar uchun

Izoh: 8 lik sanoq sistemasidagi sonlar 0-7 gacha bo‘lgan raqamlardan tashkil topgan. Masalan: a=025674, s=1, vv=02233556677 va h.k.

16 lik s/s.dagi sonlar 0- x yoki 0 – X (bu 0 dan boshlanib x yoki X simvoli kelishligini bildiradi). Masalan: a=0x79, v=0XV64 va h.k.

Standart funksiyalar (1-jadvalda) keltirilgan.

1-jadval

Matematik yozilishi	C++ d/t yozilishi	Matematik yozilishi	C++ d/t yozilishi
Sin x	sin(x)	$\log_a b$	log(b)/log(a)

Cos x	cos(x)	x^n	pow(x,n)
tgx	tan(x)	$\sqrt{x}$	sqrt(x)
e^x	exp(x)	|x|	abs(x)
Lnx	log(x)	arctgx	atan(x)
Lgx	log10(x)	shx	exp(x)-exp(-x)/2
x^2	sqr(x)	chx	exp(x)+exp(-x)/2

Ifoda va operatorlar. C++ dasturlash tilida operatsiyalar keng tuzilishga ega bo‘lib, ular yordamida ifodalarga yangi qiymatlar qabul qilinadi va o‘zgaruvchilar qiymatlari o‘zgartiriladi. C++ dasturlash tilining ayrim operatorlari (2-jadval).

2-jadval

amal, (vazifasi)	operatorlar	Nomlanishi, (vazifasi)	operatorlar
Massiv elementiga murojat	[ ]	Funksiyani chaqirish	( )
Inkrement (qiymat oshirish)	++	Ko‘paytirish	*
Dekrement (qiymat kamaytirish)	- -	Bo‘lish	/
Modul bo‘yicha bo‘lish	%	qo‘shish	+
Ayirish	-	katta	<
Kichik	>	Katta yoki teng	<=
Kichik yoki teng	>=	Teng (mantiqiy)	= =
Teng emas	!=	Mantiqiy inkor	!
Mantiqiy va	&&	Mantiqiy yoki	||
Shartli ifoda	?:	Ko‘paytirish bilan tenglash	*=
Bo‘lish bilan tenglash	/=	Modul bo‘yicha bo‘lish bilan tenglash	%=
Qo‘shish bilan tenglash	+=	Ayirish bilan tenglash	-=

C++ dasturlash tilida ishlatiladigan asosiy operatsiyalar va ularning strukturasi.

C++ dasturlash tili 26ta lotin harflari (katta va kichik farqlanadi, demak 52 ta).
Arab raqamlari: 0,1,2,....., 9
Maxsus belgilar: =, -, *, /, _, |, !, #, %, $, &, (), { }, [ ], ^, @ va h.k.
Maxsus simvollar bir-biridan probel (bo‘sh joy) bilan ajralib turadi.

Arifmetik operatsiyalar: +(qo‘shish), - (ayirish),
* (ko‘paytirish), / (bo‘lish).

Amal bajarish jarayonida butun sonni butun songa bo‘lganda natija har doim xam butun chiqmaydi. Masalan: 5/2=3 ko‘rinishda natija chiqadi, agar buni 5/2. deb yozsak, u holda natija 2,5 chiqadi. Modul bo‘yicha bo‘lish amaliga to‘xtalamiz.

```
….int  a=3, b=8, c=0, d;
d=b%a; // natija 2
d=a%b; // natija 0
d=b% c; // xatolik haqida ma’lumot
```

Modul bo‘yicha bo‘lishda, bo‘lishdan qolgan qoldiq qiymati hosil bo‘ladi. Oxirgi satrda nolga bo‘lish sodir bo‘lganligi uchun kompilyator xato haqida xabar beradi.

Taqqoslash operatsiyalari: >, <, <=, >=, (!= -teng emas), (== mantiqiy teng).

Mantiqiy operatsiyalar: && - mantiqiy ko‘paytirish (va), || - mantiqiy qo‘shish (yoki), ! –mantiqiy inkor (emas).

Mantiqiy operatsiyalarni ixtiyoriy toifadagi sonlar ustida bajarish mumkin. Agar javob **rost** bo‘lsa, natija 1 ga teng chiqadi, agar javob yolg‘on bo‘lsa, natija 0 chiqadi.

Masalan: (i<25 && j=64); (s1<s2 && (s3>6,4|| s4= =5))

Qiymat berish operatsiyalari: yuborish operatsiyasi (=). Masalan: a=5, b=2, x=y=z=1 yoki a=(b=c)+d ko‘rinishda yozish mumkin.

Qabul qildim va almashtirdim deb nomlanadigan operatsiyalar:

+=: a+=v yozuvi a=a+v ni anglatadi

-=: a-=v yozuvi a=a-v ni anglatadi

=: a=v yozuvi a=a*v ni anglatadi

/=: a/=v yozuvi a=a/v ni anglatadi

Inkrement (++) va dekrement (--) operatsiyalari ikki ma’noda ishlatiladi.

a) i++ - o‘zgaruvchining qiymati unga murojaat qilganimizdan keyin 1 ta oshiriladi (i= i+1).

++ i - o‘zgaruvchining qiymati unga murojat qilishdan oldin 1 taga oshiriladi.

v) i - - - o‘zgaruvchining qiymati unga murojat qilganimizdan keyin 1 ta kamayadi.

- - i - o‘zgaruvchining qiymati unga murojat qilishdan oldin 1 taga kamayadi.

Izoh: murojat qilishdan oldin ko‘payadigan yoki kamayadigan operatsiyalar **prefiks** amallari, murojat qilganidan keyin ko‘payadigan yoki kamayadigan operatsiyalar **postfiks** amallari deb yuritiladi.

C++ tilida oddiy dastur. C++ tilida yozilgan dastur quyidagi tarkibdan tashkil topgan.

1. Direktivalar- funksiyalar kutubxonasini chaqirish (yuklash). Ular maxsus **include** katalogida joylashgan va .h fayllar deb nomlanadi. Dasturda masalaning qo‘yilishiga qarab kerakli **include** lar chaqiriladi**.** Bu dasturning xotirada egallaydigan joyini minimallashtiradi.

Agar # include satrini T.Paskal tiliga solishtiradigan bo‘lsak, modullar yoki maxsus operator va funksiyalar joylashgan bibliotekalar ko‘rsatiladi. C++ dasturlash tilining yana bir ahamiyatli tomoni shundan iboratki, maxsus bibliotekalarga yo‘l ko‘rsatish imkoniyati mavjud. Masalan:

include “math.h”

include “/user/pol/math2.h”

Bu yerda math.h fayli joriy foydalanuvchi katalogida, math2.h esa /user/pol katalogida joylashgan, ya’ni kompilyator shu yerga murojat qiladi.

C++ dasturlash tilini o‘rganamiz. Ushbu dasturni ko‘rib chiqamiz:

```
# include <iostream.h>
# define ITF “C++ dasturlash tilini o‘rganamiz”
# define Dars 1
# define kerakliligi 100 %
void main(void)
{
cout <<”kitob nomi:“ << ITF<< endl;
cout <<”navbatdagi mavzu:“ << Dasr<< endl;
cout <<”Umrboqiy:@“ << kerakliligi << endl;
}
```

Natija:

Kitob nomi: C++ dasturlash tilini o‘rganamiz
navbatdagi mavzu: 7
Umrboqiy: 100 %

Masalan: berilgan x va u qiymatlarida A va V ifodalarni hisoblash dasturini tuzish kerak bo‘lsin.

$$A=\left|\frac{\sin^3(\pi-x)}{\sqrt{(x-y)^2+e^{(-x^2)}}}\right|;\ldots\ldots B=\frac{\sqrt{tg\pi}}{A*\ln(2*10^3-\cos^2(x-y))}$$

Bu yerda x=6,3 u=1,2.

Dastur ko‘rinishi:

```
{ arifmetik ifodalar C++ dasturlash tilida}
#include <iostream>
#include <math.h>
using namespace std;
```

```
void main()
{
double pi=3.14159265;
double x=6.3,y=1.2,a,b;
cout<<"Natija :"<<endl;
a=abs(exp(3*log(sin(pi-x)))/sqrt((x-y)*(x-y)+exp(-x*x)));
b=exp((1/3)*log(sin(pi/3)/cos(pi/3)))/(a*log(2000-cos(x-y)*cos(x-y)));
cout<<a<<endl;
cout<<b<<endl;
    }
```

include - preprotsessorning komandasi bo‘lib, u quyidagicha tarjima qilinadi: «Bu komandani ortidan fayl nomi keladi. Ushbu nomdagi faylni topish va fayldagi mazmunni dasturning joriy kismiga yozish lozim».

Burchakli qavs ichidagi faylni mos fayllar joylashtirilgan barcha papkalardan izlash lozimligini ko‘rsatadi. Agarda kompilyator to‘g‘ri sozlangan bo‘lsa burchakli qavslar `iostream.h` faylini sizning kompilyatoringiz uchun mo‘ljallangan `.h` kengaytmali fayllarni o‘zida saqlovchi papkadan izlashi kerakligini ko‘rsatadi. `iostream.h` (`input-output stream` – kiritish–chiqarish oqimi) faylida ekranga ma’lumotlarni chiqarish jarayonini ta’minlaydigan cout ob’ekti aniqlangan. Birinchi qator bajarilgandan so‘ng iostream.h fayli joriy dasturga xuddi uning mazmunini qo‘l bilan yozganimizdek biriktiriladi. Preprotsessor kompilyatordan keyin yuklanadi va funt (#) belgii bilan boshlanuvchi barcha qatorlarni bajaradi, dastur kodlarini kompilyatsiyaga tayyorlaydi.

Dasturning asosiy kodi main() funksiyasini chaqirish bilan boshlanadi. C++ tilidagi har bir dastur main() fuksiyasini o‘zida saqlaydi. Funksiya bu bir yoki bir necha amalni bajaruvchi dastur blokidir. Odatda funksiyalar boshqa funksiyalar orqali chaqiriladi, lekin `main()` funksiyasi alohida xususiyatga ega bo‘lib u dastur ishga tushirilishi bilan avtomatik tarzda chaqiriladi. Dasturni butunlay xotirangizdan o‘chirib yubormaslik va boshqalarga ham tushunarli bo‘lishi uchun izohlardan foydalanish lozim. Izohlar kompilyator tomonidan tushirib qoldiriladigan dasturning alohida satrida yoki butun bir blokida qo‘llaniladi. Quyidagi listingni ko‘rib chiqamiz.

1-listing. Salom.CPP dasturi misolida C++ tilida tuzilgan dastur qismlarini namoyish qilish.

```
1: // Salom.CPP dasturi
2: #include <iostream.h >
3: int main( )
4: {
5:  cout << “Salom!\n”;
6:  return 0;
```

```
7:  }
```
NATIJA:
Salom!

TAHLIL: 1-satrda iostream.h fayli joriy faylga biriktilyapti. Dasturda birinchi funta (#) belgisi joylashgan. U prepotsessorga signal uzatadi. Kompilyatorning har safar ishga tushirilishida preprotsessor ham ishga tushiriladi. U dasturdagi funta (#) belgisi bilan boshlanuvchi qatorlarni o‘qiydi.

Salom.cpp dasturi misolida izohlarni namoyish qilish.

```
1:  # include < iostream.h >
2:   main()
3:   {
4:    cout << “Salom !\ n”;
5:  /* bu izoh toki izohning
6:  oxirini ko‘rsatuvchi belgi, ya’ni  yulduzcha
7:  va slesh belgisi uchramaguncha davom etadi */
8:  cout << “Bu kommentariy yakunlandi\ n”;
9:  // bu izoh satrni oxirida tugaydi.
10:// Ikkita  sleshdan so‘ng xech qanday tekst
11:// bo‘lmasligi mumkin.
12:return 0;
13:}
```

NATIJA
Salom
Bu kommentariy yakunlandi

main() funksiyasini boshqa funksiyalar kabi qaytaradigan qiymat tipini e’lon qilish lozim. Salom.cpp dasturida `main()` funksiyasi `int` (`integer` – butun so‘zidan olingan) tipli qiymat qaytaradi, ya’ni bu funksiya ishini tugatgandan so‘ng operatsion sistemaga butun sonli qiymat qaytaradi. Operatsion sistemaga qiymat qaytarish unchalik muhim emas, umuman sistema bu qiymatdan foydalanmaydi, lekin C++ tili standarti `main()` funksiyasi barcha qoidalarga muvofiq e’lon qilinishini talab qiladi.

Изох	Ayrim kompilyatorlar `main()` funksiyasini `void` tipidagi qiymat qaytaradigan qilib e’lon qilish imkonini beradi. C++ da bundan foydalanmaslik kerak, chunki hozirda bunday uslub eskirgan. `main()` funksiyasini `int` tipini qaytaradigan qilib aniqlash lozim va buning hisobiga funksiyaning oxiriga `return 0` ifodasi yoziladi.

Barcha funksiyalar ochiluvchi figurali qavs ({) bilan boshlanadi va (}) yopiluvchi qavs bilan tugaydi.

`main( )` funksiyasi figurali qavsida 3-satrdan 6-satrgacha joylashtirilgan. Figurali qavslarni ichida joylashgan barcha satrlar funksiya tanasi deb aytiladi.

Bizning oddiy dasturimizning barcha funksional-ligi 4-satrda keltirilgan. `sout` ob'ekti ekranga ma'lumotni chiqarish uchun qo'llaniladi. `cin` va `cout` ob'ektlari mos ravishda ma'lumotlarni kiritish (masalan, klaviatura orqali) va ularni chiqarish (ekranga chiqarish) uchun qo'llaniladi. `main()` funksiyasi 6-satr bilan tugallanadi.

`cout` ob'ekti haqida qisqacha ma'lumot. Keyingi mavzularda siz `sout` ob'ektini qanday ishlatish lozimligini bilib olasiz. Hozir esa u haqida qisqacha ma'lumot beramiz. Ekranga ma'lumotni chiqarish uchun cout_so'zini, undan so'ng chiqarish operatorini (<<) kiritish lozim. C++ kompilyatori (<<) belgisini bitta operator deb qaraydi. Quyidagi listingni tahlil qilamiz. 2-l**isting. cout ob'ektini qo'llanilishi.**

```
1: //2-listing sout ob'ektini ko'llanilishi//
2: # include <iostream.h >
3: int main()
4: {
5: cout << "Bu son 5 ga teng:" << 5<< "\n";
6: cout <<"endl operatori ekranda yangi
7: cout << " satrga o'tish amalini bajaradi";
8: cout <<endl;
9: cout << "Bu katta son:\t"<< 70000 <<
10:endl;
11:cout << "Bu 5 va 8 sonlarining yig'indisi:
12:<<\t"<< 8+5 << endl;

13:cout <<Bu kasr son:\t\t"<< (float) 5\8 << endl;
14:cout << "Bu esa juda katta son: \t";
15:cout << (double) 7000*7000 << endl;
16:return 0;
17:};
```

NATIJA:

```
Bu son 5 ga teng: 5
endl operatori ekranda yangi satrga o'tish amalini bajaradi
Bu katta son: 70000
Bu  5 va  8 sonlarining yig'indisi: 13
Bu kasr son:  0.625
```

Bu esa juda katta son: 4.9e+07

Изох

endl operatori end line (catr oxiri) degan so‘zdan olingan bo‘lib «end-el» deb o‘qiladi.

Изох

Ayrim kompilyatorlarda sout ob’ektidan keyin matematik operatsiyalarni bajarish uchun figurali qavslarni ishlatish talab qilinadi. U holda 2-listingning 11-satrida quyidagicha almashtirish bajarish lozim.

11: cout << «Here is the sum of 8 and 5<< (8+5) << endl;

Funksiyalar. Biz oldinroq main() funksiyasi bilan tanishib chiqqan edik. Bu funksiya odatdagi bo‘lmagan, yagona turdagi funksiyadir. Funksiyalar dasturning ishlash davrida chaqirilishi kerak. `main()` funksiyasi esa dastur tomonidan emas, balki operatsion sistema tomonidan chaqiriladi.

Dastur berilgan matni bo‘yicha satrlarni joylashishiga qarab tartib bilan toki biror bir funksiya chaqirilguncha bajariladi. Keyin esa boshqaruv birinchi uchragan funksiyaga beriladi. Funksiya bajarilgandan so‘ng boshqaruv yana dasturning funksiya chaqirilgan joyidan keyingi satriga beriladi. (Chaqirilgan funksiyadan keyingi satrga beriladi.)

Funksiyani ishlash jarayoniga mos o‘xshashlik mavjud. Masalan, siz rasm chizib turgan vaqtingizda qalamingiz sinib qoldi. Siz rasm chizishni to‘xtatasiz va qalamni yo‘na boshlaysiz. Keyin esa, rasm chizishni qalamingiz sinib qolgan joydan boshlab davom ettirasiz. Qachonki, dastur biror bir xizmat ko‘rsatuvchi amallarni bajarilishiga ehtiyoj sezsa kerakli funksiyani chaqiradi. Bu operatsiya bajarilgandan keyin esa dastur o‘z ishini funksiya chaqirilgan joydan boshlab davom ettiradi. Bu g‘oya quyidagi listingda namoyish etilgan.

Funksiyalarning qo‘llanilishi. Funksiyalar yo `void` tipidagi, yo boshqa biror bir tipdagi qiymat qaytaradi. Ikkita butun sonni qo‘shib, ularning yig‘indisini qaytaradigan funksiya butun qiymat qaytaruvchi deyiladi. Faqatgina qandaydir amallarni bajarib, hech qanday qiymat qaytarmaydigan funksiyaning qaytaruvchi tipi `void` deb e’lon qilinadi.

Funksiya sarlavha va tanadan iboratdir. Funksiya sarlavhasida uning qaytaradigan tipi, nomi va parametrlari aniqlanadi. Parametrlar funksiyaga qiymat uzatish uchun ishlatiladi.

3-listing. Funksiyani chaqirilishiga misol.

```
1: # include  < iostream.h >
2: //namoyish funksiyasi ekranga
```

```
3:  // axborot ma'lumot chiqaradi.
4:  void namoyish funksiyasi ()
5:  {
6:  cout<< "\n namoyish funksiyasi chaqirildi\ n;
7:  }
8:  //main() funksiyasi oldin axborot chiqaradi va
9:  // namoyish funksiyasini chaqiradi
10: // Keyin yana namoyish funksiyasini chaqiradi
11: int main()
12: {
13:  cout << "\ n Bu main() funksiyasi \ n";
14: namoyish funksiyasi ();
15: cout << "main() funksiyasiga qaytildi\n";
16: return 0;
17: }
```

NATIJA:

Bu main() funksiyasi namoyish funksiyasi chaqirildi va so‘ng main() funksiyasiga qaytildi.

Parametr-bu funksiyaga uzatiladigan qiymat tipini e’lon qilishdir. Funksiya chaqirilganda unga uzatiladigan qiymat argument deb aytiladi. Ko‘pchilik dasturchilar bu ikkala tushunchani sinonim sifatida qarashadi. Ba’zilar esa bu terminlarni aralashtirishni noprofessionallik deb hisoblaydi. Mavzularda ikkala termini bir xil ma’noda kelgan.

Funksiya tanasi ochiluvchi figurali qavs bilan boshlanadi va u bir necha qatordan iborat bo‘lishi mumkin. (Funksiya tanasida hech qanday satr bo‘lmasligi ham mumkin). Satrlardan keyin esa yopiluvchi figurali qavs keladi. Funksiyaning vazifasi uning satrlarida berilgan dasturiy kodlar bilan aniqlanadi. Funksiya dasturga return operatori orqali qiymat qaytaradi. Bu operator funksiyadan chiqish ma’nosini ham anglatadi.

Agarda funksiyaga chiqish operatorini (return) qo‘ymasak funksiya satrlarini tugashi bilan u avtomatik void tipidagi qiymatni qaytaradi. Funksiya qaytaradigan tip uning sarlavhasida ko‘rsatilgan tip bilan bir xil bo‘lishi lozim. Bundan tashqari bizga uchburchakning ma’lum parametrlariga ko‘ra noma’lum parametrlarini topish kerak bo‘lsin. U holda quyidagi formulalardan foydalanamiz:

$$\frac{a}{\sin\alpha} = \frac{b}{\sin\beta} = \frac{c}{\sin\gamma}$$

$$c = \sqrt{a^2 + b^2 - 2ab\cos\gamma}$$

Va xakazo. Masalan: Ixtiyoriy uchburchakning 3ta (a,b,c) tomonlari berilgan bo‘lsin, qolgan 7ta noma’lum parametrlarini aniqlash dasturini tuzing.

Dasturi:

```
#include <iostream>
```

```
#include <math.h>
using namespace std;
void main()
{    double pi=3.14159265;
     double a=3,b=4,c,A,B,C=pi/4,r,R,p,S;
     c=sqrt(a*a+b*b-2*a*b*cos(C));
     A=asin((a/c)*sin(C));
     B=asin((b/c)*sin(C));
     p=(a+b+c)/2; S=a*b*sin(C)/2;
     r=S/p; R=a*b*c/(4*S);
     cout<<"Yarim perimetr :  p= "
<<p<<endl;
     cout<<"Alfa burchak   :  A= "
<<A<<endl;
     cout<<"Beta burchak   :  B= "
<<B<<endl;
     cout<<"c tomon uzunligi:  c= "
<<c<<endl;
     cout<<"Ichki aylana radiusi:r= "
<<r<<endl;
     cout<<"Tashqi aylana radiusi: R="
<<R<<endl;
     cout<<"Yuzi: S= "
<<S<<endl;
     getchar();
    }
```

2- misol.

```
1:  # include  <iostream. h>
2:  int main()
3:  {
4:  cout << “int tipining o‘lchami: \t”
5:  <<sizeof(int)<<” bayt.”;
6:  cout<< “short int  tipining o‘lchami:\t”
7:  << sizeof(short)<< “bayt.”;
8:  cout<<“long int tipining o‘lchami:\t”
9:  << sizeof(long)<<“bayt.”;
10:cout<< char tipining o‘lchami:\t”
11:<< sizeof(long)<<”\t bayt.”;
12:return 0;
13:};
```

NATIJA:

int tipining o‘lchami: 4 bayt.
short int tipining o‘lchami: 2 bayt.
long int tipining o‘lchami: 4 bayt.;
char tipining o‘lchami: 1 bayt;

Ishorali va ishorasiz tiplar. Dasturda qo‘llaniladigan butun sonli tiplar ishorali va ishorasiz bo‘lishi mumkin. Ba’zan o‘zgaruvchi uchun faqatgina musbat sonni qo‘llash foydali bo‘ladi. `Unsingned` kalitli so‘ziciz keltirilgan butun sonli tiplar (`short` va `long`) ishorali hisoblanadi. Ishorali butun sonlar manfiy va musbat bo‘lishi mumkin. Ishorasiz sonlar esa doimo musbat bo‘ladi.

O‘zgaruvchilarning tayanch tiplari. C++ tilida boshqa berilganlar tiplari ham qaralgan. Ular butun sonli, haqiqiy va belgili bo‘lishi mumkin. Haqiqiy o‘zgaruvchilar kasr ko‘rinishda ifodalanuvchi qiymatlarni ham o‘zida saqlaydi. Belgili o‘zgaruvchilar bir bayt joy egallaydi va 266 ta belgi hamda `ASCII` belgilarni saqlash uchun ishlatiladi.

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

Kalit so‘zlar. C++ tilida ayrim so‘zlar oldindan zahiralanadi. Bular kalitli so‘zlar deb aytiladi. Bunday so‘zlarni o‘zgaruvchilarni nomlashda ishlatish mumkin emas. Ularga `if, while, for` va `main` kabi so‘zlar kiradi. Kompilyatorning texnik xujjatlarida barcha zahiralangan so‘zlarning ro‘yxati turadi.

O‘zgaruvchiga qiymat berish. O‘zgaruvchilarga qiymat berish uchun o‘zlashtirish operatori qo‘llaniladi.

Masalan, `Width` o‘zgaruvchisiga 5 qiymatni berish uchun quyidagilarni yozish lozim:

```
unsigned short Width;
Width = 5;
```

Bu ikkala satrni `Width` o‘zgaruvchisini aniqlash jarayonida birgalikda yozish mumkin.

```
unsigned short Wigth = 5;
```

Bir necha o‘zgaruvchilarni aniqlash vaqtida ham ularga qiymat berish mumkin:

```
long width = 5,  length = 7;
```

Bu misolda `long` tipidagi `width` o‘zgaruvchisi 5 qiymatni, shu tipdagi `length` o‘zgaruvchisi esa 7 qiymatni qabul qildi. Quyidagi 5-listingda o‘zgaruvchilarni aniqlashga oid misolni qaraymiz.

5-listing. O‘zgaruvchilarning qo‘llanishi.

```
1: # include < iostream. h >
2: int main()
3: {
4: int Bo‘yi=5, Eni=10, Yuzasi;
5: cout << “Bo‘yi:” <<Bo‘yi <<  “ \n”;
6: cout << “Eni:” << Eni << endl;
7: Yuzasi= Bo‘yi*Eni;
8: cout << “Yuzasi:” << Yuza << endl;
9: return 0;
}
```

NATIJA:

Bo‘yi: 5

Eni: 10

Yuzasi: 50

typedef kalitli so‘zi. unsigned short int kabi kalit so‘zlarni ko‘p martalab dasturda yozilishi zerikarli va diqqatvozlik talab qilganligi uchun TS/C++ tilida bunday tiplarni `typedef` kalitli so‘zi yordamida psevdonimini (taxallusini) tuzish imkoniyati berilgan. `typedef` so‘zi tipni aniqlash ma’nosini bildiradi.

Psevdonim tuzishda tipning nomi yangi tuzuladigan tip nomidan farqli bo‘lishi lozim. Bunda birinchi `typedef` kalitli so‘zi, keyin mavjud tip nomi, undan so‘ng esa yangi nom yoziladi. Masalan:

```
typedef unsigned short int  ushort
```

Bu satrdan so‘ng `ushort` nomli yangi tip hosil bo‘ladi va u qaerda `unsigned short int` tipidagi o‘zgaruvchini aniqlash lozim bulsa, shu joyda ishlatiladi.

6 - listing. **typedef** operatori orqali tiplarning aniqlanishi

```
1. # include <iostream.h>
2. typedef  insigned short int ushort
3. int main()
4. { ushort Bo‘yi = 5;
5. ushort Eni = 10;
6. ushort Yuzasi = Bo‘yi* Eni;
7. cout << “Yuzasi:” << Yuzasi << end;
8. }
```

NATIJA:

Yuzasi: 50

Belgilar. Belgili o‘zgaruvchilar odatda bir bayt joyni egallaydi va bu 256 xil belgini saqlash uchun yetarlidir. `Char` tipi qiymatlarini 0..255 sonlar to‘plamiga yoki `ASCII` belgilar to‘plamiga interpretatsiya qilish mumkin.

Maxsus belgilar. C++ kompilyatori matnlarni formatlovchi bir nechta maxsus belgilardan tashkil topgan. Ulardan eng ko‘p tarqalgani (3- jadvalda keltirilgan). Bu belgilarni dasturda ishlatishda «teskari slesh»dan foydalanamiz. Teskari sleshdan keyin boshqaruvchi belgi yoziladi. Masalan, tabulyatsiya belgiini dasturga qo‘yish uchun quyidagicha yozuvni yozish kerak.

```
Char tab ='\t';
```

Bu misoldagi `char` tipidagi o‘zgaruvchi `\t` qiymatini qabul qiladi. Maxsus belgilar axborotlarni ekranga, faylga va boshqa chiqarish qurilmalariga chiqarishda formatlash uchun qo‘llaniladi.

3-jadval.

Belgilar	Qiymati
\n	Yangi satrga o‘tish
\t	Tabulyatsiya
\b	Bitta pozitsiyaga o‘tish
\”	Ikkitalik qavscha
\’	Bittalik qavscha
\?	So‘roq belgisi
\\	Teskari slesh

O‘zgarmaslar. O‘zgaruvchilar kabi o‘zgarmaslar ham ma’lumotlarni saqlash uchun mo‘ljallangan xotira yacheykalarini o‘zida ifodalaydi. O‘zgaruvchilardan farqli ravishda ular dasturni bajarilishi jarayonida qiymati o‘zgarmaydi. O‘zgarmas e’lon qilinishi bilan unga qiymat berish lozim, keyinchalik bu qiymatni o‘zgartirib bo‘lmaydi. TS/C++ tilida ikki turdagi, literal va belgili o‘zgarmaslar aniqlangan.

Literal o‘zgarmaslar. Literalli o‘zgarmaslar to‘g‘ridan-to‘g‘ri dasturga kiritiladi. Masalan:

```
int myAge =39;
```

Bu ifodada `MyAge int` tipidagi o‘zgaruvchi, 39 soni esa literal o‘zgarmasdir.

Belgili o‘zgarmaslar. Belgili o‘zgarmas – bu nomga ega bo‘lgan o‘zgarmasdir. C++ tilida belgili o‘zgarmasni aniqlashning ikki usuli mavjud:

1. `# define` direktivasi yordamida o‘zgarmasni aniqlash.
2. `const` kalitli so‘zi orqali o‘zgarmasni aniqlash.

An’anaviy usul hisoblangan `#define` direktivasi orqali o‘zgarmasni aniqlashni quyidagi misolda ko‘rishimiz mumkin.

```
#define STAT 15
```

Bu holda STAT o‘zgarmas hech qanday tipga tegishli bo‘lmaydi.

Preprotsessor STAT so‘ziga duch kelganida uni `15` literaliga almashtiradi.

C++ tilida `#define` direktivasidan tashqari o‘zgarmasni aniqlashning nisbatan qulayroq bo‘lgan yangi usuli ham mavjud:

```
const unsigned short int STAT=15
```

Bu misolda ham belgili konstanta `STAT` nomi bilan aniqlanayapti va unga `unsigned short int` tipi berilyapti. Bu usul bir qancha imkoniyatlarga ega bo‘lib, u sizning dasturingizni keyingi himoyasini yengillashtiradi. Bu o‘zgarmasni oldingisidan eng muhim afzalligi uning tipga egaligidir.

Belgili o‘zgarmaslarni literal o‘zgarmaslarga nisbatan ishlatish qulayroqdir. Chunki agarda bir xil nomli literalli o‘zgaruvchini qiymatini o‘zgartirmoqchi bo‘lsangiz butun dastur bo‘yicha uni o‘zgartirishga to‘g‘ri keladi, belgili o‘zgarmaslarni esa faqatgina birining
qiymatini o‘zgartirish yetarli.

To‘plam o‘zgarmaslari. Bunday o‘zgarmaslarni hosil qilish uchun yangi berilgan ma’lumotlar tiplari tuziladi va undan so‘ng bu tipga tegishli o‘zgarmasli qiymatlar to‘plami bilan chegaralangan o‘zgaruvchilar aniqlanadi. Masalan, `RANG` nomli sanoqli tip deb e’lon qilaylik va uning uchun 5 ta **qizil, ko‘k, yashil, oq, qora** qiymatlarini aniqlaylik.

Sanoqli tiplarni hosil qilish uchun `enum` kalitli so‘zi va undan keyin tip nomi hamda figurali qavs ichida vergullar bilan ajratilgan o‘zgarmas qiymatlari ro‘yxati ishlatiladi. Masalan, `enum RANG { qizil, ko‘k, yashil,oq, qora};`

Bunda ifoda ikkita ishni bajaradi:

1. RANG nomli yangi sanoqli tip hosil qiladi;
2. Quyidagi belgili o‘zgarmaslarni aniqlaydi:

0 qiymat bilan qizil;
1 qiymat bilan ko‘k;
2 qiymat bilan yashil va hokazo;

Har bir sanoqli o‘zgarmas biror-bir aniqlangan butun qiymatga mos keladi.

Boshlang‘ich holatda o‘zgarmaslarga 0 dan boshlab qiymat beriladi. Lekin, ixtiyoriy o‘zgarmasga boshqa qiymatni o‘zlashtirish ham mumkin. Bunda ularga qiymat berish o‘sish tartibida bo‘lishi lozim. Masalan, **enum** RANG{qizil=100, ko‘k=200, yashil=300,oq, qora=500};
ko‘rinishda sanoqli tipni aniqlasak qizil o‘zgarmasi 100 ga, ko‘k-200 ga, yashil-300 ga, oq-301 ga, qora-500 ga teng bo‘ladi.

7-listing. Sanoqli o‘zgarmasni qo‘llanilishi.

```
1: # include <iostream.h >
2: int main( )
3: {
```

```
4:  enum Kunlar{Dushanba, Seshanba, Chorshanba,
5:  Payshanba, Juma, Shanba, Yakshanba}
6:  int tanlash;
7:  sout << “Kun nomerini kiriting (0-6):” ;
8:  sin << tanlash;
9:  if (tanlash=Yakshanba || tanlash = Shanba)
10:cout <<“\n Bugun Siz uchun dam olish kuni!”
11:<<endl;
12:else
13:cout << “\n Bugun Siz uchun ish kuni.\n”;
14:return 0;
15:};
```

HATIJA:

```
Kun nomerini kiriting (0-6): 6
Bugun Siz uchun dam olish kuni!
```

NAZORAT CAVOLLAR

1. Kompilyator va preprotsessorning farqi nimadan iborat?
2. #include direktivasi qanday vazifani bajaradi.
3. main() funksiyasining o‘ziga xos xususiyati
4. nimadan iborat?
5. Qanday izoh turlarini bilasiz va ular nima bilan
6. farq qiladi?
7. Izohlar bir necha qatorda yozilishi mumkinmi?
8. Nima uchun literalli o‘zgarmasga nisbatan belgili o‘zgarmasni ishlatish yaxshiroq?
9. Butun sonli va haqiqiy tiplarni qanday farqi bor?
10. unsigned short int va long int tiplarining o‘zaro farqi nimada?
11. const kalitli so‘zini #define direktivasi o‘rniga qo‘llashni afzalligi nimada?
12. Dastur ishiga “yaxshi” va “yomon” nomlangan o‘zgaruvchi qanday ta’sir qiladi?

3-MAVZU: DASTURLASH OPERATORLARI. SHARTLI OPERATORLAR

Reja

1. Ifodalar va operatorlar.
2. Mantiqiy amallar va munosabatlar
3. Funktsiyada ishlatiladigan operatorlar.
4. Tarmoqlanish operatorlari. Shartli operator
5. Takrorlanuvchi (tsiklik) algoritmlar va operatorlari.

Tayanch iboralar: Ifoda, operator, mantiqiy amallar va munosabatlar, funktsiya, tarmoqlanish operatorlari. shartli operator, takrorlanuvchi (tsiklik) algoritmlar va operatorlari

***Dars maqsadi*:** Talabalarda **C++** dasturlash tili operatorlari va ulardan foydalanish ko'nikmalarini hosil qilish.

1. Ifodalar va operatorlar.

Dastur biror bir aniqlangan ketma - ketlikda bajariluvchi komandalar to'plamidan iborat. Siz ushbu mavzudan quyidagilar bilan tanishasiz:

- Operator nima
- Blok nima
- Ifoda nima
- Dasturni

berilgan mantiqiy shartlarini bajarilishi natijasi asosida tarmoqlanishini qanday tashkil etish lozim.

- C++ dasturchisi nuqtai-nazarida "Rost" va " Yolg'on" nima?

Ifoda. C++ tilida ifodalar biror-bir hisoblash natijasini qaytaruvchi boshqa ifodalar ketma-ketligini boshqaradi yoki hech nima qilmaydi (nol ifodalar).

C++ tilida barcha ifodalar nuqtali vergul bilan yakunlanadi. Ifodaga misol qilib o'zlashtirish amalini olish mumkin.

O'zlashtirish operatori. O'zlashtirish operatori (=) o'zidan chap tomonda turgan operand qiymatini tenglik belgisidan o'ng tomondagilarni hisoblangan qiymatiga almashtiradi. Masalan:

```
x = a+b;
```

ifodasi `x` operandga `a` va `v` o'zgaruvchilarni qiymatlarini qo'shishdan hosil bo'lgan natijani o'zlashtiradi.

O'zlashtirish operatoridan chapda joylashgan operand adresli operand yoki `l-kiymat` (`chap`-chap so'zidan olingan) deyiladi. O'zlashtirish operatoridan o'ngda joylashgan operand operatsion operand yoki `r-kiymat` deyiladi.

O'zgarmaslar faqatgina `r-kiymat` bo'lishi mumkin va hech qachon adresli operand bo'la olmaydi, chunki dasturning bajarilishi jarayonida o'zgarmas qiymatini o'zgartirib bo'lmaydi.

35 = x // notugri!

l–qiymat esa r–qiymat bo'lishi mumkin.

Algebradan farqli ravishda bu ifoda `x=a+v` ga teng ekanligini anglatmaydi. Bu ifodani quyidagicha tushunish kerak:

a va v o‘zgaruvchilarni qiymatlarini yig‘ib natijasini `x` o‘zgaruvchiga beramiz yoki `x` o‘zgaruvchiga `a+v` qiymatni o‘zlashtiramiz. Bu ifoda birdaniga ikkita amalni bajaradi, yig‘indini hisoblaydi va natijani o‘zlashtiradi. Ifodadan so‘ng nuqtali vergul qo‘yiladi. `(=)` operatori o‘zidan chap tomondagi operandga o‘ng tomondagi operandlar ustida bajarilgan amallar natijasini o‘zlashtiradi.

Matematik operatorlar. C++ tilida `5` ta asosiy matematik operatorlar ko‘llaniladi: qo‘shish `(+)`,

ayirish `(-)`, ko‘paytirish `(*)`, butun songa bo‘lish `(\)` va modul bo‘yicha bo‘lish `(%)` (qoldiqni olish).

Ishorasiz butun sonlarni ayirishda, agarda natija manfiy son bo‘lsa g‘ayrioddiy natija beradi. Buni 8- listingdan ko‘rishimiz mumkin.

8-listing. Ayirish natijasida butun sonni to‘lib qolishiga misol

```
16: #include < iostream.h >
17: int main()
18: {unsiegned int ayirma
19: unsisgned int katta son = 100;
20: unsigned int kichik son = 50;
21: ayirma = katta son – kichik son;
22: cout<< “Ayirma“:<<ayirma<<“ga  teng\n”;
23: ayirma = kichik son – katta son ;
24: cout<<“Ayirma“:<<ayirma<<“ga teng\n”
25: <<endl;
26: return 0;
27: }
```

HATIJA:
Ayirma: 50 ga teng
Ayirma Ayirma: ga teng

Butun songa bo‘lish va qoldiqni olish operatorlari

Butun songa bo‘lish odatdagi bo‘lishdan farq qiladi. Butun songa bo‘lishdan hosil bo‘lgan bo‘linmaning faqatgina butun qismi olinadi. Masalan, 21 sonini 4 ga bo‘lsak 5 soni va 1 qoldiq hosil bo‘ladi. 5 butun songa bo‘lishni qiymati, 1 esa qoldiqni olish qiymati hisoblanadi.

Inkrement va dekrement. Dasturlarda o‘zgaruvchiga 1 ni qo‘shish va ayirish amallari juda ko‘p hollarda uchraydi. C++ tilida qiymatni 1 ga oshirish inkrement, 1 ga kamaytirish esa dekrement deyiladi. Bu amallar uchun maxsus operatorlar mavjuddir.

Inkrement operatori (++) o‘zgaruvchi qiymatini 1 ga oshiradi, dekrement operatori (—) esa, o‘zgaruvchi qiymatini 1 ga kamaytiradi. Masalan, s o‘zgaruvchisiga 1 qiymatni qo‘shmoqchi (oshirmoqchi) bo‘lsak quyidagi ifodani yozishimiz lozim.

C++ //s o‘zgaruvchi qiymatini 1 ga oshirdik.

Bu ifodani quyidagicha yozishimiz xam mumkin edi.

s=s+1;

Bu ifoda o‘z navbatida quyidagi ifodaga teng kuchli:

s+=1;

Prefiks va postfiks. Inkrement operatori ham, dekrement operatori ham ikki variantda ishlaydi: prefiksli va postfiksli. Prefiksli variantda ular o‘zgaruvchidan oldin (++Age), postfiksli variantda esa o‘zgaruvchidan keyin (Age++) yoziladi.

Oddiy ifodalarda bu variantlarni qo‘llanishida farq katta emas, lekin bir o‘zgaruvchiga boshqa o‘zgaruvchining qiymatini o‘zlashtirishda ularning qo‘llanilishi boshqacha xarakterga ega. Prefeksli operator qiymat o‘zlashtirilguncha, postfiksli operator esa qiymat o‘zlashtirilgandan keyin bajariladi. Buni quyidagi listingdan ko‘rishimiz mumkin:

Prefiksli va postfiksli operatorlarni qo‘llanilishi.

```
1:# include < iostream. h >
2:int main()
3:{
4:int myAge = 39;
5:int yourAge = 39;
6:cout <<"Men"<< MyAge <<"yoshdaman \n";
7:cout <<"Siz"<<yourAge<<"yoshdasiz \n";
8:myAge++ ; // postfiksli inkrement
9:++yourAge; // prefeksli inkrement
10:cout<< "Bir yil o'tdi …\ n";
11:cout<< "Men" << MyAge <<" yoshdaman\n";
12:cout<<"Siz"<<yourAge<<"yoshdasiz \n";
13:cout<< "Yana bir yil o'tdi\n";
14:cout<<"Men"<<myAge++<<"yoshdaman \n";
15:cout<<"Siz"<<++yourAge<<"yoshdasiz\n";
16:cout<< "Ma'lumotlarni qaytadan"
17:cout<< "chiqaraylik\n";
18:cout<<" Men"<<myAge<<"yoshdaman\n";
19:cout<<"Siz"<<yourAge<<"yoshdasiz \n";
20:return 0;
21:}
```

HATIJA:

```
Men 39 yoshdaman
Siz 39 yoshdasiz
Bir yil o'tdi …
Men 40 yoshdaman
Siz 40 yoshdasiz
Yana bir yil o'tdi …
Men 40 yoshdaman
Siz 41 yoshdasiz
Ma'lumotlarni qaytadan chiqaraylik
Men 41 yoshdaman
Siz 41 yoshdasiz
```

Operatorlar prioriteti. Murakkab ifodalarda qaysi amal birinchi navbatda bajariladi, qo‘shishmi yoki ko‘paytirishmi? Masalan:

```
x=5+3*8;
```

ifodada agarda birinchi qo‘shish bajarilsa natija 64 ga, agarda ko‘paytirish birinchi bajarilsa natija 29 ga teng bo‘ladi.

Har bir operator prioritet qiymatiga ega. Ko‘paytirish qo‘shishga nisbatan yuqoriroq prioritetga ega. Shuning uchun bu ifoda qiymati 29 ga teng bo‘ladi.

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

Demak, x=5+3+8*9+6*4

ifodada birinchi ko‘paytirish amallari chapdan o‘ngga qarab bajariladi 8*9=72 va 6*4=24. Keyin bu ifoda soddaroq ko‘rinish hosil qiladi.

```
x=5+3+72+24
```

Endi qo‘shishni ham xuddi shunday chapdan unga qarab bajaramiz:

```
5+3=8; 8+72= 80; 80+24=104;
```

Lekin, barcha operatorlar ham bu tartibga amal qilmaydi. Masalan, o‘zlashtirish operatori o‘ngdan chapga qarab bajariladi.

Ichki qavslar. Murakkab ifodalarni tuzishda ichki qavslardan foydalaniladi. Masalan, sizga sekundlarning umumiy soni keyin esa barcha qaralayotgan odamlar soni, undan keyin esa ularning ko‘paytmasini hisoblash kerak bo‘lsin.

```
TotalPersonSeconds=((NumMinutesToThink+
NumMinutesToType)*60*(PeopleInTheOffice+ PeopleOnVocation ))
```

Bu ifoda quyidagicha bajariladi. Oldin NumMinutesToThink o‘zgaruvchisining qiymati NumMinutesToType o‘zgaruvchisi qiymatiga qo‘shiladi. Keyin esa hosil qilingan yig‘indi 60 ga ko‘paytiriladi. Bundan keyin PeopleInTheOffice o‘zgaruvchi qiymati PeopleOnVocation qiymatiga qo‘shiladi. Keyin esa sekundlar soni kishilar soniga ko‘paytiriladi.

Bo‘sh joy (probel) belgisi. Bo‘sh joy belgilariga nafaqat probel, balki yangi satrga o‘tish va tabulyatsiya belgilari ham kiradi. Yuqorida keltirilgan ifodani quyidagicha ham yozish mumkin:

```
x= a+b   ;
```

Bu variantda keltirilgan ifoda ko‘rimsiz va tushunarsiz bo‘lsa ham to‘g‘ridir.

Bo‘sh joy belgilari dasturning o‘qilishliligini ta’minlaydi.

Blok va kompleks ifodalar. Ba’zan dastur tushunarli bo‘lishi uchun o‘zaro mantiqiy bog‘langan ifodalarni blok deb ataluvchi komplekslarga birlashtirish qulaydir. Blok ochiluvchi figurali qavs ({) bilan boshlanadi va yopiluvchi figurali qavs (}) bilan tugaydi. Blok ochilganda va yopilganda nuqtali vergul qo‘yilmaydi.

```
.....
{
 temp= a;
 a = b;
 b = temp;
}
```

Bu blok xuddi bir ifodadek bajariladi, u a va v o‘zgaruvchilar qiymatlarini almashtiradi.

Mantiqiy amallar va munosabatlar

Dasturlashda bir emas balki bir nechta shartli ifodalarni tekshirish zaruriyati juda ko‘p uchraydi. Masalan, x o‘zgaruvchisi y o‘zgaruvchisidan, y esa o‘z navbatida z o‘zgaruvchisidan kattami sharti bunga misol bo‘la oladi. Bizning dasturimiz mos amalni bajarishdan oldin bu ikkala shart rost yoki yolg‘onligini tekshirishi lozim.

Quyidagi mantiq asosida yuqori darajada tashkil qilingan signalizatsiya sistemasini tasavvur qiling. Agarda eshikda signalizatsiya o‘rnatilgan bo‘lsa VA kun vaqti kech soat olti VA bugun bayram YoKI dam olish kuni BO‘LMASA politsiya chaqirilsin. Barcha shartlarni tekshirish uchun C++ tilida uchta mantiqiy operatori ishlatiladi.

Ular 4-jadvalda keltirilgan. Mantiqiy operatorlar (4-jadval).

4 – jadval

Operator	Belgi	Misol
VA	&&	1ifoda && 2ifoda
YoKI	||	1ifoda||2ifoda
INKOR	!	!ifoda

Mantiqiy ko‘paytirish operatori

Mantiqiy ko‘paytirish operatori ikkita ifodani hisoblaydi, agar ikkala ifoda true qiymat qaytarsa VA operatori ham true qiymat qaytardi. Agarda sizning qorningiz ochligi rost bo‘lsa VA sizda pul borligi ham rost bo‘lsa siz supermarketga borishingiz va u yerdan o‘zingizga tushlik qilish uchun biror bir narsa xarid qilishingiz mumkin. Yoki yana bir misol, masalan, if(x==5)&&(y==5) mantiqiy ifodasi agarda x va u

o‘zgaruvchilarini ikkalasining ham qiymatlari 5 ga teng bo‘lsagina true qiymat qaytaradi. Bu ifoda agarda o‘zgaruvchilardan birortasi 5 ga teng bo‘lmagan qiymat qabul qilsa false qiymatini qaytaradi. Mantiqiy ko‘paytirish operatori faqatgina o‘zining ikkala ifodasi ham rost bo‘lsagina true qiymat qaytaradi.

Mantiqiy ko‘paytirish operatori && belgi orqali belgilanadi.

Mantiqiy qo‘shish operatori

Mantiqiy qo‘shish operatori ham ikkita ifoda orqali hisoblanadi. Agarda ulardan birortasi rost bo‘lsa mantiqiy qo‘shish operatori true qiymat qaytaradi. Agarda sizda pul YoKI kredit kartochkasi bo‘lsa, siz schotni to‘lay olasiz. Bu holda ikkita shartning birdaniga bajarilishi: pulga ham va kredit kartochkasiga ham ega

bo‘lishingiz shart emas. Sizga ulardan birini bajarilishi yetarli. Bu operatorga oid yana bir misolni qaraymiz. Masalan,

if(x==5)||(y==5)

ifodasi yoki x o‘zgaruvchi qiymati, yoki u o‘zgaruvchi qiymati, yoki ikkala o‘zgaruvchining qiymati ham 5 ga teng bo‘lsa rost qiymat qaytaradi.

Mantiqiy inkor operatori

Mantiqiy inkor operatori tekshirilayotgan ifoda yolg‘on bo‘lsa true qiymat qaytaradi. Agarda tekshirilayotgan ifoda rost bo‘lsa inkor operatori false qiymat qaytaradi. Masalan, (if!(x==5))
ifodasining qiymati, agarda x o‘zgaruvchisi 5 ga teng bo‘lmasa true qiymat qaytaradi. Bu ifodani boshqacha ham yozish mumkin: if(x!=5)

Amallar (Operatsiyalar)

Bajarilishi natijasida biror bir qiymat qaytaradigan barcha ifodalar TS/C++ tilida amallar deyiladi. Amallar albatta biror bir qiymat qaytaradi. Masalan, 3+2 amali 5 qiymatni qaytaradi.

Operatorlar. Operator - bu qandaydir amalni bajarish tug‘risida kompilyatorga uzatiladigan literaldir. Operatorlar operandlarga ta’sir qiladi. TS/C++ da operandlar deb alohida literallar va butun ifodalar tushuniladi.

TS/C++ tilida ikki ko‘rinishdagi operatorlar bor:

- o‘zlashtirish operatorlari
- matematik operatorlar

Munosabat operatorlari. Bunday operatorlar ikkita qiymatni teng yoki teng emasligini aniqlash uchun ishlatiladi. Taqqoslash ifodasi doimo true (rost) yoki false (yolg‘on) qiymat qaytaradi. Munosabat operatorlarining ko‘llanilishiga oid misol 4.1. jadvalda keltirilgan.

Munosabat operatorlari.

4.1-jadval

Nomi	Operator	Misol	Qaytaradigan qiymat
Tenglik	==	100==50 50==50	false true
Teng emas	!=	100!=50 50!=50	true false
Katta	>	100>50 50>50	true false
Katta yoki teng	>=	100>=50 50>=50	true true
Kichik	<	100<50 50<50	true false
Kichik yoki teng	<=	100<=50 50<=50	true true

```
Mantiqiy solishtirish operatorlari */
  . . .
if (s1 == s2) " teng "
if (s1 < s2) " kichik
if (s1 >= s2) " katta yoki teng
if (s1 != s2) " teng emas "
. . . .
return (0);
}
 . . . .
```

Bu yerda true ni o‘rniga 1, false ni qiymati o‘rniga 0 ni qo‘llashimiz mumkin. Boshqa misol: while (g<10 || f<4){ ... } Bu yerda ikki o‘zgaruvchi bor (g va f). Birinchisi 10 dan kichkina yoki ikkinchisi 4 dan kichkina bo‘lganda while ning tanasitakrorlanaveradi, ya’ni shart bajarilishi uchun eng kamida bitta true bo‘lishi kerak, AND da (&&) esa xamma oddiy shartlar true bo‘lishi kerak.

AND(&&) ning jadvali

ifoda1	ifoda2	ifoda1&&ifoda2
false (0)	false (0)	false (0)
true (1)	false (0)	false (0)
false (0)	true (1)	false (0)
true (1)	true (1)	true (1)

OR (||) ning jadvali

ifoda1	ifoda2	ifoda1&&ifoda2
false (0)	false (0)	false (0)
true (1)	false (0)	true (1)
false (0)	true (1)	true (1)
true (1)	true (1)	true (1)

NOT (!) ning jadvali

ifoda	!(ifoda)
false (0)	true (1)
true (1)	false (0)

- && va || operatorlari ikkita argument olishadi.
- Bulardan farqli o‘laroq, ! (mantiqiy inkor) operatori bitta argument oladi va bu argumentlardan oldin qo‘yiladi.
- Inkor operatori ifodaning mantiqiy qiymatini teskarisiga o‘zgartiradi, ya’ni false ni true deb beradi , true ni esa false deb beradi. if ifodasi malum bir shartning to‘g‘ri (true) yoki noto‘g‘ri (false) bo‘lishiga qarab, dasturning u yoki bu blokini bajarishga imkon beradi. Agar shart to‘g‘ri bo‘lsa, if dan so‘ng keluvchi amal bajariladi. Agar shart bajarilmasa, u holda if tanasidagi ifoda bajarilmay, if dan so‘ng keluvchi ifodalar ijrosi davom ettiriladi. Bu strukturaning ko‘rinishi quyidagichadir: switch strukturasi.

if-else-if yordami bilan bir necha shartni test qilishimiz mumkin. Lekin bunday yozuv nisbatan o‘qishga qiyin va ko‘rinishi qo‘pol bo‘ladi. Agar shart ifoda butun son toifaga mansub bo‘lsa, yoki bu toifaga keltirishi mumkin bo‘lsa, biz switch (tanlash) ifodalarini ishlata olamiz. switch strukturasi bir necha case xizmatchi so‘zlaridan (label) va majburiy bo‘lmagan default so‘zlaridan iboratdir. Etiket – bu bir nomdir. U dasturning bir nuqtasiga qo‘yiladi. Dasturning boshqa yeridan ushbu etiketga o‘tishni bajarish mumkin. O‘tish yoki sakrash goto bilan amalga oshiriladi, switch blokida xam qo‘llasa bo‘ladi.

Funksiyada ishlatiladigan operatorlar. Funksiyada ishlatiladigan operatorlar soni va turiga hech qanday chegara yo‘q. Funksiya ichida istalgancha boshqa

funksiyalarni chaqirish mumkin bo‘lsada, lekin funksiyalarni aniqlash mumkin emas.

C++ tilida funksiyaning hajmiga hech qanday talab qo‘yilmasa ham uni kichikroq tarzda tuzgan ma’quldir. Har bir funksiya tushunish oson bo‘ladigan bitta masalani bajarishi lozim. Agarda funksiya hajmi kattalashayotganligini sezsangiz, bu funksiyani bir nechta kichikroq funksiyalarga ajratish haqida o‘ylashingiz kerak.

Funksiya argumentlari. Funksiyaning argumentlari turli tipda bo‘lishi mumkin. Shuningdek, argument sifatida C++ tilidagi biror bir qiymat qaytaradigan o‘zgarmaslar, matematik va mantiqiy ifodalar va boshqa ixtiyoriy funksiyalarni berish mumkin.

Misol sifatida bizda biror bir qiymat qaytaruvchi `double()`, triple(), square() va cube() funksiyalari berilgan bo‘lsin. Biz quyidagi ifodani yozishimiz mumkin:

```
Answer=(double(triple(square(my Value))))
```

Bu ifodada `myValue` o‘zgaruvchisini qabul qiladi va u `cube()` funksiyasiga argument sifatida uzatiladi.

`cube()` funksiyasi qaytargan qiymat esa `square()` funksiyasiga argument cifatida uzatiladi. `square()` funksiyasi qiymat qaytargandan keyin, buning qiymati o‘z navbatida `triple()` fuknsiyasiga argument sifatida beriladi. `triple()` funksiyasining qiymati esa `double()` funksiyasiga argument qilib beriladi va u `Answer` o‘zgaruvchisiga o‘zlashtiriladi.

Parametrlar bu lokal o‘zgaruvchilardir. Funksiyaga uzatilgan har bir argument, bu funksiyaga nisbatan lokal munosabatda bo‘ladi. Funksiyani bajarilishi jarayonida argumentlar ustida bajarilgan o‘zgartirishlar funksiyaga qiymat sifatida berilgan o‘zgaruvchilarga ta’sir qilmaydi. Bu fikr ushbu dasturda namoyish qilingan.

Funksiyaga parametrlarni qiymat sifatida uzatish.

```
1. // Funksiyaga parametrlarni kiymat sifatida
3. // uzatish
4. #include <iostream.h>
5. void Almashtirish (int x, int y);
6. int main()
7. { int x = 5,  y =10;
8. cout << “Main().Almashtirishdan oldin,x:”
9. <<x <<”  y:”  << y <<“\n”;
10.Almashtirish (x,y);
11.cout<<“Main().Almashtirishdan keyin, x:”
12.<<x<< “y:” <<y<< “\n”;
13.return 0;
14.}
15.void Almashtirish (int x, int y)
16.{
17.int temp;
```

```
18.cout<<“Almashtirish(). Almashtirishdan ”<<
19.“oldin, x: ”<<x<<“ y: ”<<y<< “\n”;
20.temp = x ;
21.x = y;
22.y = temp;
23.cout<<“ Almashtirish ().Almashtirishdan ”<<
24.“keyin, x: ”<<x<<“ y: ”<<y<< “\n”;
25.}
```

HATIJA:

```
Main().Almashtirishdan oldin,x: 5 y:10
Almashtirish().Almashtirishdan oldin, x:5 y:10
Almashtirish().Almashtirishdan keyin, x:10 y:5
Main().Almashtirishdan keyin, x:5 y:10 x:
```

Funksiyaning qaytaradigan qiymatlari

Funksiya yo biror bir real qiymatni, yo kompilyatorga hech qanday qiymat qaytarilmasligi haqida xabar beruvchi void tipidagi qiymatni qaytaradi.

Funksiyani qiymat qaytarishi uchun return kalitli so‘zidan foydalaniladi. Bunda oldin return kalitli so‘zi, keyin esa qaytariladigan qiymat yoziladi. Qiymat sifatida esa o‘zgarmaslar kabi butun bir ifodalarni ham berish mumkin. Masalan:

```
return 5 ;
return (x > 5) ;
return (MyFunction()) ;
```

MyFunction() funksiyasi biror bir qiymat qaytarishidan kelib chiqsak, yuqoridagi barcha ifodalar to‘g‘ri keltirilgan. return(x>5) ifodasi esa x 5 dan katta bo‘lsa true, kichik yoki teng bo‘lsa false mantiqiy qiymatini qaytaradi.

Agarda funksiyada return kalit so‘zi uchrasa undan keyingi ifoda bajariladi va uning natijasi funksiya chaqirilgan joyga uzatiladi. return operatori bajarilgandan keyin dastur funksiya chaqirilgan satrdan keyingi ifodaga o‘tadi. return kalitli so‘zidan keyingi funksiya tanasidagi operatorlar bajarilmaydi.

Tarmoqlanish operatorlari. Shartli operator

Agar dasturning bajarilishi davomida buyruqlar ketma-ketligi biror shartga asosan o‘zgarsa, bunday hollarda tarmoqlanish jarayonini tashkil etadigan operatorlardan foydalaniladi. Shartli operatorning umumiy strukturasi quyidagicha:

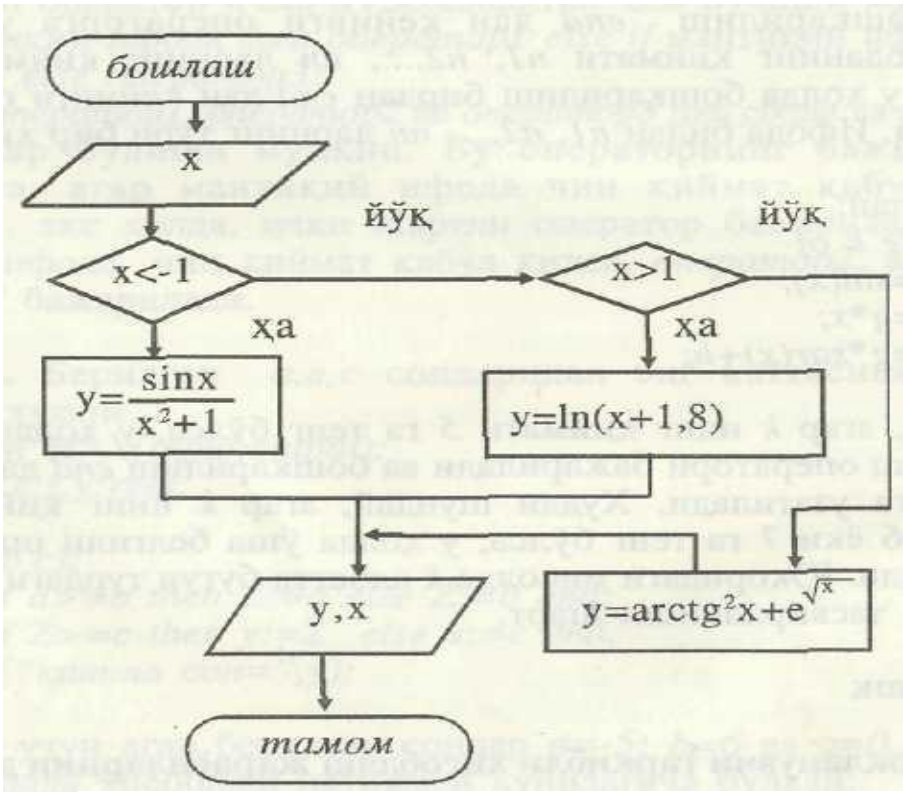
```
if (mantiqiy ifoda)
operator_1;
else
operator_2;
```

Bu yerda else qismi qoldirilsa xam bo‘ladi, lekin ushbu ko‘rinishni ishlashi oldin qavs ichidagi mantiqiy ifoda ya’ni shart bajariladi. Agar qavs ichidagi shart rost bo‘lsa, u holda birinchi operator bajariladi, aks holda operator 2 bajariladi.

1. Argument *x* ning ixtiyoriy qiymatida ushbu funksiyaning qiymatini hisoblash algoritmi tuzilsin.

$$y = \begin{cases} \dfrac{\sin x}{x^2+1}, & agar \quad x < -1 \\ arctg^2 x + e^{\sqrt{x}}, & agar \quad -1 \le x \le 1 \\ \ln(x+1{,}8), & agar \quad x > 1 \end{cases}$$

Algoritmi: grafik ko‘rinishi (blok-sxema)



Shartli o‘tish operatorlarini qo‘llash. Masalan: Ikkita sonni yig‘indisi uchinchi sondan katta bo‘lsa, s=1, aks holda s=0 deb olish dasturini tuzing.

Dasturi C++ tilida:

```
#include <iostream>
#include <math.h>
using namespace std;
void main()
{
        double a,b,c;
        unsigned int s;
        cout<<"3 ta sonni kiriting : \n";
        cout<<"a= ";cin>>a;cout<<"\n";
        cout<<"b= ";cin>>b;cout<<"\n";
        cout<<"c= ";cin>>c;cout<<"\n";
        if (a+b>c)
```

```
        {
                s=1;
        }
        else    s=0;
        cout<<"s= "<<s;
        getchar();
}
```

E’tibor berish kerakki, shart qavs ichida yoziladi. Bajarilishi: agar mantiqiy ifoda natijasi **rost** bo‘lsa, keltirilgan operator bajariladi va keyingi satrga o‘tiladi. Agar shart natijasi yolg‘on false bo‘lsa, ifoda bajarilmasdan keyingi satrga o‘tadi. Ko‘pincha bunday hollarda 2 ta operator aralashib ketmasligi uchun shartsiz o‘tish operatori – goto n ishlatiladi. Bu yerda n – nishon bo‘lib, u harflar, sonlar yoki xarfsonlar bo‘lishi mumkin. Nishon operatordan ikki nuqta belgisi bilan ajratiladi.

Yuqoridagi dastur TS tilida:

```
# include <stdio.h>
# include <math.h>
  main()
{
float a,b,c; clrscr ();
printf(“shartni tekshirish \n”);
printf(“ agar a+b>c bo‘lsa natija: ”);
scanf(“%f “, &a,&b,&c);
if (a+b>c)
printf(“ s=1 \n\n”);
else
printf(“ s=0\n\n”);
getch ();
}
```

Misol . Ushbu berilgan sistemani shartli operatordan foydalangan holda dasturini tuzing.

$$Z=\begin{cases}\dfrac{x}{2+x^2}-\sqrt{x}\ldots agar\ldots x>3\\[2ex](\dfrac{\ln x}{x})^3\ldots\ldots agar\ldots x\le 3\end{cases}$$

```
# include <stdio.h>
# include <math.h>
        main()
{ float x,z; clrscr ();
printf(“ x ga qiymat kiriting\n”);
scanf(“%f “, &x);
if (x>3)
z=x/(2+sqr(x))-sqrt(x);
```

```
else
z=pow((log(x)/x),3);
printf(“ z yig‘indi=%f \n\n”,z);
getch (); }
```

Ushbu dasturni ko‘rib chiqamiz:

```
# include <stdio.h>
# include <conio.h>
# include <math.h>"
main()
{
float x,y; clrscr();
printf (“X o‘zgaruvchisining qiymatini kiriting”);
scanf (“%f”, &X);
if(x < 5)
{ y=sin(x); goto a;}
y=pow(x, 2/3.);
a:printf (“x=%f y=%f\n”, x,y);
}
```

Dasturga izoh:

1. Dasturda standart funksiyalarni hisoblash kerak bo‘lgani uchun qo‘shimcha math.h bibliotekasi yuklandi.
2. x va y o‘zgaruvchilari haqiqiy toifaga ega va x o‘zgaruvchisining qiymatini klaviatura orqali kiritamiz (ixtiyoriy son).
3. Agar x ning qiymati 5 dan kichik bo‘lsa, y=sin(x) funksiyasi hisoblanadi va jarayon natijani olish operatoriga uzatadi. Bu yerda 2 ta operator y=sin(x) va y=pow(x,2/3); aralashib ketmasligi uchun ular o‘rtasiga goto a operatori yoziladi (a-nishon) va ular birga kelgani uchun ularni alohida {} belgilar orasiga olinadi.
4. Natija qilib x va y lar olinadi, ular haqiqiy toifaga mansub bo‘lganligi uchun f format ishlatiladi. Natijada kursor 1 qator pastga tushadi (\n).

Masalan: Yuqoridagi misolni ko‘radigan bo‘lsak, x<5? (y=sin(x)): (y=pow(x, 2/3.)); ko‘rinishda bo‘ladi. Bu ko‘rinishda shart qavslarga olinmaydi, operatorlar esa qavs ichida yoziladi.

```
. . .
    switch (ifoda yoki o‘zgaruvchi);
        {
        case 1- qiymat: operator (lar); break;
        case 2- qiymat: operator (lar); break;
        . . .
        case n- qiymat: operator (lar); break;
        default: o‘zgaruvchi yuqoridagi qiymatlardan boshqa qiymatni qabul
qilganidagi operator (lar);
        }
```

Bu operatorni tanlash operatori deb ataladi. **switch** yonidagi ifoda (ifoda yoki o‘zgaruvchi) butun toifali bo‘lishi kerak. Case operatori esa o‘zgaruvchi yoki ifodaning qabul qilishi mumkin. Bunda mavjud bo‘lgan qiymatlari tanlanadi va mos

satrdagi funksiya hisoblanadi. Masalan: quyidagi ifoda uchun dastur tuzish kerak bo‘lsin.

```
#include<stdio.h>
# include <conio.h>
#include<math.h>
main()
        {
        int x; float y; clrscr();
        printf(“x o’zgaruvchisining qiymatini kiriting”);
        scanf(“%d”, &x);
        switch(x)
        {
        case1 :y=sin(x); break;
        case2 :y=cos(x); break;
        case3 :y=tan(x); break;
        default :y=pow(x, 1/2.);
        }
        printf(“x=%d y=%g\d”, x,y);
        }
```

Nima uchun goto operatorini ishlatmaslik kerak

Shartsiz goto operatori orqali dasturning ixtiyoriy nuqtasiga borish mumkin. Lekin goto operatorining tartibsiz qo‘llanilishi bu dasturni umuman tushunarsiz bulishiga olib keladi. Shuning uchun oxirgi 20 yillikda butun jahon bo‘yicha dasturlashni o‘rganuvchilarga qo‘yidagi fikr ta’kidlanib kelinmokda “Hech qachon goto operatorini ishlatmang’’.

goto operatorining o‘rnini bir muncha mukammalroq strukturaga ega bo‘lgan konstruksiyalar egalladi. Bular for, while va do…while operatorlari bo‘lib, ular goto operatoriga nisbatan ko‘prok imkoniyatlarga egadir. Lekin dasturlashda har qanday instrument to‘g‘ri qo‘llanilgandagina foydali bo‘lishi hisobga olinib ANSI komiteti C++ tilida goto operatorini qoldirishga qaror qildi. Albatta, bu bilan quyidagi hazil fikr ham paydo bo‘ldi: “Bolalar! Bu operatordan uy sharoitida foydalanish zararsizdir’’.

goto operatori tarixi

Dasturlashni ilk davrlarida kichikroq hajmdagi va yetarlicha sodda dasturlar ishlatilar edi. Bunday dasturlarda sikllar nishonlardan, operatorlar va komandalar ketma – ketligidan hamda o‘tish operatoridan iborat edi.

C++ tilida nishon deb orqasidan ikki nuqta (:) yoziladigan identifikatorga aytiladi. Nishon doimo boshqaruv o‘tishi lozim bo‘lgan operatordan oldin o‘rnatiladi. Kerakli nishonga o‘tish uchun goto operatori qo‘llaniladi.

Bunda kalit so‘zdan keyin nishon nomi yoziladi. goto operatoriga misol ushbu dasturda keltirilgan.

goto operatori yordamida sikl tashkil etish

```
1.include  <iostream.h>
2. int main()
3.  {
4.  int counter=0; // cchyotchikni initsializatsiya qilish
5.  loop:
6.  counter ++ ;   //  siklni boshlanishi
7.  sout <<“counter: ” <<counter<< “\n”;
8.  if(counter <s) //kiymatni tekshirish
9.  goto loop;  //sikl boshiga qaytish
10.sout<<“T sikl tugadi.
11.sounter:”<<counter<<endl;
12.return 0;
13.}
```

NATIJA:

```
counter : 1
counter : 2
counter : 3
counter : 4
counter : 5
T sikl tugadi.Counter: 5.
```

4-MAVZU: C++ DASTURLASH TILIDA TAKRORLANUVCHI JARAYONLAR

Reja:

1. C++ tilida for operatori.
2. C++ tilida while operatori.
3. C++ tilida do while operatori.
4. C++ tilida break va continue operatorlari.
5. C++ tilida goto operatori.

1. Dastur bajarilishini boshqarishning yana bir kuchli mexanizmlaridan biri – takrorlash operatorlari hisoblanadi.

Takrorlash operatori takrorlash sharti deb nomlanuvchi mantiqiy ifodani rost (true) qiymatida dasturning ma‘lum bir qnomlaridagi operatorlarni (takrorlash tanasini) ko‗p marta takror ravishda bajarishni amalga oshiradi (itaratsiya). Takrorlash o‗zining kirish va chiqish nuqtalariga ega, lekin chiqish nuqtasining bo‗lishi shart emas. Oxirgi holda takrorlashga cheksiz takrorlash deyiladi. Cheksiz takrorlash uchun takrorlashni davom ettirish sharti doimo rost bo‗ladi.

Takrorlash shartini tekshirish takrorlash tanasidagi operatorlarini bajarishdan oldin tekshirilishi mumkin (for, while takrorlashlari) yoki takrorlash tanasidagi operatorlarini har bir bajargandan keyin tekshirilishi mumkin (do-while).

Takrorlash operatorlari ixtiyoriy ravishda ichma-ich joylashgan bo‗lishi mumkin.

for takrorlash operatori for takrorlash
operatorining sintaksisi qo‗yidagi ko‗rinishga
ega: for (<ifoda >1; <ifoda>2;<ifoda>3)
<operator yoki blok>;

Bu operator amal qilishni <ifoda >1 bajarishdan boshlaydi. Keyin takrorlash qadamlari boshlanadi. Har bir qadamda <ifoda>2 bajariladi, agar natija rost (true) bo‗lsa, takrorlash tanasi - <operator> bajariladi va oxirida <ifoda>3 bajariladi, aks holda boshqaruv takrorlash operatoridan keyingi operatorga o‗tiladi. Takrorlash tanasi – <operator yoki blok> sifatida bitta operator yoki operatorlar bloki kelishi mumkin.

Misol uchun 10 dan 20 gacha bo‗lgan butun sonlar yig‗indisini hisoblash masalasini ko’raylik.

```
#include
<iostream.h>
int main() {
int Summa=0;
for (int i=10; i<=20; i++)
Summa +=i;
sout<<‖Yig\‘indi=‖ <<Summa;
return 0; }
```

Dasturdagi takrorlash operatori o’z ishini takrorlash parametri i (takrorlash sanagichi) o’zgaruvchisiga boshlang_ich qiymat – 10 berishdan boshlaydi va har bir takrorlash qadamidan (itaratsiyadan) keyin uning qiymati bittaga oshadi (qavs ichidagi uchinchi operator bajarilishi hisobiga). Har bir takrorlash qadamida takrorlash tanasidagi operator bajariladi, ya‘ni Summa o_zgaruvchisiga i qiymati qo_shiladi.

Takrorlash sanagichi i qiymati 21 bo_lganda i<=20 takrorlash sharti yolg_on bo_ladi va takrorlash tugaydi va boshqaruv takrorlash operatoridan keyingi cout operatoriga o_tiladi va ekranga yig_indi chop etiladi.

Yuqorida keltirilgan misolga qarab takrorlash operatorlarining qavs ichidagi ifodalariga izoh berish mumkin:

<ifoda>1 – takrorlash sanagichi vazifasini bajaruvchi o_zgaruvchisiga boshlag_ich qiymat berishga xizmat qiladi;

<ifoda>2 – takrorlashni bajarish yoki yo_qligini aniqlab beruvchi mantiqiy ifoda, agar shart rost bo_lsa, takrorlash davom etadi, aks holda yo_q;

<ifoda>3 – odatda takrorlash sanagichi qiymatini oshirish (kamaytirish) uchun xizmat qiladi yoki bu yerda takrorlash shartiga ta‘sir qiluvchi boshqa amallar bo_lishi mumkin.

Takrorlash operatorila qavs ichidagi ifodalar bo_lmasiligi mumkin, lekin sintaksis _;‘ bo_lmasligiga ruxsat bermaydi. Shu sababli, eng sodda ko_rinishdagi takrorlash operatori quyidagicha bo_ladi:

```
for ( ; ; ) cout <<‖Cheksiz takrorlash…‖ ;
```

Agar takrorlash jarayonida bir nechta o_zgaruvchilarning qiymati sinxron ravishda o_zgarishi kerak bo_lsa, <ifoda>1 va <ifoda>3 ifodalarida bir nechta operatorlarni _,‘ bilan yozish orqali bunga erishish mumkin:

```
for (int i=10, j=2; i<=20;  i++,  j=i+10)
 {
   …
 };
```

Takror operatorining har bir qadamida j va i qiymatlari mos ravishda o_zgarib boradi.

for operatorida takrorlash tanasi bo_lmasligi ham mumkin. Masalan, dastur bajarilishini ma‘lum bir muddatga «to_xtatib» turish zarur bo_lsa, takrorlashni hech qanday qo_shimcha ishlarni bajarmasdan amalga oshirish orqali erishish mumkin:

```
#include
<iostream.h>
int main() {
int delay;
 . . .
 for (delay =500; delay>0; delay--)
 ; // bo_sh operator
 . . .
 return 0;
}
```

Yuqorida keltirilgan 10 dan 20 gacha bo_lgan sonlar yig_indisini bo_sh tanali takrorlash operatori orqali hisoblash mumkin:

```
. . .
for (int i=10; i<=20; Summa +=i++);
      . . .
```

Takrorlash operatori tanasi sifatida operatorlar blokini ishlatish faktorialni hisoblash misolida ko_rsatish mumkin. Aytish zarurki, faktorialni hisoblashning bu yo_lini samarali deb bo_lmaydi. #include <iostream.h> int main() {

```
int a;
unsigned long fact=1;
cout <<‖Butun sonni
kiriting:_ ‖; cin >>a;
if ((a>=0)&&(a<33))
 {
  for (int i=1; i<=a; i++)
   {
     if (a!=0)  fact
*=i;      else
fact=1;
   }
   cout <<a<<‖!= —<<fact<<‘\n‘;
  }
ret
urn
0;
}
```

Dastur foydalanuvchi tomonidan 0 dan 33 gacha oraliqdagi son kiritilganda amal qiladi.

while takrorlash operatori

while takrorlash operatori operator yoki blokni takrorlash sharti yolg_on (false) bo_lguncha takror bajaradi. U quyidagi sintaksisga ega:

while (<ifoda>) <operator yoki blok>;

Agar <ifoda> rost qiymatli konstanta ifoda bo_lsa, takrorlash cheksiz bo_ladi. Xuddi shunday, <ifoda> takrorlash boshlanishida rost bo_lib, uning qiymatiga takrorlash tanasidagi hisoblash ta‘sir etmasa, ya‘ni uning qiymati o_zgarmasa, takrorlash cheksiz bo_ladi. while takrorlash shartini oldindan tekshiruvchi takrorlash operatori hisoblanadi. Agar takrorlash boshida <ifoda> yolg_on bo_lsa while operatori bajarilmasdan cheklab o_tiladi.

Ayrim hollarda <ifoda> qiymat berish operatori ko_rinishida kelishi mumkin. Bunda qiymat berish amali bajariladi va natija 0 bilan solishtiriladi (0 – yolg_on). Natija noldan farqli bo_lsa, takrorlash davom ettiriladi.

Xuddi for operatoridek, _‘ yordamida <ifoda> da bir nechta amallar sinxron ravishda bajarish mumkin. Masalan, son va uning kvadratlarini chop qiladigan dasturda shu holat ko_rsatilagan:

```
#include <iostream.h>
int main() {  int n,n2;
cout<<‖Sonni
kiriting(0..10):_‖;
cin>>n; n+=1;
while(n-=1, n2=n*n, n>0)
 cout<<‖ n=‖<<n<<‖  n^2=‖<<n2;

retu
rn
0; }
```

Dastur n sonini kamayish tartibida 1 gacha, n soni va uning kvadratini chop qiladi. Shunga e‘tibor berish kerakki, shart ifodasida operatorlarni yozilish ketma–ketligining ahamiyati bor, chunki, eng oxirgi operator takrorlash sharti hisoblanadi. Dasturda n qiymati 0 bo_lganda takrorlash tugaydi.

Quyida keltirilgan dasturda berilgan o_nlik sonning ikkilik ko_rinishi chop qilish masalasini yechishda while operatoridan qo_llash ko_rsatilgan.

```
#include <iosteam.h>
int
main
() {
  int sanagich
=4;    short
son10,
jarayon=1;
  while (jarayon) // cheksiz takrorlash
   {
     cout <<‖O‘nlik sonni
kiriting(0..15)_‖;      cin >>son10;
     cout<<‘/n‘<<son10<<‖ sonining ikkilik
ko‘rinishi: ‖;      while (sanagich)
      {
        if (son10 & 8)       // son102
&   00001000                  cout<<1;
else cout <<0;
      son10=son10<<1 // razradlarni chapga 1 pozitsiyaga surish
      sanagich□□;
     }
cout <<‘\n‘;
cout<<‖Jarayonni to‘xtatish (0), davom
ettirish (1):_ ‖; cin >> jarayon;
```

```
sanagich=4;
} return 0; }
```

Dasturda ichma–ich joylashgan takrorlash operatorlari ishlatilgan. Birinchisi, sonning ikkilik ko‗rinishini chop qilish jarayonini davom ettirish sharti bo‗yicha amal qiladi. Ichki joylashgan ikkinchi takrorlash operatorida har qanday 0 dan 15 bo‗lgan sonlar to‗rtta razryadli ikkilik son ko‗rinishida bo‗lishiga asoslangan holda, kiritilgan sonning ichki, ikkilik ko‗rinishida to‗rtinchi razryadda 0 yoki 1 turganligi aniqlanada (―son10 & 8‖) va natija 1 (rost) bo‗lsa 1, aks holda 0 chop etiladi. Keyingi qadamda sonning ichki ko‗rinishidagi razryadlar chapga bittaga suriladi va yana to‗rtinchi pozitsiyadagi raqam chop etiladi. Takrorlash sanagich qiymati 0 bo‗lguncha davom etadi (to‗rt marta) va boshqaruv ichki takrorlash operatoridan chiqadi.

do - while takrorlash operatori

do-while takrorlash operatori while operatoridan farqli ravishda oldin operator yoki blokni bajaridi va keyin takrorlash shartini tekshiradi. Bu qurilma takrorlash tanasini kamida bir marta bajarilishini ta‘minlaydi. do-while takrorlash operatori quyidagi sintaksisga ega:

```
do
 <operator yoki
blok>;  while
(<ifoda>);
```

Bunday takrorlash operatorining keng qo‗llaniladigan holatlari – birorta jarayonni davom ettirish yoki to‗xtatish haqidagi murojaatdir.

```
#include <iosteam.h>
int main() {   char javob;
 do      {
          …// dastur tanasi
         cout<<‖Jarayonni to‘xtatish (N):_ ‖;
         cin >> javob;
          }
        while (javob
!=N) return 0;
}
```

Dastur toki ‖Jarayonni to‘xtatish (N):_ ‖ so‗roviga ―N‖ javobi kiritilmaguncha davom etadi.

break operatori

Takrorlash operatorlari bajarilishida shunday holatlar yuzaga kelishi mumkinki, unda takrorlashni oxiriga yetkazmasdan, qaysidir qadamda takrorlashdan chiqish zarurati bo‗lishi mumkin. Boshqacha aytganda takrorlashni ―uzish‖ kerak bo‗lishi mumkin. Bunda break operatoridan foydalanish mumkin. break operatori takrorlash operatori tanasining ixtiyoriy (zarur) joylariga qo‗yish orqali shu joydan

qurilmadan chiqishni ta‘minlash mumkin. switch-case operatorini mohiyatiga ham break operatorini qo‗llash orqali erishilgan.

Quyidagi dasturda ikkita ichma-ich joylshagan takrorlash operatoridan foydalangan holda foydalanuvchi tomonidan kiritilgan sonni qandaydir 3 va 7 sonlariga nisbatan qanday oraliqqa tushishi aniqlanadi. Tashqi takrorlashda ―Son kiriting (0-to‘xtash):_‖ so‗rovi beriladi va javob javob_son o‗zgaruvchisiga o‗qiladi. Agar son 0 farqli bo‗lsa, ichki takrorlash operatorida sonning qandaydir oraliqqa tushsa shu haqda xabar beriladi va ichki takrorlash operatoridan chiqiladi. Javob tariqasida 0 kiritilsa dastur o‗z ishini tugatadi.

```
#include <iosteam.h>
int main() {
  int javob_son=0;
  do
    {
     while (javob_son)
      {
        if (javob_son<3)
          { cout<<‖3 kichik !‖;
break; }          if
(3>=javob_son<=7)
          { cout<<‖3 va 7 oraligida !‖;
break; }          if (javob_son<7)
          { cout<<‖7 dan katta !‖;  break; }
       }
      cout<<‖\nSon kiriting (0-
to‘xtash):_‖;       cin >> javob_son;
     }
   while (javob_son !=0)
return 0;
}
```

Amaliyotda break operatoridan cheksiz takrorlashlardan chiqishda foydalaniladi. for (; ;)

```
 {
// 1– shart
if (…)
  {
    …
break;      }
        // 2– shart
        if (…)
          {
            ...
           break;     }   ...  }
```

Bu misolda cheksiz for takrorlashidan 1 yoki 2 shart bajarilganda chiqiladi.

continue operatori

continue operatori xuddi break operatoridek takrorlash operatori tanasini bajarishni to_xtatadi, lekin dasturni qurilmadan chiqib ketmasdan takrorlashning keyingi qadamiga ―sakrab‖ o_tishini tayinlaydi.

continue operatorini qo_llanishiga misol tariqasida 2 va 50 oralig_idagi tub sonlarni topadigan dastur matnini keltiramiz.

```
#include <iosteam.h>
int main()
{
  bool Bulinadi=false;
for (int i=2; i<50; i++)
  {
    for (int j=2; j<i; j++)
     {
       if (i%j) continue;
       else
        {
          bo`linadi=true;
          break;
         }
    }
if (!bo`linadi) cout <<i<<‖ ―;
bo`linadi=false;
}
      return 0;
      }
```

Dasturda qo_yilgan masala ichma–ich joylashgan ikkita takrorlash operatorlari yordamida yechilgan. Birinchi takrorlash operatori 2 dan 50 gacha sonlarni hosil qilishga xizmat qiladi. Ichki takrorlash esa har hosil qilinayotgan sonni 2 dan shu sonning o_zigacha bo_lgan sonlarga bo_lib, qoldig_ini tekshiradi, agar qoldiq 0 sonidan farqli bo_lsa, navbatdagi songa bo_lish davom etadi, aks holda bo`linadi o_zgaruvchisiga true qiymat berib, ichki takrorlashni uzadi (son 2 dan o_zigacha bo_lgan qandaydir songa bo_linar ekan, demak u tub emas va keyingi sonlarga bo_lib tekshirishga hojat yo_q). Ichki takrorlashdan chiqqandan keyin bo`linadi qiymati false bo_lsa (!bo`linadi), son tub bo_ladi va u chop qilinadi.

goto operatori va nishonlar

Nishon – bu davomida ikkita nuqta (_:‘) qo_yilgan identifikator. Nishon bilan qandaydir operator belgilanadi va keyinchalik, dasturning boshqa bir qnomidan unga

shartsiz o‗tish amalga oshiriladi. Nishonga shartsiz o‗tish goto operatori yordamida bajariladi. goto operatori orqali faqat uning o‗zi joylashgan funksiya ichidagi operatorlarga o‗tish mumkin.

goto operatorining sintaksisi quyidagicha:

goto <nishon>;

Shartsiz o‗tish operatori – dasturni bajarishda kuchli va xavfli vositalardan hisoblanadi. Kuchliligi shundaki, uning yordamida algoritmning ―boshi berk‖ joylaridan ―chiqib‖ ketish mumkin. Ikkinchi tomondan, bloklarning ichiga o‗tish, takrorlash operatorlarini ichiga ―sakrab‖ kirish kutilmagan holatlarni yuzaga keltirishi mumkin.

Quyidagi misolda nishon yordamida takrorlashni amalga oshirish ko‗rsatilgan.

```
#include <iosteam.h>
int main()
{   int a,b;
        cout<<‖ Ikkita A va B natural sonlarning EKUB
        topish dastursi\n‖; cout<<‖A va B natural sonlarni
        kiriting: ‖  cin >>a>>b; nishon: if (a==b)
                {cout <<‖Bu sonlar EKUB = ‖<<a;

        return 0; } if
        (a>b) a-=b;
        else b-=a; goto
        nishon;
        }
```

Dastur ikkita natural sonlarning eng katta umumiy bo‗luvchisi (EKUB) Evklid algoritmi bilan topish masalasini yechadi. Nishon bilan belgilangan operatorda a va b sonlarni tengligi tekshiriladi. Agar ular teng bo‗lsa, ixtiyoriy bittasi, masalan a EKUB bo‗ladi va funksiyadan chiqiladi. Aks holda bu sonlarning kattasidan kichigi ayriladi va goto orqali yana ularning tengligini tekshiriladi.Takrorlash jarayoni a va b sonlar o‗zaro teng bo‗lib qolguncha davom etadi.

5-MAVZU: C++ DASTURLASH TILIDA FUNKTSIYALAR

Reja:

1. C++ tilida funksiyalar.
2. C++ tilida matematik funksiyalar.
3. C++ tilida foydalanuvchi funksiyalarini yaratish.

Funksiyalar

C++ da dasturlashning asosiy bloklaridan biri funksiya-lardir. Funksiyalarning foydasi shundaki, katta masala bir necha kichik bo'laklarga bo'linib, har biriga alohida funksiya yozilganda, masala yechish algoritmi ancha soddalashadi. Bunda dasturchi yozgan funksiyalar C++ ning standart kutubxonasi va boshqa firmalar yozgan kutub-honalar ichidagi funksiyalar bilan birlashtiriladi. Bu esa ishni osonlashtiradi. Ko'p holda dasturda takroran bejariladigan amalni funksiya sifatida yozish va kerakli joyda ushbu funksiyani chaqirish mumkin. Funksiyani dastur tanasida ishlatish uchun u chaqiriladi, yani uning ismi yoziladi va unga kerakli argumentlar beriladi. () qavslar ushbu funksiya chaqirig'ini ifodalaydi. Masalan:

```
foo(); k = square(l);
```

Demak, agar funksiya argumentlar olsa, ular () qavs ichida yoziladi. Argumentsiz funksiyadan keyin esa () qavslarning o'zi quyiladi.

Matematik kutubxona funksiyalari

Standart kutubxonaning matematik funksiyalari ko'pgina amallarni bajarishga imkon beradi. Biz bu kutubxona misolida funksiyalar bilan ishlashni ko'rib chiqamiz. Masalan bizning dasturimizda quyidagi satr bor bo'lsin: double = k; int m = 123; k = sin(m); kompilyator uchbu satrni ko'rganida,standart kutubxonadan sin funksiyasini chaqiradi. Kirish qiymati sifatida m ni berdik. Javob, yani funksiyadan qaytgan qiymat k ga berildi.Funksiya agumentlari o'zgarmas sonlar (konstanta) o'zgaruvchilar, ifodalar va boshqa mos keluvchi qiymat qaytaradigan funksiyalar bo'lishi mumkin. Masalan:

```
int g = 49, k = 100;
cout << "4900 ning ildizi -> "<< sqrt( g * k );
```

Ekranda:

4900 ning ildizi -> 70;

Matematik funksiyalar aksariyat hollarda double tipidagi qiymat qaytarishadi.Kiruvchi argumentning tipi sifatida esa double ga keltirilishi mumkin bo'lgan tip beriladi. Bu funksiyalarni ishlatish uchun math.h (yangi ko'rinishda cmath)e'lon faylini include bilan asosiy dastur tanasiga kiritish kerak.Quyida

matematik funksiya-lar kutubxonasining bazi bir a'zolarini beraylik. x va y o'zgaruvchilari double tipiga ega.

Funksiya	Aniqlanishi	Misol
ceil(x)	x ni x dan katta yoki unga teng b-n eng kichik butun songacha yahlitlaydi	ceil(12.6) = 13.0 ceil(-2.4) = -2.0
cos(x)	x ning trigonometrik kosinusi (x radianda)	cos(0.0) = 1.0
exp(x)	e ning x chi darajasi (eskponetsial f-ya)	exp(1.0) = 2.71828 exp(2.0) = 7.38906
fabs(x)	x ning absolut qiymati x>0 => abs(x) = x x=0 => abs(x) = 0.0 x<0 => abs(x) = -x	
floor(x)	x ni x dan kichik bo'lgan eng katta butun songacha yahlitlaydi	floor(4.8) = 4.0 floor(-15.9) = -16.0
fmod(x,y)	x/y ning qoldig'ini kasr son tipida beradi	fmod(7.3,1.7) = 0.5
log(x)	x ning natural lagorifmi (e asosiga ko'ra)	log(2.718282) = 1.0
log10(x)	x ning 10 asosiga ko'ra lagorifmi	log10(1000.0) = 3.0
pow(x,y)	x ning y chi darajasini beradi	pow(3,4) = 81.0 pow(16,0.25) = 2
sin(x)	x ning trigonometrik sinusi (x radianda)	sin(0.0) = 0.0
sqrt(x)	x ning kvadrat ildizi	sqrt(625.0) = 25.0
tan(x)	x ning trigonometrik tangensi (x radianda)	tan(0.0) = 0

Funksiyalarning tuzilishi

Funksiyalar dasturchi ishini juda yengillashtiradi. Funksiyalar yordamida dastur modullashadi, qismlarga bo'limadi. Bu esa keyinchalik dasturni rivojlantirishni osonlashtiradi. Dastur yozilish davrida hatolarni topishni yengillashtiradi. Bir misolda funksiyaning asosiy qismlarini ko'rib chiqaylik.

```
int foo(int
k, int t) {
int result;
result = k *
```

```
t;      return
(result); }
```

Yuqoridagi foo funksiyamizning ismi, () qavslar ichidagi parametrlar – int tipidagi k va t lar kirish argument-laridir, ular faqat ushbu funksiya ichida ko'rinadi va qo'llaniladi. Bunday o'zgaruvchilar lokal(local-mahalliy) deyiladi. result foo() ning ichida e'lon qilinganligi uchun u ham lokaldir. Demak biz funksiya ichida o'zgaruvchilarni va klaslarni (class) e'lon qilishimiz mumkin ekan. Lekin funksiya ichida boshqa funksiyani e'lon qilib bo'lmaydi. foo() funksiyamiz qiymat ham qaytaradi. Qaytish qiymatining tipi foo() ning e'lonida eng boshida kelgan - int tipiga ega. Biz funksiyadan qaytarmoqchi bo'lgan qiymatning tipi ham funksiya e'lon qilgan qaytish qiymati tipiga mos kelishi kerak - ayni o'sha tipda bo'lishi yoki o'sha tipga keltirilishi mumkin bo'lgan tipga ega bo'lishi shart. Funksiyadan qiymatni return ifodasi bilan qaytaramiz. Agar funksiya hech narsa qaytarmasa e'londa void tipini yozamiz. Yani:

```
void funk(){
  int g = 10;

  cout <<
g;   return;
}
```

Bu funksiya void (bo'sh, hech narsasiz) tipidagi qiymatni qaytaradi. Boshqacha qilib aytganda qaytargan qiymati bo'sh to'plamdir. Lekin funksiya hech narsa qaytarmaydi deya olmaymiz. Chunki hech narsa qaytarmaydigan mahsus funksiyalar ham bor. Ularning qaytish qiymati belgilana-digan joyga hech narsa yozilmaydi. Biz unday funksiyalarni keyinroq qo'rib chiqamiz. Bu yerda bir nuqta shuki, agar funksiya mahsus bo'lmasa, lekin oldida qaytish qiymati tipi ko'rsatilmagan bo'lsa, qaytish qiymati int tipiga ega deb qabul qilinadi.

Void qaytish tipli funksiyalardan chiqish uchun return; deb yozsak yetarlidir.Yoki return ni qoldirib ketsak ham bo'ladi. Funksiyaning qismlari bajaradan vazifasiga ko'ra turlicha nomlanadi. Yuqorida korib chiqqanimiz funksiya aniqlanishi (function definition) deyiladi, chunki biz bunda funksiyaning bajaradigan amallarini funksiya nomidan keyin,{} qavslar ichida aniqlab yozib chiqyapmiz. Funksiya aniqlanishida {} qavslardan oldin nuqta-vergul (;) quyish hatodir. Bundan tashqari funksiya e'loni, prototipi yoki deklaratsiyasi (function prototype) tushunchasi qo'llaniladi. Bunda funksiyaning nomidan keyin hamon nuqta-vergul quyiladi, funksiya tanasi esa berilmaydi. C++ da funksiya qo'llanilishidan oldin uning aniqlanishi yoki hech bo'lmaganda e'loni kompilyatorga uchragan bo'lishi kerak. Agar funksiya e'loni boshqa funksiyalar aniqlanishidan tashqarida berilgan bo'lsa, uning

kuchi ushbu fayl ohirigacha boradi. Biror bir funksiya ichida berilgan bo'lsa kuchi faqat o'cha funksiya ichida tarqaladi. E'lon fayllarda aynan shu funksiya e'lonlari berilgan bo'ladi. Funksiya e'loni va funksiya aniqlanishi bir-biriga mos tushishi kerak. Funksiya e'loniga misol:

```
double square(char,
bool); float
average(int a, int b,
int c);
```

Funksiya e'lonlarda kirish parametrlarining faqat tipi yozish kifoya, huddi square() funksiyasidek. Yoki kiruvchi parametrlarning nomi ham berilishi mumkin, bu nomlar kompilyator tarafidan etiborga olinmaydi, biroq dasturning o'qilishini ancha osonlashtiradi. Bulardan tashqari C++ da funksiya imzosi (function signature) tushunchasi bor. Funksiya imzosiga funksiya nomi, kiruvchi parametrlar tipi, soni, ketma-ketligi kiradi. Funksiyadan qaytuvchi qiymat tipi imzoga kirmaydi.

```
int foo();          //No1  int foo(char, int);
//No2 double foo();        //No3 - No1 funksiya
bilan imzolari ayni.  void foo(int, char);
//No4 - No2 bilan imzolari farqli.  char
foo(char, int);   //No5 - No2 bilan imzolari
ayni.
int foo(void);       //No6 - No1 va No3 bilan
imzolari ayni,                //     No1 bilan
e'lonlari ham ayni.
```

Yuqoridagi misolda kirish parametrlari bo'lmasa biz () qavsning ichiga void deb yozishimiz mumkin (No6 ga qarang). Yoki () qavslarning quruq o'zini yozaversak ham bo'ladi (No1 ga qarang). Yana bir tushuncha - funksiya chaqirig'idir. Dasturda funksiyani chaqirib,qo'llashimiz uchun uning chaqiriq ko'rinishini ishlatamiz. () qavslari funksiya chaqirig'ida qo'llaniladi. Agar funksiyaning kirish argumentlari bo'lmasa, () qavslar bo'sh holda qo'llaniladi. Aslida () qavslar C++ da operatorlardir. Funksiya kirish parametrlarini har birini ayri-ayri yozish kerak, masalan yuqoridagi

```
float average(int a, int
b, int c);  funksiyasini
float average(int
a,b,c); // Hato!
```

deb yozishimiz hatodir.

Hali etib o'tganimizdek, funksiya kirish parametrlari ushbu funksiyaning lokal o'zgaruvchilaridir. Bu o'zgaruvchilarni funksiya tanasida boshqattan e'lon qilish sintaksis hatoga olib keladi. Bir dastur yozaylik.

```
//Funksiya bilan ishlash
# include <iostream.h>

int foo(int a, int b); //Funksiya prototipi,
//argumentlar ismi shart emas.

int main() {    for (int k = 1; k <6; k++){
for (int l = 5; l>0; l--){          cout <<
foo(k,l) << " ";        //Funksiya chaqirig'i.
     }//end
for (l...)
cout <<
endl;
}//end for
(k...)
return (0);
} //end
main()

//foo() funksiyasining aniqlanishi
int foo(int c,
int d) {
//Funksiya
tanasi
return(c *
d);
}
```

Ekranda:

```
5 4 3 2 1
10 8 6 4 2
15 12 9 6 3
20 16 12 8 4
25 20 15 10 5
```

Bizda ikki sikl ichida foo() funksiyamiz chaqirilmoqda. Funksiyaga k va l o'zgaruvchilarining nushalari uzatil-moqda. Nushalarning qiymati mos ravishda funksiyaning aniqlanishida berilgan c va d o'zgaruvchilarga berilmoqda. k va l ning nushalari deganimizda adashmadik, chunki ushbu o'zgaruvchilarining qiymatlari funksiya chaqirig'idan hech qanday ta'sir ko'rmaydi. C++ dagi funksiyalarning bir noqulay tarafi shundaki, funksiyadan faqat bitta qiymat qaytadi. Undan tashqari yuqorida ko'rganimizdek, funksiyaga berilgan o'zgaruvchilarning faqat nushalari bilan ish ko'rilarkan. Ularning qiymatini normal sharoitda funksiya ichida o'zgartirish mumkin emas. Lekin bu muammolar ko'rsatkichlar yordamida osonlikcha hal etiladi. Funksiya chaqiriqlarida avtomatik ma'lumot tipining konversiyasi bajariladi. Bu amal kompilyator tomonidan bajarilganligi sababli funksiyalarni chaqirganda ehtiyot bo'lish kerak. Javob hato ham bo'lishi mumkin. Shu sababli kirish parametrlar tipi sifatida katta hajmli tiplarni qo'llash maqsadga muofiq bo'ladi. Masalan double tipi har qanday sonli tipdagi qiymatni o'z ichiga olishi mumkin. Lekin bunday qiladigan bo'lsak, biz tezlikdan yutqazishimiz turgan gap. Avtomatik konversiyaga misol keltiraylik.

```
int division(int m, int
k){ return (m / k);
}  dasturda
chaqirsak:...
float f =
14.7; double
d = 3.6;
int j = division(f,d); //f 14 bo'lib kiradi, d 3 bo'lib kiradi
                    // 14/3 - butun sonli bo'lish esa 4
javobini beradi   cout << j;
...
 Ekranda:
 4
```

Demak kompilyator f va d o'zgaruvchilarining kasr qismlarini tashlab yuborar ekan. Qiymatlarni pastroq sig'imli tiplarga o'zgartirish hatoga olib keladi.

6-MAVZU: C++ DASTURLASH TILIDA MASSIVLAR

Reja

1. Massiv tushunchasi
2. Bir o'lchovli massivlar
3. Ikki o'lchovli massivlar
4. Ikki o`lchamli statik massivlarni e`lon qilish.
5. Massivlarga qiymat berish.
6. Satrli massivlar.
7. Ko`p o`lchovli massivlar.

Tayanch iboralar: Massiv, bir o'lchovli massiv, ikki o'lchovli massiv.

.

***Dars maqsadi*:** Talabalarda C++ dasturlash tilida massivlar va ulardan foydalanish ko'nikmalarini hosil qilish.

Massiv - bu bir xil toifali, chekli qiymatlarning tartiblangan to`plamidir. Massivlarga misol qilib matematika kursidan ma`lum bo`lgan vektorlar, matritsalarni ko`rsatish mumkin.

Massiv bir o`lchamli deyiladi, agar uning elementiga bir indeks orqali murojaat qilish mumkin bo`lsa.

Bir o`lchamli massivni e`lon qilish quyidagicha bo`ladi:

<toifa> <massiv_nomi> [elementlar_soni] = { boshlang'ich qiymatlar }; Quyida massivlarni e`lon qilishga bir necha misollar keltirilgan: 1) float a[5];

2) int m[6];

3) bool b[10];

1) a elementlari haqiqiy sonlardan iborat bo`lgan, 5 ta elementdan tashkil topgan massiv. Indekslari esa 0 dan 4 gacha bo`lgan sonlar

float a[5];					
Massiv elementilari	a[0]	a[1]	a[2]	a[3]	a[4]
qiymati	4	-7	15	5.5	3

2) m elementlari butun sonlardan iborat bo`lgan, 6 ta elementdan tashkil topgan massiv. Indekslari esa 0 dan 5 gacha bo`lgan sonlar.

int m[6];						
Massiv elementilari	m[0]	m[1]	m[2]	m[3]	mas2[4]	mas2[5]
qiymati	2	-17	6	7	13	-3

3) b elementlari mantiqiy qiymatlardan (true, false) iborat bo`lgan 10 ta elementdan tashkil topgan massiv. Indekslari esa 0 dan 9 gacha bo`lgan sonlar.

Massiv elementlariga murojaat qilish oddiy o`zgaruvchilarga murojaat qilishdan biroz farq qiladi. Massiv elementiga murojaat qilish uning indeksi orqali bo`ladi. a[1] = 10; a massivining 1 – elementi 10 qiymat o‘zlashtirsin; cin >> a[2]; a massivining 2 – elementi kirtilsin;

cout << a[3]; a massivining 3 – elementi ekranga chiqarilsin;

Massivni e'lon qilishda uning elementlariga boshlang'ich qiymat berish mumkin va buning bir nechta usuli mavjud.

1) O'lchami ko'rsatilgan massivni to'liq initsializatsiyalash. int k[5] = { 2, 3, 7, 8, 6};

Bu yerda 5 ta elementdan iborat bo'lgan k massivi e'lon qilingan va massivning barcha elemantlariga boshlang'ich qiymat berilgan.

2) O'lchami ko'rsatilgan massivni to'liqmas initsializatsiyalash.

int k[5] = { 2, 3, 7 };

Bu yerda 5 ta elementdan iborat bo'lgan k massivi e'lon qilingan va massivning dastlabki 3 ta elemantlariga boshlang'ich qiymat berilgan.

3) O'lchami ko'rsatilmagan massivni to'liq initsializatsiyalash.

int k[] = { 2, 3, 7, 8, 6};

Shuni takidlash lozimki, agar massiv o'lchami ko'rsatilmasa, uni to'liq initsializatsiyalash shart. Bu xolda massiv o'lchami kompilyatsiya jarayonida massiv elementlari soniga qarab aniqlanadi. Bu yerda massiv o'lchami 5 ga teng.

4) O'lchami ko'rsatilgan massivning barcha elementlariga boshlang'ich qiymat 0 berish: int k[5] = { 0 };

O'lchami ko'rsatilgan massivning barcha elementlariga boshlang'ich qiymat 0 berish

```
#include
<iostream>  using
namespace std;
int main()
{
   int a[10] = { 0 };
   //massivning barcha elementlariga 0 qiymat berish

   for (int i = 0; i < 10; i++)
   cout << "a[" << i << "]=" << a[i] << endl;

   return 0;
}
```

Misol. A[1..5] massiv elementlarini teskari tartibda chiqaruvchi dastur tuzing.

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

```
#include <conio.h>;
#include <math.h>;
 int main()
{
      int a[17]={0};
      int s;
        for( int i=1; i<=5;
i++)          {cout
<<"a["<<i<<"]=";          cin
>>a[i]; }          for( int i=5;
i>=1; i--)
cout<<"a[i]="<<a[i]<<endl;
getch();
}
```

Simvolli massivlar.

C ++ tilida satrlar simvolli massivlar sifatida ta'riflanadi. Simvolli massivlar quyidagicha tasvirlanishi mumkin: Char pas[10];
Simvolli massivlar quyidagicha initsializatsiya qilinadi:

Char capital[]=‖TASHKENT‖; Bu holda avtomatik ravishda massiv elementlari soni aniqlanadi va massiv ohiriga satr ko‘chirish _\n‘ simvoli qo‘shiladi.
Yuqoridagi initsializatsiyani quyidagicha amalga oshirish mumkin:
Char capital[]={_T‘,‘A‘,‘S‘,‘H‘,‘K‘,‘E‘,‘N‘,‘T‘,‘\n‘}; Bu holda so‘z ohirida _\n‘ simvoli aniq ko‘rsatilishi shart.

Misol uchun palindrom masalasini ko‘rib chiqamiz. Palindrom deb oldidan ham ohiridan ham bir hil o‘qiladigan so‘zlarga aytiladi. Misol uchun non. Dasturda kiritilgan so‘z palindrom ekanligi aniqlanadi:

```
#include
<iostream.h> void
main()
{    gets(a); for(
int j=0,
a[j]!=‘\0‘;j++);
I=0;
while(I<j) if (a[I++]!=a[j--]) break;
if ((j-I)>1) Cout<<(—Palindrom emas‖) else Cout<<(—Palindrom‖);
```

Keyingi misolimizda kiritilgan so‘zdan berilgan harf olib tashlash dasturi berilgan:

```
#include <iostream.h> void
main()
{  char s[];
int c;
gets(a);
  int i, j;
  for ( i = j = 0;
```

```
s[i] != '\0'; i++) if ( s[i] != c )
s[j++] = s[i];
s[j] = '\0';
puts(s); }
```

Har gal 's' dan farqli simvol uchraganda , u J pozitsiyaga yoziladi va faqat shundan so‘ng J qiymati 1 ga oshadi. Bu quyidagi yozuvga ekvivalent:

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

Bir o`lchamli massivlar uchun ishlatilgan o`zgaruvchilar, bir xil jinsdagi berilganlarni xotirada saqlash uchun foydalaniladi. Ikki o`lchamli massivlarda esa, satr va ustunlar orqali bir xil jinsdagi qiymatlarni ikki o`lchamli o`zgaruvchilar ichida saqlash uchun foydalaniladi.

Ikki o`lchamli statik massivlarni e`lon qilish. toifa massiv_nomi [massiv_satrlari_soni][massiv_ustunlari_soni];

Ikki o`lchamli statik massivlarning e`lon qilinishida, bir o`lchamlidan farqi, massiv nomidan keyin qirrali qavs ichida ikkita qiymat yozilganligidadir. Bulardan birinchisi, satrlar sonini, ikkinchisi esa ustunlar sonini bildiradi. Ya'ni ikki o'lchamli massiv elementiga ikkita indeks orqali murojaat qilinadi. Ikki o`lchamli massivlar matematika kursidan ma`lum bo`lgan matritsalarni eslatadi.

Ikki o'lchamli massiv e'loniga misol:

```
int a[3][3], b[2][4];
```

A matritsa 3 ta satr, 3 ta ustunga ega;

B matritsa 2 ta satr, 4 ta ustunga ega;

Ikki o'lchamli massivlarda 1 - indeks satrni, 2 - indeks ustunni bildiradi.

Birinchi satrning dastlabki elementi a10 – a biru nol element deb o`qiladi. a o`n deyilmaydi.

m ta satr va n ta ustunga ega bo`lgan massivga (mxn) o`lchamli massiv deyiladi. Agar

m=n (satrlar va ustunlar soni teng) bo'lsa kvadrat massiv deyiladi. Ko'p o'lchamli massivlarni initsializats iyalash misollar: int a[2][2]={1,2,7,3};

int b[2][3]={ {0,1,2}, {3,4,5} };

Massivlarni qo`llanilishiga misol keltiradigan bo`lsak, satrlar talabalarni, ustunlar fanlardan olgan baholarini bildirsin. Ya`ni m ta talaba, n ta fan. n - ustunga talabalarning o`rtacha baholari hisoblanib, shu asosida stipendiya bilan ta`minlansin. Va hakazo, bunga o`xshash ko`plab misollar keltirish mumkin. Bu masalalarga to`xtalishdan oldin bir ikkita oddiy masalar bilan tanishib chiqaylik.

1 - Masala. A(mxn) matritsa berilgan. Shu matritsa elementlarini kirituvchi va ekranga jadval ko`rinishida chiqaruvchi dastur tuzilsin. #include <iostream> using namespace std;

```
int main()
{
  int  m, n, a[10][10];
  cout << "Satrlar sonini kiriting \nm="; cin >> m;
```

```
cout << "Ustunlar sonini kiriting \nn=";    cin >> n;
cout <<"Massiv elementlarini kiriting \n";    for(int satr = 0;  satr < m ; satr++)
for(int ustun = 0; ustun < n; ustun++)
{
  cout << "a[" << satr << "][" << ustun << "]=";
  cin >> a[satr][ustun];
}

// matritsani jadval shaklida chiqarish
for(int satr = 0; satr < m; satr++)
{
  for(int ustun = 0; ustun < n; ustun++)
  cout << a[satr][ustun] << "\t";     cout<<"\n";
}
return 0; }
```

JADVALLAR.

Ikki o‘lchovli massivlar matematikada matritsa yoki jadval tushunchasiga mos keladi. Jadvallarning initsializatsiya qilish qoidasi, ikki o‘lchovli massivning elementlari massivlardan iborat bo‘lgan bir o‘lchovli massiv ta'rifiga asoslangandir. Misol uchun ikki qator va uch ustundan iborat bo‘lgan haqiqiy tipga tegishli d massiv boshlang‘ich qiymatlari quyidagicha ko‘rsatilishi mumkin:

float d[2][3]={(1,-2.5,10),(-5.3,2,14)};

Bu yozuv quyidagi qiymat berish operatorlariga mosdir:

d[0][0]=1;d[0][1]=-2.5;d[0][2]=10;d[1][0]=-5.3;d[1][1]=2;d[1][2]=14; Bu qiymatlarni bitta ro‘yhat bilan hosil qilish mumkin: float d[2][3]={1,-2.5,10,-5.3,2,14}; Initsializatsiya yordamida boshlang‘ich qiymatlar aniqlanganda massivning hamma elementlariga qiymat berish shart emas.

Misol uchun: int x[3][3]={(1,-2,3),(1,2),(-4)}.

Bu yozuv quyidagi qiymat berish operatorlariga mosdir: x[0][0]=1;x[0][1]=-2;x[0][2]=3;x[1][0]=-1;x[1][1]=2;x[2][0]=-4;

Initsializatsiya yordamida boshlang‘ich qiymatlar aniqlanganda massivning birinchi indeksi chegarasi ko‘rsatilishi shart emas, lekin qolgan indekslar chegaralari ko‘rsatilishi shart. Misol uchun:

Double x[][2]={(1.1,1.5),(-1.6,2.5),(3,-4)}

Bu misolda avtomatik ravishda qatorlar soni uchga teng deb olinadi.

Quyidagi ko‘radigan misolimizda jadval kiritilib har bir qatorning maksimal elementi aniqlanadi va bu elementlar orasida eng kichigi aniqlanadi:

```
#include <iostream.h> void main()
{ double a[4,3]; double s,max=0.0,min=0.0; int i,j;
for(i=0;i<4;i++) { for(j=0;j<3;j++) { Cout<<(—
a[%d][%d]=‖,i,j);Cin>>(—%f‖,s);a[i,j]=s; if (max<s) max=s;
};
Cout<<(—\n‖); if (max<min) min=max;  }
Cout<<(—\n min=%f‖,min);
}
```

SATRLI MASSIVLAR

C ++ tilida so‘zlar massivlari ikki o‘lchovli simvolli massivlar sifatida ta'riflanadi. Misol uchun:

Char Name[4][5]. Bu ta'rif yordamida har biri 5 ta harfdan iborat bo‘lgan 4 ta so‘zli massiv kiritiladi. So‘zlar massivlari quyidagicha initsializatsiya qilinishi mumkin: Char Name[3][8]={—Anvar‖,‖Mirkomil‖,‖Yusuf‖}.

Bu ta'rifda har bir so‘z uchun hotiradan 8 bayt joy ajratiladi va har bir so‘z ohiriga _\0‘ belgisi quyiladi.

So‘zlar massivlari initsializatsiya qilinganda so‘zlar soni ko‘rsatilmasligi mumkin. Bu holda so‘zlar soni avtomatik aniqlanadi:

Char comp[][9]={—komp'yuter‖,‖printer‖,‖kartridj‖}.

Quyidagi dasturda berilgan harf bilan boshlanuvchi so‘zlar ruyhati bosib chiqariladi:

#include <iostream.h> void main() { char a[10][10]; char c;
for (int i=0;i<10;i++) gets(a[i]); c=getchar();
for (i=0;i<10;i++) if (a[i][0]==c) puts(a[i]);
}

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

#include <iostream.h> void main() { char a[10][10]; char s[10]; int k[10];
gets(s);
for (int i=0;i<10;i++) gets(a[i]); for (i=0;i<10;i++) {Cin>>(—%d‖,k[i]}; for (int i=0;i<10;i++) if (k[i]==2) puts(a[i]);
}

BIR NECHA INDEKSLI MASSIVLAR.

Massivlar bir necha indeksga ega bo'lishlari mumkin. C++ kompilyatorlari eng kamida 12 ta indeks bilan ishlashlari mumkin. Masalan, matematikadagi m x n kattalikdagi matritsani ikkita indeksli massiv yordamida berisak bo'ladi. int matritsa [4][10];

Yuqorida to'rt satrlik, 10 ustunlik matritsani e'lon qildik. Bir indeksli massivlar kabi ko'p indeksli massivlarni initsalizatsiya ro'yhati bilan birga e'lon qilish mumkin. Masalan:

```
char c[3][4] = {
                { 2, 3,9, 5}, // birinchi satr qiymatlari
                {-10, 77,5, 1}, // ikkinchi  "       "
                { 90,233,3,-3}  // uchinchi  "       "
              };
int m[2][2] = {56,77,8,-3}; // oldin birinchi satrga qiymatlar beriladi,
// keyin esa ikkinchi satrga

double d[4][3][6] = {2.55, -46,0988}; // birinchi satrning dastlabki ikkita
                                      // elementi qiymat oladi,
                                      // massivning qolgan elementlari esa
                                      // nolga tenglashtiriladi
```

Massivning har bir indeksi alohida [] qavslar ichiga olinishi kerak. Yuqoridagi c[][] massivining ikkinchi satr, birinchi ustunidagi elementi qiymatini birga oshirish uchun ++c[1][0]; // yoki c[1][0]++; // c[1][0] += 1;
// c[1][0] = c[1][0] + 1;
deb yozishimiz mumkin. Massiv indekslari 0 dan boshlanishini unutmaslik zarur. Agar ++c[1,0];
deb yozganimizda hato bo'lar edi. C++ bu yozuvni ++c[0]; deb tushunar edi, chunki
kompilyator vergul bilan ajratilgan ro'yhatning eng ohirgi elementini qabul qilardi. Hullas, C++ dagi ko'p indeksli massivlar dasturchiga behisob imkoniyatlar beradi. Undan tashqari, ular hotirada statik joylashganligi uchun ularning ishlash tezligi kattadir. C++ dagi ko'p indeksli massivlar hotirada ketma-ket joylashgandir. Shu sababli agar massiv funksiyaga kirish parametri sifatida berilsa, faqat birinchi indeks tushurilib qoldiriladi, qolgan indekslar esa yozilishi shartdir. Aks taqdirda funksiya massiv kattaligini to'g'ri keltirib chiqarolmaydi. Massiv parametrli bir funksiya e'lonini beraylik.

```
//Ko'p indeksli massivlar  # include <iostream.h>
int indeks = 3;
int intArray[indeks][4] = {}; // hamma elementlar 0 ga tenglashtirildi void
printArray(int mass[][4], int idx){   // funksiya e'loni
   for (int i = 0; i < idx; i++) { // massivning birinchi indeksini
                                   // o'zgartirsa bo'ladi
     for (int k = 0; k < 4; k++){ // massivning ikkinchi indeksi o'zgarmaydi
cout << mass[i][k];
     }
     cout << endl;
  }    return; }
...
int main() {
...
printArray(intArray); // funksiya chaqirig'i
...
return (0);
}
```

Massivning indekslarini funksiyaga bildirish yana muammoligicha qoladi. Albatta, birinchi indeksdan tashqari qolgan boshqa indekslar kattaligini funksiya ichida berish ma'noga egadir. Lekin birinchi indeks kattaligini tashqaridan, qo'shimcha parametr sifatida bersak, funksiyamiz chiroyliroq chiqadi, turli kattalikdagi massivlarni o'lish imkoniga ega bo'ladi.

7-MAVZU: C++ TILIDA KO‘RSATKICHLAR. C++DA SATRLAR VA ULAR USTIDA AMALLAR

Reja:

1. C++ tilida ko`rsatkichlar. Funksiyaga ko_rsatkich.
3. C++ tilida ko_rsatkichga boshlang_ich qiymatlar berish.
4. C++ tilida murojaatlar.
5. C++ tilida satrlar..

1. Dastur matnida o_zgaruvchi e‘lon qilinganda, kompilyator o_zgaruvchiga xotiradan joy ajratadi (dastur kodi xotiraga yuklanganda berilganlar uchun segmentning boshiga nisbatan siljishini aniqlaydi) va ob‘ekt kod hosil qilishda shu o_zgaruvchi uchragan joyga uning adresini joylashtiradi.

Umuman olganda, dastur ob‘ektlarining (o_zgaruvchilar, funksiyalarning) adreslarini xotiraning alohida joyida saqlash va ular ustidan amallar bajarish mumkin. Qiymatlari adres bo_lgan o_zgaruvchilarga ko_rsatkich o_zgaruvchilar deyiladi.

Ko_rsatkich uch xil turda bo_lishi mumkin:

a) birorta ob‘ektga, xususan o_zgaruvchiga ko_rsatkich;

b) funksiyaga ko_rsatkich;

v) voidga ko_rsatgich.

Ko_rsatgichning bu xususiyatlari uning qabul qilishi mumkin bo_lgan qiymatlari bilan farqlanadi.

Ko_rsatkich albatta birorta turga bog_langan bo_lishi kerak, ya‘ni u ko_rsatgan adresda qandaydir qiymat joylanishi mumkin va bu qiymatning xotirada qancha joy egallashi oldindan ma‘lum bo_lishi shart.

Funksiyaga ko_rsatkich dastur joylashgan xotiradagi funksiya kodining boshlang_ich adresi ko_rsatadi, ya‘ni funksiyaga murojaat bo_lganda (chaqirilganda) boshqarish shu adresga uzatiladi.

Funksiyaga ko_rsatkich orqali murojaat, funksiyaga vositali murojaat hisoblanadi. Chunki, funksiyaga uning nomi bo_yicha emas, balki funksiyaga ko_rsatuvchi o_zgaruvchi orqali amalga oshiriladi. Funksiyani boshqa funksiyaga argument sifatida uzatish ham funksiya ko_rsatkichi orqali bajariladi.

Funksiyaga ko_rsatkichning yozilish sintaksisi quyidagicha:

<tur> (* <nom>) (<parametrlar (turining) ro_yxati>);

bunda <tur> – funksiya qaytaruvchi qiymat turi; * <nom> - ko_rsatkich o_zgaruvchining nomi; <argumentlar (turining) ro_yxati> - funksiya parametrlarining (yoki ularning turlarining) ro_yxati. Masalan:

int (*fun)(float,float);

Bu yerda fun nomidagi funksiyaga ko_rsatkich e‘lon qilingan - funksiya butun son turida qiymat qaytaradi va u ikkita haqiqiy turdagi parametrlardan iborat.

Biror ob‘ektga (shu jumladan o_zgaruvchiga) ko_rsatkich. Bunday ko_rsatkichda ma‘lum turdagi (tayanch yoki hosilaviy) berilganlarning xotiradagi adresi joylashadi. Ob‘ektga ko_rsatkich quyidagicha e‘lon qilinadi:

<tur> *nom;

Bu yerda <tur> - ko_rsatkich aniqlaydigan adresdagi qiymat turi bo_lib, u maydon turidan tashqari xar qanday tur bo_lishi mumkin. Agar bir turda bir nechta ko_rsatkichlar e‘lon qilinadigan bo_lsa, har bir ko_rsatkich uchun _*‘ belgisi qo_yilishi shart: int *i, j,* k; float x,*y,*z;

Bu misolda i va k - butun turdagi ko_rsatkichlar va j - butun turdagi o_zgaruvchi, ikkinchi operatorda x- haqiqiy o_zgaruvchi va y,z haqiqiy turdagi ko_rsatkichlar e‘lon qilingan .

void ko_rsatkich. Bu ko_rsatkich ob‘ekt turi oldindan ma‘lum bo_lmaganda ishlatiladi, ya‘ni bir ko_rsatkichda turli vaqtda har xil turdagi ob‘ektlar adresi saqlanishi mumkin. Faqat void ko_rsatkichga har qanday turdagi ko_rsatkichning qiymatini yuklash mumkin. Lekin, ko_rsatilgan adresdagi qiymatni ishlatishdan oldin, uni turi aniq bir turga oshkor ravishda keltirilishi kerak.

void ko_rsatkichni e‘lon qilish kuyidagicha bo_ladi: void *<nom>;

Ko_rsatrichni o_zi o_zgarmas yoki o_zgaruvchan bo_lishi va o_zgarmas yoki o_zgaruvchiga ko_rsatiishi mumkin, masalan:

```
int i;       // butun o_zgaruvchi          const int ci=1;
// butun o_zgarmas          int *pi;      //  butun o_zgaruvchiga
ko_rsatkich          const int *pci;    //  butun o_zgarmasga
ko_rsatkich          int *const cp=&i;    //  butun
o_zgaruvchiga o_zgarmas-ko_rsatkich          const int * const
cpc=&ci; //  butun o_zgarmasga o_zgarmas-ko_rsatkich
```

Misollardan ko_rinib turibdiki, _*‘ va ko_rsatkich nomi orasida turgan const modifikatori faqat ko_rsatkichning o_ziga tegishli hisoblanadi va uni o_zgartish mumkin emasligini bildiradi, _*‘ belgisidan chapda turgan const esa ko_rsatgan adresdagi qiymatni o_zgarmas ekanligini bildiradi. Ko_rsatkichga boshlang_ich qiymatni berish uchun _&‘ - adresni olish amali ishlatiladi.

Ko_rsatkich o_zgaruvchilarning amal qilish sohasi, yashash davri va ko_rinish sohasi umumiy qoyidalarga bo_ysunadi.

Ko_rsatkichga boshlang_ich qiymatlar berish (initsializatsiyalash)

Ko_rsatkichlar ko_pincha dinamik xotira (boshqa nomi ―uyum‖ yoki ―heap‖) bilan bog_liq holda ishlatiladi. Xotiraning deyilishiga sabab, bu sohadagi bo_sh xotira dastur ishlash jarayonida, kerakli paytida ajratib olinadi va bu xotiraga zarurat qolmaganida qaytariladi
(bo_shatiladi) va u keyinchalik dastur tomonidan yana ishlatilishi mumkin. Bunday xotiraga faqat ko_rsatkichlar yordamida murojaat qilish mumkin.

Bunday o_zgaruvchilar dinamik o_zgaruvchilar deyiladi va ularni yashash vaqti yaratilgan nuqtadan boshlab dastur oxirigacha yoki oshkor ravishda bo_shatilish joyigacha.

Ko_rsatkichlarni e‘lon qilishda unga boshlang_ich qiymatlar berish mumkin. Boshlang_ich qiymat (initsializator) ko_rsatkich nomidan so_ng yoki qavs ichida yoki _=‘ belgidan keyin beriladi. Boshlang_ich qiymatlar quyidagi usullar bilan berilishi mumkin:

1. Ko‘rsatkichga mavjud bo_lgan ob‘ektning adresini berish: adresni olish amal orqali:

```
int i=5,k=4;  // butun o_zgaruvchilar
int *p = &i;   // p ko_rsatkichga i o_zgaruvchini adresi yoziladi
int *p1(&k);  // p1 ko_rsatkichga k o_zgaruvchini adresi yoziladi
```

boshqa initsializatsiyalangan ko_rsatkichni qiymatini berish:

```
int * r=p; // p oldin e‘lon qilingan va qiymatga ega bo_lgan ko_rsatkich
```

massiv yoki funksiya nomini berish:

```
int  b[10];  //massivni e‘lon qilish
int * t=b; //massivning boshlang_ich adresini berish
…

void f(int a)  { /* … */} // funksiyani aniqlash
void (*pf)(int);    // funksiyaga ko_rsatkichni e‘lon qilish
pf=f;     // funksiyani adresini ko_rsatkichga berish
```

2. Oshkor ravishda xotirani absolyut adresini berish: char *vp = (char *)0xB8000000;

bunda 0xB8000000 – o_n oltilik o_zgarmas son va (char *) – turga keltirish amali bo_lib, u vp o_zgaruvchini xotirani absolyut adresidagi baytlarni char sifatida qayta ishlovchi ko_rsatkich turiga aylantiriladi.

3. Bo_sh qiymat berish:

```
int *suxx =
NULL; int
*r=0;
```

Birinchi satrda maxsus NULL o_zgarmasi ishlatilgan, ikkinchi satrda 0 qiymat ishlatilgan. Ikkala holda ham ko_rsatkich hech qanday ob‘ektga murojaat qilmaydi. Bo_sh ko_rsatkich asosan ko_rsatkichni aniq ob‘ektga ko_rsatyotganligi yoki yo_qligini aniqlash uchun ishlatiladi.

4. Dinamik xotirada new amali bilan joy ajratish va uni adresini ko_rsatkichga berish:

```
int * n =new int;   // birinchi
operator int * m =new int(10);
// ikkinchi operator int *
q=new int [10];  // uchinchi
operator
```

Birinchi operatorda new amali yordamida dinamik xotirada int uchun yetarli joy ajratib olinib va uning adresi n ko‗rsatkichga yuklangan. Ko‗rsatkichning o‗zi uchun joy kompilyatsiya vaqtida ajratiladi.

Ikkinchi operatorda joy ajratishdan tashqari m adresiga boshlang‗ich qiymat - 10 sonini joylashtiradi.

Uchinchi operatorda int turidagi 10 element uchun joy ajratigan va uni boshlang‗ich adresi q ko‗rsatkichga berilayapti. Bu misolda, aniqrog‗i massivga joy ajratilib, unga q nomi berilgan bo‗lib, keyinchalik shu nom orqali massiv elementlariga murojaat qilish mumkin. Xotira new amali bilan ajratilgan bo‗lsa, u delete amali bilan bo‗shatilishi kerak :

delete n; delete m; delete [ ] q;

Agarda xotira new[] amali bilan ajratilgan bo‗lsa, uni bo‗shatish uchun delete [] amalini o‗lchovi ko‗rsatilmagan holda qo‗llash kerak.

Xotira bo‗shatilganligiga qaramasdan ko‗rsatkichni o‗zini qayta ishlatish mumkin.

Ko’rsatkich ustidan amallar

Ko’rsatkich ustidan quyidagi amallar bajarilishi mumkin:

1) ob‘ektga vositali murojaat qilish amali; 2) qiymat berish amali;

3) Ko’rsatkichga konstanta qiymatni qo’shish;
4) ayirish amali;
5) inkrement va dekrement amallari;
6) solishtirish amali;
7) turga keltirish amali.

Vositali murojaat qilish amali:

Bu amal ko‗rsatkichdagi adres bo‗yicha joylashgan qiymatni olish yoki qiymat berish uchun ishlatiladi:

```
shar a; // char turidagi o‗zgaruvchi e‘loni
  shar *p =new char; // ko‗rsatkichni e‘lon qilinib,
unga dinamik   // xotiradan ajratilgan xotiraning
adresi berish
*p=‘b‘; // p adresiga qiymat joylashtirish
  a =*p; // a o‗zgaruvchiga p adresiga qiymatni berish
```

Shuni qayd qilib o‗tish kerakki, xotirani bitta joyining adresini bir paytni o‗zida bir nechta va turli turdagi ko‗rsatkichlarga berish mumkin va murojaat qilganda har xil qiymatlar olish mumkin:

```
        unsigned long int A=0Xcc77ffaa;
  unsigned short int* pint = (unsigned short
int*) &A;       unsigned char*
pshar=(unsigned char *) &A;    cout<<‖*pint
=‖<<*pint<<‖ *char=‖<<*pchar;   printf(— |
%x | %x | %x | — , A, *pint, *pchar);
```

Bu misolda har xil qiymatlar chop etiladi: cc77ffaa | ffaa | aa |

Agar har xil turdagi ko‗rsatkichlarga qiymatlar berilsa, albatta turga keltirish amalidan foydalanish kerak:

```
int n=5;  float x=1.0;  int
*pi=&n;  float *px=&x;  void
*p;
int *r,*r1;  px=(float *)&n;
p=px;  r=(int *)px;  r1=pi;
```

Ko‗rsatgich turini void turiga keltirish kerak emas (amal ma‘noga ega emas). Xuddi shunday, turlari bir xil bo‗lgan ko‗rsatkichlarga uchun turni keltirish amali bajarishga hojat yo‗q.

Ko‗rsatkich ustidan bajariladigan arifmetik amallarda avtomatik ravishda turlarni o'lchovi hisobga olinadi.

Arifmetik amallar faqat bir xil turdagi ko'rsatkichlar ustidan bajariladi va asosan, massiv tuzilmalarga ko'rsatgichlar ustida bajariladi.

Inkrement amali ko'rsatkichni massivning keyingi elementiga, dekrement esa aksincha, oldingi elementga adresiga ko'chiradi. Bunda ko'rsatkichning qiymati (adres) sizeof(<massiv elementining turi>) qiymatiga o'zgaradi. Agar ko‗rsatkichning qiymati k o'zgarmas qiymatga oshirilsa yoki kamaytirilsa, u k* sizeof(<massiv elementining turi>) qiymatiga o'zgaradi. Masalan short int *p=new short [5]; long * q = new long [5]; p++; // p qiymati 2 oshadi q++; // q qiymati 4 ga oshadi q+=3; // q qiymati 3*4=12 oshadi

Ko'rsatkichlarning ayirmasi deb, ular ayirmasining tur o'lchamiga bo'linishiga aytiladi. Ko'rsatkichlarni o'zaro qo'shish mumkin emas.

Murojaatlar Murojaatlar e‘londa ko'rsatilgan nomning sinonimi sifatida ishlatiladi, yani bitta o'zgaruvchiga xar xil nom bilan murojaat qilish mumkin. Murojaatni doimiy qiymatga ega bo'lgan ko'rsatkich deb qarash mumkin xam bo'ladi. Murojaat quyidagicha e‘lon qilinadi:

<tur> & <nom>;

Bu yerda <tur> – murojaat ko‗rsatuvchi qiymatning turi, ‗&‘ belgisi, undan keyin yozilgan <nom>- murojaat turidagi nom ekanligini bildiruvchi operator. Boshqacha aytganda ‗&‘ belgisiga adresni olish amali deyiladi.

Mis
ol:

```
int
kol;
int & pal=kol; // pal murojaati - kol o'zgaruvchisining
alternativ nomi const char & cr='\n'; // cr – konstantaga
murojaat
```

Murojaatni ishlatishda quyidagi qoidalarga rioya qilish kerak: murojaat, funksiya parametri sifatida ishlatilgan, extern bilan tavsiflangan va

sinf maydoniga murojaat qilgan hollardan tashqarida barcha holatlarda boshlang’ich qiymatga ega bo’lishi kerak.

Murojaat asosan funksiyalarda adres orqali uzatiluvchi parametrlar sifatida ishlatiladi.

Murojaatni ko’rsatkichdan farqi shundaki, u alohida xotira egallamaydi u faqat o’zgaruvchining boshqa nomi sifatida ishlatiladi.

Satrlar

Standart C++ tili ikki xildagi belgilar majmuasini qo_llab–quvvatlaydi. Birinchi toifaga, an‘anaviy, ―tor‖ belgilar deb nomlanuvchi 8-bitli belgilar majmuasi kiradi, ikkinchisiga 16-bitli ―keng‖ belgilar kiradi. Til kutubxonasida har bir guruh belgilari uchun maxsus funksiyalar to_plami aniqlangan.

C++ tilida satr uchun maxsus tur aniqlanmagan. Satr char turidagi belgilar massivi sifatida qaraladi va bu belgilar ketma–ketligi satr terminatori deb nomlanuvchi nol kodli belgi bilan tugaydi (_\0‘). Odatda, nol-terminator bilan tugaydigan satrlarni ASCIIZ –satrlar deyiladi.

Sart konstanta deb qo’shtirnoqlar ichiga olingan belgilar ketma–ketligiga aytiladi:

―Ushbu belgilar ketma–ketligiga satr deyiladi.‖

Quyidagi jadvalda C++ tilida belgi sifatida ishlatilishi mumkin bo_lgan konstantalar to_plami keltirilgan.

Belgilar sinflari Belgi konstantalar

Katta harflar _A‘ …_Z‘, _A‘…‘Я‘

Kichik harflar _a‘ …_z‘, _a‘…‘я‘

Raqamlar _0‘ …_9‘

Bo_sh joy gorizontal tabulyatsiya (ASCII kodi 9), satrni o_tkazish (ASCII kodi 10), vertikal tabulyatsiya (ASCII kodi 11), formani o_tkazish (ASCII kodi 12), karetkani qaytarish
(ASCII kodi 13)

Punktuatsiya belgilari (ajratuvchilar) ! ‖ # $ & _ ( ) * + - , . / : ; < = > ? @ [\ ] ^ _ { | } ~

Boshqaruv belgilari ASCII kodi 0...1Fh oralig_ida va 7Fh bo_lgan belgilar

Probel ASCII kodi 32 bo_lgan belgi

O_n oltilik raqamlar _0‘…‘9‘, _A‘…‘F‘, _a‘…‘f‘

Satr uzunligini aniqlash funksiyalari

Satrlar bilan ishlashda, aksariyat hollarda satr uzunligini bilish zarur bo’ladi. Buning uchun string.h kutubxonasida strlen() funksiyasi aniqlangan bo’lib, uning sintaksisi quyidagicha bo’ladi:

size_t strlen (const char* string)

Bu funksiya uzunligi hisoblanishi kerak bo’lgan satr boshiga ko’rsatgich bo’lgan yagona parametrga ega va u ishlash natijasi sifatida ishorasiz butun sonni qaytaradi. strlen() funksiyasi satrning real uzunligidan bitta kam qiymat qaytaradi, ya‘ni nol-terminator o’rni hisobga olinmaydi.

Xuddi shu maqsadda sizeof() funksiyasidan ham foydalanish mumkin va u strlen() funksiyasidan farqli ravishda satrning real uzunligini qaytaradi. Quyida keltirilgan misolda satr uzunligini hisoblashning har ikkita varianti keltirilgan:

```
#include
<iostream.h>
#include
<string.h> int
main()
{
char Str[]="1234567890"; cout
<<"strlen(Str)="<<strlen(Str)<
<endl;
cout<<"sizeof(Str)="<<sizeof(
Str)<<endl; return 0;
}
```

Programma ishlashi natijasida ekranga strlen(Str)=10 sizeof(Str)=11 xabarlari chiqadi.

Odatda sizeof() funksiyasidan getline() funksiyasining ikkinchi argumenti sifati ishlatiladi va satr uzunligini yaqqol ko‗rsatmaslik imkonini beradi:

```
cin.getline(Satr, sizeof(Satr));
```

Satrlarni nusxalash

Satr qiymatini biridan ikkinchisiga nusxalash mumkin. Buning uchun bir qator standart funksiyalar aniqlangan bo'lib, ularning tavsiflari quyida keltiramiz.

strcpy() funksiyasi prototipi char* strcpy(char* str1, const char* str2) ko'rinishga ega va bu funksiya str2 ko'rsatib turgan satrdagi belgilarni str1 ko'rsatib turgan satrga baytma-bayt nusxalaydi. Nusxalash str2 ko'rsatib turgan satrdagi nol-terminal uchraguncha davom etadi. Shu sababli, str2 satr uzunligi str1 satr uzunligidan katta emasligiga ishonch hosil qilish kerak, aks holda berilgan sohasida (segmentida) str1 satrdan keyin joylashgan berilganlar ―ustiga‖ str2 satrning ―ortiqcha‖ qnomi yozilishi mumkin. Navbatdagi programma qnomi ―Satrni nusxalash!‖ satrini Str satrga nusxalaydi: char Str[20];

```
strcpy(Str, ―Satrni nusxalash!‖);
```

Zarur bo'lganda satrning qaysidir joyidan boshlab, oxirigacha nusxadash mumkin.

Masalan, - Satrni nusxalash!‖ satrini 8 belgisidan boshlab nusxa olish zarur bo'lsa, uni quyidagicha yechish mumkin:

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

```
<string.h> int
main()
{
  char Str1[20]=—Satrni nusxalash!‖;
  char Str2[20];
char*
kursatgich=Str1;
    kursatgich+=7;
strcpy(Str2, kursatgich);
    cout<<Str2<<endl;
         return 0;
        }
```

strncpy() funksiyasining strcpy() funksiyasidan farqli joyi shundaki, unda bir satrdan ikkinchisiga nusxalanadigan belgilar soni ko’rsatiladi. Uning sintaksisi quyidagi ko_rinishga ega:

```
char* strncpy(char* str1, const char* str2, size_t num)
```

Agar str1 satr uzunligi str2 satr uzunligidan kichik bo’lsa, ortiqcha belgilar —kesib‖ tashlanadi. strncpy() funksiyasi ishlatilishiga misol ko’raylik:

```
#include
<iostream.h>
#include
<string.h> int
main()
{
char Uzun_str[]="01234567890123456789";
char Qisqa_str[]="ABCDEF";
strncpy(Qisqa_str,Uzun_str,4)
; cout <<"Uzun_str=
"<<Uzun_str<<endl;
 cout<<"Qisqa_str="<<Qisqa
_str<<endl;  return 0; }
```

Programmada Uzun_str satri boshidan 4 belgi Qisqa_str satriga oldingi qiymatlar ustiga nusxalanadi va natijada ekranga

```
01234567890123456789
0123EF
```

xabarlari chop etiladi.

strdup() funksiyasiga yagona parametr sifatida satr–manbaga ko_rsatgich uzatiladi. Funksiya, satrga mos xotiradan joy ajratadi, unga satrni nusxalaydi va yuzaga kelgan satrnusxa adresini qaytaradi. strdup() funksiya sintaksisi:

```
char* strdup(const char* sourse)
```

Quyidagi programma bo_lagida satr1 satrining nusxasi xotiraning satr2 ko_rsatgan joyida paydo bo_ladi:

```
char* satr1=‖Satr
nusxasini olish.‖; char*
```

```
satr2;
satr2=strdup(satr1);
```

Satrlarni ulash

Satrlarni ulash (konkatenatsiya) amali yangi satrlarni hosil qilishda keng qo_llaniladi. Bu maqsadda string.h kutubxonasida strcat() va strncat() funksiyalari aniqlangan.

strcat( ) funksiyasi sintaksisi quyidagi ko_rinishga ega: char* strcat(char* str1, const char* str2)

Funksiya ishlashi natiajasida str2 ko_rsatayotgan satr, funksiya qaytaruvchi satr – str1 ko_rsatayotgan satr oxiriga ulanadi. Funksiyani chaqirishdan oldin str1 satr uzunligi, unga str2 satr ulanishi uchun yetarli bo'lishi hisobga olingan bo'lishi kerak.

Quyida keltirilgan amallar ketma-ketligi bajarilishi natijasida satr satriga qo_shimcha satr ostilari ulanishi ko'rsatilgan:

```
char satr[80];
strcpy(satr,‖Bu
satrga ―);
strcat(satr,‖satr osti
ulandi.‖);
```

Amallar ketma-ketligini bajarilishi natiajasida satr satri - Bu satrga satr osti ulandi.‖ qiymatiga ega bo'ladi.

strncat( ) funksiyasi strcat( ) funksiyadan farqli ravishda str1 satrga str2 satrning ko'rsatilgan uzunligidagi satr ostini ulaydi. Ulanadigan satr osti uzunligi funksiyaning uchinchi parametri sifatida beriladi. Funksiya sintaksisi char* strncat(char* str1, const char* str2, size_t num)

Pastda keltirilgan programma bo'lagida str1 satrga str2 satrning boshlang'ich 10 ta belgidan iborat satr ostini ulaydi:

```
char satr1[80]=‖Programmalash tillariga
misol bu-―; char satr2[80]=‖C++,Pascal,
Basic‖;
strncpy(satr1,satr2,10);
cout<<satr1;
```

Amallar bajarilishi natijasida ekranga ―Programmalash tillariga misol bu-
C++,Pascal‖ satri chop etiladi.

Satrlarni solishtirish

Satrlarni solishtirish ularning mos o'rindagi belgilarini solishtirish (katta yoki qichikligi) bilan aniqlanadi. Buning uchun string.h kutubxonasida standart funksiyalar mavjud. strcmp( ) funksiyasi sintaksisi

int strcmp(const char* str1, const char* str2)

ko_rinishiga ega bo_ltb, funksiya str1 va str2 solishtirish natijasi sifatida son qiymatlarni qaytaradi va ular quyidagicha izohlanadi:

<0 – agar str1 satri str2 satridan kichik bo‗lsa; =0 – agar str1 satri str2 satriga teng bo‗lsa;
>0 – agar str1 satri str2 satridan katta bo‗lsa.

Funksiya harflarning bosh va kichikligini farqlaydi. Buni misolda ko‗rishimiz mumkin: char satr1[80]=‖Programmalash tillariga bu-C++, pascal, Basic.―; char satr2[80]=‖Programmalash tillariga bu-C++, Pascal, Basic.―; int i;
i= strcmp(satr1,satr2);
Natijada i o‗zgaruvchisi musbat qiymat qabul qiladi, chunki solishtirilayotgan satrlardagi ―pascal‖ va ―Pascal‖ satr ostilarida birinchi harflar farq qiladi. Keltirilgan misolda i qiymati 32 bo‗ladi – farqlanuvchi harflar satrning 32 elementi hisoblanadi. Agar funksiyaga i= strcmp(satr2,satr1); ko‗rinishida murojaat qilinsa i qiymati –32 bo‗ladi.

Agar satrlardagi bosh yoki kichik harflarni farqlamasdan solishtirish amalini bajarish zarur bo'lsa, buning uchun stricmp() funksiyasidan foydalanish mumkin. Yuqorida keltirilgan misoldagi satrlar uchun i=strcimp(satr2,satr1); amali bajarilganda i qiymati 0 bo‗ladi. strncmp( ) funksiyasi sintaksisi int strncmp(const char* str1, const char* str2, size_t num) ko‗rinishida bo'lib, str1 str2 satrlarni boshlang‗ich num sonidagi belgilarini solishtiradi. Funksiya harflar registrini inobatga oladi. Yuqorida misolda aniqlangan satr1 va satr2 satrlar uchun i=strncimp(satr1,satr2,31); amali bajarilishida i qiymati 0 bo'ladi, chunki satrlar boshidagi 31 belgilar bir xil.

strnicmp( ) funksiyasi strncmp( ) funksiyasidek amal qiladi, farqli tomoni shundaki, solishtirishda harflarning registrini hisobga olinmaydi. Xuddi shu satrlar uchun i=strcnimp(satr1,satr2,32); amali bajarilishi natijasida i o‗zgaruvchi qiymati 0 bo‗ladi.

Satrdagi harflar registrini almashtirish

Berilgan satrdagi kichik harflarni bosh harflarga yoki teskari almashtirishga mos ravishda _strupr( ) va _strlwr( ) funksiyalar yordamida amalga oshirish mumkin. Kompilyatorlarning ayrim variantlarida funksiyalar nomidagi tagchiziq (‗_') bo‗lmasligi mumkin.

_strlwr( ) funksiyasi sintaksisi
char* _strlwr(char* str)

Ko'rinishida bo'lib, argument sifatida berilgan satrdagi bosh harflarni kichik harflarga almashtiradi va hosil bo'lgan satr adresini funksiya natijasida qaytaradi. Quyidagi programma bo‗lagi _strlwr( ) funksiyasidan foydalnishga misol bo‗ladi.

```
char str[]=‖10 TA KATTA HARFLAR‖;
_strlwr(str);
cout<<str;
```

Natijada ekranga ―10 ta katta harflar‖ satri chop etiladi.

_strupr( ) funksiyasi xuddi _strlwr( ) funksiyasidek amal qiladi, lekin satrdagi kichik harflarni bosh harflarga almashtiradi:

char str[]=―10 ta katta harflar‖;

```
_strupr(str);
cout<<str;
```

Natijada ekranga ‖10 TA KATTA HARFLAR‖ satri chop etiladi.

Programmalash amaliyotida belgilarni qaysidir oraliqqa tegishli ekanligini bilish zarur bo_ladi. Buni ctype.h sarlavha faylida e‘lon qilingan funksiyalar yordamida bilsa bo_ladi. Quyida ularning bir qnomining tavsifi keltirilgan:

isalnum( ) – belgi raqam yoki harf (true) yoki yo_qligini (false) aniqlaydi;

isalpha( ) – belgini harf (true) yoki yo_qligini (false) aniqlaydi;

isascii( ) – belgini kodi 0..127 oralig_ida (true) yoki yo_qligini (false) aniqlaydi; isdigit( ) – belgini raqamlar diapazoniga tegishli (true) yoki yo_qligini (false) aniqlaydi.

Bu funksiyalardan foydalanishga misol keltiramiz.

```
#include <iostream.h>
#include
<ctype.h>
#include
<string.h> int
main()
{
        char satr[5];
 do
          {
              cout<<"Tug\'ilgan yilingizni kiriting, marhamat...";
       cin.getline(satr,5);
           if(isalpha(satr[0]))
              {
                cout<<"Siz harf kiritdingiz !";
                continue;
               }
           if(iscntrl(satr[0]))
              {
                cout<<"Siz boshqaruv belgilarini
kiritdingiz !";                  continue;
               }
             if(ispunct(satr[0]))
              {
                cout<<"Siz punctuatsiya belgilarini kiritdingiz !";
               continue;
               }
          for (int i=0; i<=strlen(satr); i++)
             {
               if (!isdigit(satr[i]))
continue;                  else
                {
```

```
                    cout << "Sizni tug\'ilgan yilingiz:
"<<satr;                    return 0;
                  }
                }
}
while (1);
}
```

Programada foydalanuvchiga tug'ilgan yilini kiritish taklif etiladi. Kiritilgan sana satr o'zgaruvchisiga o'qiladi va agar satrning birinchi (satr[0]) belgisi harf yoki boshqaruv belgisi yoki punktuatsiya belgisi bo'lsa, shu haqda xabar beriladi va tug'ilgan yilni qayta kiritish taklif etiladi. Programma tug_ilgan yil (to'rtta raqam) to'g'ri kiritilganda "Sizni tug'ilgan yilingiz: XXXX‖ satrini chop qilish bilan o'z ishini tugatadi.

Satrni teskari tartiblash

Satrni teskari tartiblashni uchun strrev() funksiyasidan foydalanish mumkin. Bu funksiya quyidagicha prototipga ega: char* strrev(char* str)

Satr reversini hosil etishga misol:

```
char str[]=‖telefon‖;        cout
<<strrev(str);
```

amallar bajarilishi natijasida ekranga ―nofelet‖ satri chop etiladi.

Satrda belgini izlash funksiyalari

Satrlar bilan ishlashda satrda birorta belgini yoki satr ostini izlash masalasi nisbatan ko'p uchraydi. Bu turdagi masalalaru uchun string.h kutubxonasida bir qator standart funksiyalar mavjud.

Satrda belgi bor yoki yo'qligini aniqlab beruvchi strchr() funksiyasining prototipi char* strchr(const char* string, int c)

Ko'rinishida bo'lib, u s belginining satr string satrida izlaydi. Agar izlash muvofaqqiyatli bo'lsa, funksiya shu belgining satrdagi o_rnini (adresini) funksiya natijasi sifatida qaytaradi, aks holda, ya'ni belgi satrda uchramasa funksiya NULL qiymatini qaytaradi. Belgini izlash satr boshidan boshlanadi.

Quyida keltirilgan programma bo_lagi belgini satrdan izlash bilan bog_liq.

```
char satr[]=‖0123456789‖;
char* pSatr;
pSatr=strchr(satr,'6');
```

Programma ishlashi natijasida pSatr ko'rsatgichi satr satrining _6' belgisi joylashgan o'rni adresini ko_rsatadi.

strrchr() funksiyasi berilgan belgini (s) berilgan satr (string) oxiridan boshlab izlaydi. Agar izlash muvoffaqiyatli bo'lsa, belgini satrga oxirgi kirishining o'rnini qaytaradi, aks holda NULL.

Misol uchun

char
satr[]=‖0123456789101112‖;
char* pSatr;
pSatr=strrchr(satr,‘0‘);
amallarini bajarilishida pSatr ko’rsatgichi satr satrining _01112‘ satr ostining
boshlanishiga ko’rsatadi.

strspn() funksiyasi ikkita satrni mos o’rindagi belgilarni solishtiradi va birinchi ustma-ust tushmagan o_rnini aniqlab beradi (registrni hisobga olgan holda). Funksiya quyidagi ko’rinishdagi prototipga ega:

size_t strspn(const char* string, const char* group)

Funksiya qaytaruvchi qiymatiga boshqacha mazmun berish mumkin – funksiya ikkita satrda ustma-ust tushadigan elementlar sonini beradi:

char satr1[]=‖0123456789101112‖;
char
satr2[]=‖012345678901234567
8‖; int mos_belgilar;
mos_belgilar=strspn(satr1,satr2);
cout<<‖Satrlardagi mos tushgan belgilar soni= —<<mos_belgilar;
amallar bajarilishi natijasida ekranga ‖Satrlardagi mos tushgan belgilar soni= 10— satri
chop etiladi. strcspn() funksiyasi
prototipi
size_t strcspn(const char* str1, const char* str2)

ko_rinishida bo_lib, u str1 va str2 satrlarni solishtiradi va str1satrining str2 satriga kirmaydigan satr ostining uzunligini beradi. Boshqacha aytganda, funksiya satrlarning birinchi kesishish o_rnini qaytaradi. Masalan char satr[]=‖Birinchi satr‖; int index;
index=strcspn(satr,‖sanoq tizimi‖);
amallar bajarilgandan keyin index o_zgaruvchisi 9 qiymatini qabul qiladi, chunki 9
joydagi belgi ikkinchi satrning birinchi belgisi bilan mos tushadi.

strpbrk() funksiyasi prototipi
char* strpbrk(const char* str1, const char* str2)

ko_rinishga ega bo_lib, u str1 satrdagi str2 satrga kiruvchi birorta belgini izlaydi va agar bunday element topilsa, uning adresi funksiya qiymati sifatida qaytariladi, aks holda funksiya NULL qiymati qaytaradi. Quyidagi misol funksiyani qanday ishlashini ko_rsatadi.

char satr1[]=‖0123456789ABCDEF‖;
char satr2[]=‖ZXYabcdefABC‖;
char* element;
element = strpbrk(satr1,satr2);
cout<<element<<‘\n‘;
Programma ishlashi natijasida ekranga str1 satrining —ABCDEF‖ satr ostisi chop etiladi.

8-MAVZU. C++ DA STRUKTURALAR VA BIRLASHMALAR

Reja:

1. C++ da strukturalar va ko‘rsatkichlar.
1. C++ da strukturalarga ko‘rsatkichlar ustida amallar.
2. C++ da birlashmalar.

Strukturali tip.

Struktura bu turli tipdagi ma'lumotlarning birlashtirilgan tipdir. Struktura har hil tipdagi elementlar-komponentalardan iborat bo‘ladi. Strukturalar quyidagicha ta'riflanishi mumkin:

Struct struturali_tip_nomi
{Elementlar_ta'riflari}

Misol uchun ombordagi mollarni tasvirlovchi strukturani quramiz. Bu struktura quyidagi komponentalarga ega bo‘lishi mumkin:

- Mol nomi (char*)
- Sotib olish narhi (long)
- Ustiga quyilgan narh, foizda (float)
- Mol soni (int)
- Mol kelib tushgan sana (char[9])

Bu struktura dasturda quyidagicha ta'riflanadi:
struct goods { char* name; long price;
float percent; int vol; char date[9]; }
year;

Konkret strukturalar va strukturaga ko‘rsatkichlar bu tip yordamida quyidagicha ta'riflanishi mumkin:

Struct goods food, percon; struct goods *point_to;

Strukturalarni tasvirlashda ihtiyoriy murakkab tip uchun nom berishga imkon beruvchi typedef hizmatchi so‘zidan foydalanish mumkin. Bu holda strukturali tip quyidagi shaklda ta'riflanadi:

Typedef struct {Elementlar_ta'riflari}
strukturali_tip_nomi Misol uchun: Typedef
struct
{ double real; double imag;
} complex;

Bu misolda kompleks sonni tasvirlovchi strukturali tip complex kiritilgan bo‘lib, kompleks son haqiqiy qismini tasvirlovchi real va mavhum qismini tasvirlovchi komponentalaridan iboratdir. Konkret strukturalar bu holda quyidagicha tasvirlanadi:

Complex sigma,alfa;

Strukturali tip typedef yordamida aniqlangan nomdan tashqari, standart usulda aniqlangan nomga ega bo‘lishi mumkin. Quyidagi misolda kasr sonni tasvirlovchi numerator – sur'at va denominator-mahraj komponentalaridan iborat struktura ta'rifi keltirilgan.

```
Typedef struct  rational_fraction
{ int
numerator;
int
denominat
or; }
fraction;
```

Bu misolda fraction kasrning Typedef orqali kiritilgan nomi, rational_fraction standart usulda kiritilgan nom. Bu holda konkret strukturalar quyidagicha tasvirlanishi mumkin: Struct rational_fraction alfa; fraction beta;

KONKRET STRUKTURALARNI TASVIRLASH.

Yuqoridagi misollarda konkret strukturalarni ta'riflashni ikki usuli ko‘rib chiqilgan. Agar strukturali tip standart usulda kiritilgan bo‘lsa konkret strukturalar quyidagi shaklda ta'riflanadi:

Struct < struktura nomi> <konkret strukturalar ruyhati>

Masalan Struct goods food

Agar strukturali tip typedef hizmatchi so‘zi yordamida kiritilgan bo‘lsa konkret strukturalar quyidagi shaklda ta'riflanadi:

< struktura nomi> <konkret strukturalar ruyhati>

Masalan Complex sigma

Bu usullardan tashqari konkret strukturalarni ta'riflashning boshqa usullari ham mavjuddir. Strukturalar ta'riflanganda konkret strukturalar ruyhatini kiritish mumkin:

```
Struct struturali_tip_nomi
{Elementlar_ta'riflari}
Konkret_strukturalar_ruyhati.
```

Misol:

```
Struct student
{ char
name[15];
char
surname[2
0];
int year;
} student_1, student_2, student_3;
```

Bu holda student strukturali tip bilan birga uchta konkret struktura kiritiladi. Bu strukturalar student ismi (name[15]), familiyasi (surname[20]), tugilgan yilidan (year) iborat.

Strukturali tip ta'riflanganda tip nomi ko‘rsatilmay, konkret st'rukturalar ruyhati ko‘rsatilishi mumkin:

```
Struct
{Elementlar_ta'riflari}
Konkret_strukturalar_ruyhati.
```

Quyidagi ta'rif yordamida uchta konkret struktura kiritiladi, lekin strukturali tip kiritilmaydi. struct { char processor [10]; int frequency;

```
int
memory
; int
disk;
} IBM_486, IBM_386, Compaq;
```

STRUKTURALAR UCHUN HOTIRADAN JOY AJRATISH.

Strukturali tip kiritilishi bu tip uchun hotiradan joy ajratilishiga olib kelmaydi. Har bir konkret struktura (ob'ekt) ta'riflanganda, shu ob'ekt uchun elementlar tiplariga qarab hotiradan joy ajratiladi. Hotiradan joy zich ajratilganda struktura uchun ajratilgan joy hajmi har bir element uchun zarur bo‘lgan hotira hajmlari yig‘indisiga teng bo‘ladi. Shu bilan birga hotiradan joy zich ajratilmasligi ham mumkin ya'ni elementlar orasida bo‘sh joylar ham qolishi mumkin. Bu bo‘sh joy keyingi elementni hotira qismlarining qabul qilingan chegaralari bo‘yicha tekislash uchun qoldiriladi. Misol uchun butun tipdagi elementlar juft adreslardan boshlansa, bu elementlarga murojaat tezroq amalga oshiriladi. Konkret strukturalarni joylashuviga ba'zi kompilyatorlarda #pragma preprotsessor direktivasi yordamida ta'sir o‘tkazish mumkin. Bu direktivadan quyidagi shaklda:

Pragma pack(n)

Bu erda n qiymati 1,2 eki 4 ga teng bo‘lishi mumkin.

Pack(1) – elementlarni bayt chegaralari bo‘yicha tekislash; Pack(2) – elementlarni so‘zlar chegaralariga qarab tekislash;

Pack(4) – elementlarni ikkilangan muzlar chegaralariga qarab tekislash.

Struktura uchun ajratilgan joy hajmini quyidagi amallar yordamida aniqlash mumkin:

```
Sizeof
(strukturali_tip_nom
i);          Sizeof
(struktura_nomi);
Sizeof
struktura_nomi.
```

Ohirgi holda struktura nomi ifoda deb qaraladi. Ifodaning tipi aniqlanib, hajmi hisoblanadi.

Misol uchun:

```
Sizeof (struct goods)
Sizeof (tea)
Sizeof coat
```

STRUKTURALARGA MUROJAAT.

Konkret strukturalar ta'riflanganda massivlar kabi initsializatsiya qilinishi mumkin.

Masalan complex

sigma {1.3;12.6};

Struct goods coats={—pidjak‘,40000,7.5,220,‖12.01.97‖);

Bir hil tipdagi strukturalarga kiymat berish amalini kullash mumkin:

Complex alfa; alfa=sigma;

Lekin strukturalar uchun solishtirish amallari aniqlanmagan.

Strukturalar elementlariga quyidagicha murojaat qilish mumkin: Struktura nomi.element_nomi.

Nuqta amali‘ struktura elementiga murojaat qilish amali deyiladi. Bu amal qavs amallari bilan birga eng yuqori ustivorlikka egadir.

Misol:

Complex alfa={1.2,-4.5},betta={5.6,-7.8),sigma;

Sigma.real=alfa.real+betta.real;

Sigma.imag=alfa.imag+betta.imag;

Konkret strukturalar elementlari dasturda alohida kiritilishi va chiqarilishi zarurdir. Quyidagi misolda ikki kompleks son qiymatlari kiritilib, yigindisi hosil qilinadi:

```
#include
<iostream.h>
typedef struct {
double real;
double
imag; }
comple
x; void
main()
{
comple
x x,y,z;
Cout<<(—\n                :‖);Cin>>(—%f‖,&x.real);
Cout<<(—\n                :‖);Cin>>(—%f‖,&x.imag);
Cout<<(—\n                :‖);Cin>>(—%f‖,&y.real);
Cout<<(—\n                :‖);Cin>>(—%f‖,&y.imag);
z.real=x.real+y.real;
z.imag=x.imag+y.imag;
Cout<<(—\n            %f‖,&z.real);
Cout<<(—\n            %f‖,&z.imag);
}
```

STRUKTURALAR VA MASSIVLAR.

Massivlar strukturalar elementlari sifatida.

Massivlarni strukturalar elementi sifatida ishlatilishi hech qanday qiyinchilik tug‘dirmaydi. Biz yuqorida simvolli massivlardan foydalanishni ko‘rdik. Quyidagi misolda fazoda berilgan nuqtaviy jismni tasvirlovchi komponentalari jism massasi va koordinatalaridan iborat struktura kiritilgan bo‘lib, nuqtaning koordinatalar markazigacha bo‘lgan masofasi hisoblangan.

```
Include <iostream.h> #include <math.h> void main()
{ struct
 {
double mass; float coord[3]
} point={12.3,{1.0,2.0,-3.0}}; int i; float s=0.0; for (i=0;i<3; i++)
        s+=point.coord[i]*point.coord[i];
        Cout<<(—\n
        masofa=%f‖,sqrt(s));
        }
```

Bu misolda point strukturasi nomsiz strukturali tip orqali aniqlangan bo‘lib, qiymatlari initsializatsiya yordamida aniqlanadi.

Strukturalar massivlari.

Strukturalar massivlari oddiy masivlar kabi tasvirlanadi. Yuqorida kiritilgan strukturali tiplar asosida quyidagi strukturalar massivlarini kiritish mumkin:

```
Struct goods list[100];
Complex set [80];
```

Bu ta'riflarda list va set strukturalar nomlari emas, elemenlari strukturalardan iborat massivlar nomlaridir. Konkret strukturalar nomlari bo‘lib set[0],set[1] va hokazolar hizmat qiladi. Konkret strukturalar elementlariga quyidagicha murojaat qilinadi: set[0].real– set massivi birinchi elementining real nomli komponentasiga murojaat.

Quyidagi misolda nuktaviy jismlarni tasvirlovchi strukturalar massivi kiritiladi va bu nuqtalar sistemasi uchun og‘irlik markazi koordinatalari (xc,yc,zc) hisoblanadi. Bu koordinatalar quyidagi formulalar asosida hisoblanadi:

```
m=?mi;  xc =(?ximi )/m; yc =(?yimi )/m; zc =(?zimi )/m;

#Include <iostream.h>
struct
particle {
double
mass;
float
coord [3];
};

struct particle mass_point[]={ 20.0, {2.0,4.0,6.0}
                               40.0, {6.0,-2.0,6.0}
```

```
                                    10.0,
{1.0,3.0,2.0}
}; int N; struct particle center ={ 0.0,
{0.0,0.0,0.0}

} int I;
N=sizeof(mass_point)/sizeof(mass_point[0]);
For (I=0;I<N;I++)
{
center.mass+=mass_point[I].mass
center.coord[0]+=  mass_point[I].coord[0]*
mass_point[I].mass;       center.coord[1]+=
mass_point[I].coord[1]* mass_point[I].mass;
center.coord[2]+=  mass_point[I].coord[2]*
mass_point[I].mass;       }
Cout<<(—\n Koordinatih
tsentra mass:‖); for
(I=0;I<3;I++)
{
center.coord[I]/=center.mass;
Cout<<(—\n Koordinata %d:%f‖,(I+1),center.coord[I]);
}
}
```

Strukturalar va ko‘rsatkichlar.

Strukturaga ko‘rsatkichlar oddiy ko‘rsatkichlar kabi tasvirlanadi:

```
Complex *cc,*ss; struct goods *p_goods;
```

Strukturaga ko‘rsatkich ta'riflanganda initsializatsiya qilinishi mumkin. Misol uchun ekrandagi rangli nuktani tasvirlovchi quyidagi strukturali tip va strukturalar massivi kiritiladi. Strukturaga ko‘rsatkich qiymatlari initsializatsiya va qiymat berish orqali aniqlanadi:

```
Struct point {int color;   int  x,y; } a,b;
 struct point *pa=&a,pb; pb=&b;
```

Ko‘rsatkich orqali struktura elementlariga ikki usulda murojaat qilish mumkin. Birinchi usul adres bo‘yicha qiymat olish amaliga asoslangan bo‘lib quyidagi shaklda qo‘llaniladi:

(* strukturaga ko‘rsatkich).element nomi;

Ikkinchi usul mahsus strelka (->) amaliga asoslangan bo‘lib quyidagi ko‘rinishga ega: strukturaga ko‘rsatkich->element nomi

Struktura elementlariga quyidagi murojaatlar o‘zaro tengdir:

(*pa).color==a.color==pa->color

Struktura elementlari qiymatlarini ko‘rsatkichlar yordamida quyidagicha o‘zgartirish mumkin:

```
(*pa).color=red
; pa->x=125;
pa->y=300;
```

Dasturda nuqtaviy jismni tasvirlovchi particle strukturali tipga tegishli m_point strukturasi aniqlangan bo‘lsin. Shu strukturaga pinta ko‘rsatkichini kiritamiz: Struct particle * pinta=&m_point;

Bu holda m_point struktura elementlarini quyidagicha o‘zgartirish mumkin:

```
Pinta->mass=18.4;
For (I=0;I<3;I++)
Pinta->coord[I]=0.1*I;
```

3. Strukturalarga ko‘rsatkichlar ustida amallar.

Strukturalarga ko‘rsatkichlar ustida amallar oddiy ko‘rsatkichlar ustida amallardan farq qilmaydi. Agar ko‘rsatkichga strukturalar massivining biror elementi adresi qiymat sifatida berilsa, massiv buyicha uzluksiz siljish mumkin bo‘ladi. Misol tariqasida kompleks sonlar massivi summasini hisoblash masalasini ko‘rib chiqamiz:

```
#include
<iostream.h> void
main()
{
sruct
complex
{float x;
  float y;} array[]={1.0,2.0,3.0,-4.0,-5.0,-6.0,-7.0,-8.0};
struct complex summa={0.0,0.0};
struct complex
*point=&array[0]; int k,I;
k=sizeof(array)/sizeof(array[0]);
for(i=0;i<k;i++)
{
summa.x+=point->x;
summa.y+=point->y;
poi
nt+
+; }
Cout<<(—\n Summa: real=%f‖,\t imag=%f‖,summa.x,summa.y);
}
```

Dastur bajarilishi natijasi:

```
Summa: real=-8.000000, imag=-16.000000
```

BIRLASHMALAR.

Strukturalarga yaqin tushuncha bu birlashma tushunchasidir. Birlashmalar union hizmatchi so‘zi yordamida kiritiladi. Misol uchun union {long h; int I,j; char c[4]}UNI;

Birlashmalarning asosiy hususiyat shundaki uning hamma elementlari bir hil boshlangich adresga ega bo‘ladi.

Quyidagi dastur yordamida bu hususiyatni tekshirish mumkin:

```
#include
<iostream.h> void
main()
{  union {long h; int k; char c[3]}U={10l;-3;‖ALI‖);
Cout<<(—\n l=%d‖;&u.l);
Cout<<(—\n k=%d‖;&u.k);
Cout<<(—\n c=%d‖;&u.c);
};
```

Birlashmalarning asosiy avfzalliklaridan biri hotira biror qismi qiymatini har hil tipdagi qiymat shaklida qarash mumkindir. Misol uchun quyidagicha birlashma union {float f; unsigned long k; char h[4];}fl;

Hotiraga fl.f=2.718 haqiqiy son yuborsak uning ichki ko‘rinishi kodini fl.l yordamida ko‘rishimiz, yoki alohida baytlardagi qiymatlarni fl.h[0]; fl.h[1] va hokazo yordamida qo‘rishimiz mumkin.

Birlashmalar imkoniyatlarini ko‘rsatish uchun bioskey() funksiyasidan foydalanishni ko‘rib chiqamiz. Bu funksiya bios.h sarlavhali faylda joylashgan bo‘lib, quyidagi prototipga ega:

int bioskey(int);

MS DOS operatsion tizimida ihtiyoriy klavishaning bosilishi klaviatura buferiga ieei bayt ma'lumot yozilishiga olib keladi.

Agar funksiyaga bioskey(0) shaklda murojat qilinsa va bufer bo‘sh bo‘lsa biror klavishaga bosilishi kutiladi, agar bufer bo‘sh bo‘lmasa funksiya buferdan ikki baytli kodli o‘qib butun son sifatida qaytaradi. Funksiyaga bioskey(0) shaklda murojat qilinsa va bufer bo‘sh bo‘lsa biror klavisha bosilishi kutiladi, agar bufer bo‘sh bo‘lmasa funksiya buferdagi navbatdagi kodni qaytaradi. Funksiyaga bioskey(1) shaklda murojat qilish bufer bush yoki bo‘shmasligini aniqlashga imkon beradi. Agar bufer bo‘sh bo‘lmasa funksiya buferdagi navbatdagi kodni qaytaradi, lekin bu kod buferdan o‘chirilmaydi.

9-MAVZU: C++ TILIDA FAYLLAR BILAN ISHLASH

Reja:

1. C++ tilida fayllar. 2. C++ tilida oqimli chiqarish va kiritish.
3. Fayllar ustida amallar bajarish.
4. Satrlar yordamida fayllar bilan bog‘lanish.
5. Fayllar bilan formatli almashinuv.

C ++ tilining asosiy hususiyatlaridan biri oldindan rejalashtirilgan fayllar strukturasi yo`qligidir. Hamma fayllar, baytlar ketma-ketligi deb ko‘riladi.

UNIX operatsion sistemasida har bir qurilmaga «Mahsus fayl» mos keladi, shuning uchun C ++ bibliotekasidagi funksiyalar fayllar bilan ham, qurilmalar bilan ham ma'lumot almashinishi uchun foydalaniladi. C ++ tili bibliotekasida kiritish – chiqarish, quyi darajadagi kiritish, chiqarish va portlar uchun kiritish – chiqarish, oqimli daraja tizim hususiyatlariga bog‘lik bulishi uchun bu yerda qaralmaydi.

Oqimli chiqarish va kiritishda ma'lumotlar bilan almashish baytma-bayt amalga oshiriladi. Lekin tashki hotira qurilmalari bilan almashish oldidan belgilangan ma'lumotlar bloki orqali amalga oshiriladi odatda u blokning minimal hajmi 512 yoki 1024 baytga teng bo‘ladi.

Diskga o‘qilishda ma'lumotlar operatsion qatordagi buferi yoziladi so‘ngra baytma bayt buferga yig‘iladi, so‘ngra diskka har bir murojaat qilinganda yagona blok sifatida uzatiladi. Shuning uchun ma'lumot almashishi diskka to‘g‘ridan to‘g‘ri murojaat qilishiga ko‘ra tezroq amalga oshadi. Shunday qilib oqim bu bu buferlash vositalari va fayldir.

Oqim bilan ishlashda quyidagi vazifalarni bajarish mumkin.

- Oqimlarni ochish va yopish.
- Simvol, qator satr ,formatlangan ma'lumot ihtiyoriy uzunlikdagi ma'lumotlarni

kiritish yoki chiqarish va fayl ohiriga etganlik shartini tahlil qilish;

- Buferlash va bufer hajmini boshqarish;
- Ko‘rsatkich oqimdagi o‘rnini aniqlash yoki yangi o‘ringa ko‘chirish.

Bu vazifalarni boshqaruvchi funksiyalar teng foydalanish dasturiga Stdio.h – faylini ulash lozim.

Dastur bajarilishi boshlanganda avtomatik ravishda 5 ta oqim ochilib, bulardan:

- Standart kiritish oqimi stdin;
- Standart chiqarish oqimi stdout;
- Hatolar haqida malumotlar standart oqimi stderr;

Oqimlarni ochish va yopish

Oqim ochilishi uchun, oldindan kiritilgan FILE tipidagi struktura bilan boglash lozimdir. FILE strukturasi ta'rifi iostream.h bibleotekasida joylashgan. Bu strukturada buferga ko‘rsatkich, o‘qilayotgan pozitsiyaga ko‘rsatkich va boshqa ma'lumotlar saqlanadi. Oqim ochilganda dasturda oqimga ko‘rsatkich ya'ni FILE

strukturali tipdagi ob'ektga ko‘rsatkich qaytariladi. Bu ko‘rsatkich quyidagicha e'lon qilinishi lozim.

FILE * <kursatkich nomi>

Misol uchun FILE * fp

Oqim ochish funksiyasi quyidagi ko‘rinishga ega;

<oqimga ko‘rsatkich nomi>=foren(<fayl-nomi>,<ochish rejimi>)

Misol uchun:fp=fopen(—t.tnt‖, —r‖)

Oqim bilan bog‘lik faylni quyidagi rejimlarda ochish mumkin:

— w‖- Yangi fayl o‘qish uchun ochiladi. Agar fayl mavjud bo‘lmasa yangidan yaratiladi.

—r‖ - Mavjud fayl faqat o‘qish uchun ochiladi.

—a‖ - Fayl da'vom ettirish uchun ochiladi.

—wt‖ - Fayl yozish va keyingi tahrirlash uchun ochiladi. Fayl ihtiyoriy joyidan o‘qish yoki yozish mumkin.

—rt‖- fayl ihtiyoriy joyidan o‘qish yoki yozish mumkin, lekin fayl ohiriga qo‘shish mumkin emas.

—at‖ - Fayl ihtiyoriy joyidan o‘qish va yozish uchun ochiladi —wt‖ rejmdan farqli fayl ohiriga ma'lumot qo‘shish mumkin.

Oqimdan o‘qilgan quyidagi simvollar -------

CR(13)-naryat nomi qaytarish

RF(10)-—yangi qator‖ boshiga o‘tish bitta simvolga —\n‖ (10)

fqkfynbhbkflb/

Agar fayl------- emas ihtiyoriy bulsa, binar rejimda ochiladi. Buning uchun rejimlar-------- harfi qo‘shiladi ------- —wb‖ yoki —rtb‖. Ba'zi ---- - matnli rejim t harifi yordamida ko‘rsatiladi masalan —yoki‖rt‖.

Oqim ochilganda quyidagi hatolar kelib chiqishi mumkin:ko‘rsatilgan fayl mavjud emas(o‘kish rejimida); disk to‘la yoki yozishdan himoyalangan va hokazo. Yana shuni aytish kerakki fopen() funksiyasi bajarilganda dinamik hotira ishlatiladi. Agar hotirada joy qolmagan bo‘lsa —not enough ‖ - hatosi kelib chiqadi.

Ko‘rsatilgan hollarda ko‘rsatkich ~ NULL qiymatga ega bo‘ladi.

Bu hatolar haqidagi ma'lumotlarni ekranga chiqarish uchun perror () funksiyasi ishlatiladi. Bu funksiya iostream.h bibliotekasida saqlanuvchi prototipi quyidagi ko‘rinishga ega.:

Void perror(court char * s);

Diskda ochilgan fayllarni berkitish uchun quyidagi funksiyadan foydalaniladi.

Int fellove(<oqimga-kursatkich nomi>).

Fayllar bilan ishlashning bitli rejimi.

Fayl bilan bitli almashish rejimi getc( ) va putc( ) funksiyalari yordamida tashkil etiladi. Bu funksiyalarga quyidagi shaklda murojat etiladi:

C=getc(fp);

Putc(c,fp);

Bu erda fp-
ko‘rsatkich S-int
tipidagi
o‘zgaruvchi

Misol tariqasida klaviaturadan simvol kiritib faylga yozishni ko‘ramiz. Matn ohirini _#‘ belgisi ko‘rsatadi. Fayl nomi foydalanuvchidan so‘raladi. Agar <enter> klavishasi bosilsa faylga CR va LF (qiymatlari 13 va 10) konstantalar yoziladi. Keyinchalik fayldan simvollarni uqishda bu konstantalar satrlarni ajratishga imkon beradi.

```
#include <iostream.h>
int main() {
file *fp;
char c;
const char CR=‘\015‘;
const char LF=‘\012‘;
char f name [20];
puts(—fayl nomini kiriting:\n‖);
gets(f name); if((fp=f open(f
name, —w‖)) ==null)
{ perror(f name);
return 1; }
while ((c=getchar())!=‘#‘)
}
if (c==‘\n‘) {
putc(CR,fp);
putc(LF,fp);
}
else putc (c,fp);
}
Fclose (fp);
Return 0;
```

Keyingi dastur fayldan simvollarni o‘qib ekranga chiqaradi.

```
#include <iostream.h>
int main() { file *fp; char c; char
f name [20]; puts(—fayl nomini
kiriting:\n‖); if((fp=f open (f
name, —r‖)) ==null)
{ perror(f name);
return 1; }
while
((c=getc(fp))!=eof)
putchar(c); f close (fp);
return 0; }
```

Satrlar yordamida fayllar bilan bog‘lanish.

Matnli fayllar bilan ishlash uchun fget va fputs funksiyalaridan foydalaniladi. Bu funksiyalari prototiplari iostream.h faylida quyidagi ko‘rinishga ega:

Int fputs (const char *s, FILE *stream);

Char *fgets (char * s, int n, FILE * stream);

Fputs funksiyasi _\0‘ simvoli bilan chegaralangan satrni stream ko‘rsatkichi orqali aniqlangan faylga yozadi. _\0‘ simvoli faylga yozilmaydi.

Fgets() funksiyasi stream ko‘rsatkichi orqali aniqlangan fayldan (n-1) simvolni o‘qiydi va S ko‘rsatgan satrga yozib qo‘yadi. Funksiya n-1 simvolni o‘qib bo‘lsa eki 1-chi qator simvoli

_\n‘ni uchratsa ishini tuhtatadi. Har bir satr ohiriga qo‘shimcha \0 belgisi qo‘shiladi. Hato bo‘lganda yoki fayl ohiriga etganda agar fayldan birorta simvol o‘qilmagan bo‘lsa NULL qiymat qaytariladi. Quyidagi dasturda bir fayldagi matnni ikkinchi faylga yozishni ko‘rib chiqamiz. Bu misolda yana bir imkoniyat komanda qatoridan dasturga ma'lumot uzatish imkoniyati ko‘rib chiqilgan. Har qanday dastur operatsion sistemada ma'lumotni argc va argv parametrlar qiymati sifatida oladi. Birinchi dasturga uzatilayotgan satrlar sonini ko‘rsatadi.

Argv[0] bu faylning nomini saklovchi satrga ko‘rsatkich massivining qolgan elementlari argv[10]…argv[argc-1] komanda qatorida fayl nomidan so‘ng bo‘shlik tashlab yozilgan parametrlarga ko‘rsatkichlar.

Dasturmiz nomi coryfile.exe bo‘lsin va bu dastur yordamida f1.dat Faylni f2.dat faylga yozmoqchimiz. Komanda qatori quyidagi ko‘rinishga ega:

<copyfile.exe f1.dat f2.txt Dastur matni: #include <iostream.h> main (int argc, char*argv[])

```
{ char cc[256];
FILE *f1, *f2;
If (argc!=3)
{ print (—\n Format bazovih dastur:‖); print f (—\n copyfile.exe‖)
Cout<< (—\n Fayl netosnihh Fayl priemnik‖); return 1;
} if ((f1=fopen(argv[1],‖r‖))==NULL)
{perror(argv[1]); return 1; } if ((f2=fopen(argv[2], —w‖))==NULL)
{perror(arg[2]); return 1; }
while (fgets(cc,256,f1)!=NULL)
fputs(CC,f2); fclose(f1); fclose(f2); return 0; }
```

Bu dastur bajarilishi natijasida int.dat fayliga Cout<< funksiyasi yordamida monitorga qanday chiqsa shunday ko‘rinishda ma'lumotlar yozadi. Keyingi misolda fayldan monitorga o‘qishni kuramiz:

```
#include <iostream.h>
int main()
{
FILE *fp;
Intn,nn,I;
If((fp=fopen(—int.dat‖,‖r‖))==NULL)
{perror (—int.dat‖); return 1; }
```

```
for(i=1; i<11;i++)
{fCin>>(fp,‖%d‖,&n)       ; Cout<<(—%d \n‖,n);
} fclose(fp); return 0;
}
```

FAYLLAR BILAN FORMATLI ALMASHINUV.

Ko‘p hollarda ma'lumotni tug‘ridan-tug‘ri monitorga chiqarishga qo‘lay shaklda faylda saqlash zarur bo‘ladi. Bu holda faylga formatli kiritish va chiqarish funksiyalaridan foydalanish mumkin. Bu funksiyalar quyidagi prototiplarga ega: Int fprint(oqimga ko‘rsatkich, formatlashqatori, o‘zgaruvchilar ro‘yhati);

Int fCin>>(oqimga ko‘rsatkich, formatlash-qatori, o‘zgaruvchilar ro‘yhati);

Misol tariqasida int .dat faylini yaratuvchi va bu faylga 1 dan 100 gacha bo‘lgan sonlarning simvolli tasvirini yozib qo‘yuvchi dasturni ko‘rib chiqamiz:

```
#include <iostream.h>
int main()
{
FILE *fp;
Int n;
If((fp=fopen(—int.dat‖,‖ts‖))==NULL)
{perror (—int.dat‖); return 1;
}
for(n=1; n<11;n++) fCout<<(fp,—%d ‖;n);
} fclose(fp); return 0;
}
```

Faylga ihtiyoriy murojat qilish

Hozirgi ko‘rib chiqilgan funksiyalar faylga ketma-ket yozish yoki ketma-ket uqishga imkon beradi holos. Fayldan uqib faylga yozishlar doim joriy pozitsiyasida bo‘ladi. Boshlang‘ich pozitsiya fayl ochilganda aniqlanadi. Faylni "r" va"w" rejimida ochilganda joriy pozitsiya ko‘rsatgichi faylning birligi baytini ko‘rsatadi, "a" rejimida ochilganda, oshish baytini ko‘rsatadi. Har bir kiritish-chiqarish amali bajarilganda, ko‘rsatgich o‘qilgan baytlar soniga qarab yangi pozitsiyaga ko‘chadi. Faylning ihtiyoriy baytiga murojat qilish uchun fseek() funksiyasidan foydalanish lozimdir.Bu funksiya quyidagi prototipga ega.

Int fseek (faylga ko‘rsatgich, oraliq,hisobat boshi) farq log tipidagi o‘zgaruvchi yoki ifoda bilan beriladi. Hisobot boshi oldin quyidagi konstantalardan biri bilan aniqlanadi.

Seek_ Set (qiymati 0)-fayl boshi;
Seek_cur (qiymati 1)-uqilayotgan pazitsiya;
Seek_ end (qiymati 2)-fayl ochish;

Fseek () funksiyasi 0 qaytaradi agar faylda ko‘chish bajarilgan bo‘lsa, aksincha noldan farqli songa teng bo‘ladi.

Ihtiyoriy pozitsiyadan fayl boshiga o‘tish: Fseek (fp,ol,seek-set)

Ihtiyoriy pozitsiyadan fayl boshiga o‘tish: Fseek (fp,ol,seek-end)

Joriy pozitsiyadan bir bayt oldinga yoki orqaga ko‘chish uchun fseek (jp,-1L,seek-cur).

Fseek funksiyasidan tashqari C ++ tili bibliotekasida pozitsiyaga ko‘rsatkichlar bilan bog‘lik quyidagi funksiyalar mavjud.

Long ftell (FIHE*)-faylda ko‘rsatkichning joriy pozitsiyasini aniqlash. Void rewind (FILE*)-joriy pozitsiya ko‘rsatkichini fayl boshiga keltirish.

Quyi darajadagi kiritish va chiqarish.

Quyi darajadagi kiritish va chiqarish funtsiyalari operatsion tizim imkoniyatlaridan to‘g‘ridan to‘g‘ri foydalanishga imkon beradi. Bu holda buferlash va formatlash bajarilmaydi.

Faylni quyi darajadagi ochishda fayl bilan fayl (oqim) ko‘rsatkichi emas, deskriptor bog‘lanadi. Fayl deskriptori fayl ochilganligi to‘grisidagi ma'lumotni operatsion tizim ichki jadvallariga joylashtiruvini belgilovchi butun sondir. Quyi darajadagi funksiyalar dasturga iostream.h bibliotekasini qo‘shishni talab qilmaydi. Lekin bu biblioteka fayllar bilan ishlashda foydali bo‘lgan ba'zi konstantalar (misol uchun fayl yakuni belgisi EOF) tarifini uz ichiga oladi. Bu konstatalarda foydalanganda iostream.h dasturga qo‘shilishi zarurdir.

Fayllarni ochish va yopish.

Fayllarni quyi darajadada ochish uchun open () funksiyasidan foydalaniladi:

int fd= open (fayl nomi, bayroqlar, murojat.) fd – fayl deskriptori, fayl nomi – simvollar massiviga ko‘rsatkichdir.

2- parametr bayroqlar fayl ochish rejimini belgilovchi ifodadir. Bu ifoda fcntl.h sarlavhali faylda saqlanuvchi konstantalardan biri yoki shu kostantalardan razryadli ‗|‘ amali yordamida hosil qilingan bo‘lishi mumkin.

Kostantalar ro‘yhati:

O_APPEND Faylni ohiriga yozuv qo‘shish uchun ochish;

O_BINARY Faylni bitli (ikkili)binar rejimda ochish

O_CREAT Yangi fayl yaratish va ochish

O_EXCL Agar O_CREAT bilan birga ko‘rsatilgan bo‘lsa va yaratilmoqchi bo‘lgan fayl mavjud bo‘lsa faylni ochish funksiyasi hatolik bilan tugaydi. Mavjud faylni o‘chib ketmaslikdan saqlaydi.

O_RDONLY Faylni faqat o‘qish uchun ochish

O_RDWR Faylni o‘qish va yozish uchun ochish

O_TEXT Faylni tekstli rejimda ochish

O_TRUNK Mavjud faylni ochish va bor ma'lumotni o‘chirish

Fayl ochilish rejimi albatta ko‘rsatilgan bo‘lishi shart. 3- parametr murojat huquqlari faqat faylni O_CREAT ochish rejimida ya'ni yangi fayl yaratishda foydalaniladi. MS DOS va MS WINDOWS operatsion tizimlarida murojat huquqlari parametrlarini berish uchun quyidagi konstantalardan foydalaniladi.

S_IWRITE Faylga yozishga ruhsat berish
S_IREAD Fayldan uqishga ruhsat berish
S_ IREAD\ S_WRITE Uqish va yozishga ruhsat berish

Ko‘rsatilgan konstantalar sys katalogida joylashgan stat.h sarvlahali faylda saqlanadi. Bu faylni qo‘shish # include <sys\stade.h> direktivasi orqali amalga oshiriladi. Agar murojaat huquqi parametri ko‘rsatilmagan bo‘lsa faqat fayldan o‘qishga ruhsat beriladi. UNIX operatsion tizimida murojaat huquqlari 3 hil foydalanuvchilar uchun ko‘rsatiladi:

Fayl egasi;
Foydalanuvchilar guruhi
a'zosi. Boshqa
foydalanuvchilar
Foydalanuvchilar huquqlari quyidagi simvollar orqali
ko‘rsatiladi: R- fayldan uqish ruhsat berilgan.
W- faylga yozish ruhsat berilgan.
X- fayllarni bajarish ruhsat berilgan.

Agar biror murojaat huquqi berilmagan bo‘lsa urniga `_` belgisi quyiladi. Agar fayl egasiga hamma huquqlar, foydalanuvchi guruhi a'zolariga o‘qish va bajarish, boshqa foydalanuvchilarga faqat bajarish huquqi berilgan bo‘lsa, murojaat qatorini quyidagicha yozish mumkin rwxr-x—x. Har bir `_` simvol urniga 0 rakami, aks holda 1 raqami quyilib hosil bo‘lgan sondagi o‘ng tomondan boshlab har bir uch raqamini sakkizlik son sifatida yozilsa, murojaat huquqini belgilovchi sakkizlik butun son hosil bo‘ladi. Yuqorida hosil qilingan rwxr-x—x qatori ikkilik 111101001 nihoyat sakkizlik 0751 son shaklida yozilib open ( ) funksiyasida murojaat huquqi parametri sifatida ko‘rsatiladi. Faylni ochishga misollar:

1. faylni o‘qish uchun ochish:
fd=open (— t.txt — , O_RONLY)

2. faylni o‘qish va yozish uchun ochish:
fd = open(— t.txt — , O_RDWR)

3. faylni yangi ma'lumotlar yozish uchun ochish:
fd = open(— new.txt — ,O_WRONLY_ |O-Creat| O_TRUNK, 0600)

Sakkizlik konstanta 0600 shaklida berilgan murojaat huquqi parametrining simvolli ko‘nishi rw -------- bulib, fayl egasiga o‘qish va yozish huquqi , qolgan foydalanuvchilarga hech qanday huquq berilmaganligini bildiradi . Faylni ochishda kelib chiqadigan hato turini aniqlash uchun errno.h

sarlavhali faylda saqlanuvchi errno o‘zgaruvchisi hizmat qiladi. Agar bu o‘zgaruvchi qiymati shu sarlavhali faylda saqlanuvchi EEXIST konstantasiga teng bo‘lsa ochilayotgan fayl mavjudligini bildiradi.

Sopen ( ) funksiyasi bitta faylga bir necha dasturlardan murojaat qilish imkonini beradi.

Albatta dasturlar faylga faqat o‘qish rejimida murojaat qilishi mumkin. Faylni ochish uchun yana Creat ( ) funksiyasi mavjud bulib quyidagi Open ( ) funksiyasini chaqirishga mos keladi.

Open (fayl nomi, O_creat |O_TRUNK| O_WRONLY); bu funksiya yangi fayl yaratadi va yozish uchun ochadi. Quyi darajada fayllarni yopish uchun close ( ) funksiyasidan foydalanish lozim. Bu funksiya ko‘rinishi quyidagichadir:

Int close (fayl deskriptori). Funksiya muvofaqiyatli bajarilganda 0 qaytaradi. Hato bo‘lganda – 1.

Ma'lumotlarni o‘qish va yozish

Quyi darajada ma'lumotlarni kiritish va chiqarish read ( ) va write ( ) funksiyalari orqali amalga oshiriladi. Bu funksiyalar prototiplari quyidagi ko‘rinishga ega:

int read (int fd, char * buffer; unrigned
int count) int write (int fd, char *
buffer; unsiqned int count)

Ikkala funksiya butun o‘qilgan yoki yozilgan baytlar sonini qaytaradi. Read funksiyasi fd deskriptori bilan ochilgan fayldan count parametrida ko‘rsatilgan miqdordagi baytlarni o‘qib, buffer ko‘rsatkichi orqali ko‘rsatilgan bufferga yozadi. Fayl ohiriga etganda read () funktsyasi 0 qiymat qaytaradi. Fayldan o‘qishda hatolik kelib chiqsa -1 qiymat qaytaradi. O‘qish fayldagi joriy pozitsiyadan boshlanadi. Agar fayl matnli rejimda ochilsa CR va LF simvollari `\n` simvoliga o‘zgartiriladi.

Write ( ) funksiyasi fd deskriptori bilan ochilgan faylga buffer ko‘rsatkichi orqali ko‘rsatilgan bufferdan count parametri orqali ko‘rsatilgan miqdordagi baytlarni yozib qo‘yadi. Yozuv joriy pozitsiyadan boshlanadi. Agar fayl matnli rejimda ochilgan bo‘lsa`\n` simvolini CR va LF simvollar sifatida yoziladi. Agar yozishda hatolik kelib chiqsa, write ( ) funksiyasi -1 qymat qaytaradi. Errno global o‘zgaruvchisi bo‘lsa Errno.h sarlavhali faylda ko‘rsatilgan quyidagi konstantalar biriga teng bo‘ladi.

EACCES – fayl yozuvdan himoyalangan
ENOSPC – tashki kurilmada bush joy kolmagan
EBADF – notugri fayl deskriptori

Bu funksiyalar io.h sarlavhali faylda joylashgandir. Quyida bir fayldan ikkinchisiga nusha olish dasturini ko‘rib chiqamiz:

dastur
#include <iostream.h>
#include <fcnfl.h>
#include <io.h>

```
int main(int argc,  char
*argv[ ] ) {
 int  fdin , fdout;  /*Deskriptorih
faylov*/  int n; /* Kolichestvo
prochitannihh  baytov*/
 char
buff[BUFSI
Z];  if  (argc
!=3)  {
  Cout<< (—Format vihzova programmih: ‖);
Cout<<(—\n %s fayl_istochnik  fayl_priemnik‖,
            argv[0]);
return 1;
  }
if  ((fdin =open(argv[1],O_RDONLY)) ==-1)
{
   perror
(argv[1]);
return 1; }
if ((fdout=open(argv[2],
       O_WRONLY|O_CREAT|O_TRUNC))== -1)
{
perror
(argv[2]);
return 1;
}
/* faylih otkrihtih – mojno
kopirovat   */ while  ((n=read(fdin,
buff, BUFSIZ))>0)
   write (fdout, buff,
n ); return 0;
} /* konets dastur */
```

BUFSIZ konstantasi iostream.h sarlavhali faylda aniqlangan bo‘lib MS DOS uchun 512 bayt ga teng

Faylga ihtiyoriy murojaat.

Quyida darajada fayllarni ihtiyoriy tartibda uqish mumkin. Buning uchun lseek ( ) funksiyasidan foydalanish lozim. Bu funksiya prototipi quyidagi ko‘rinishga ega:

AMALIY MASHG'ULOTLAR

1-2-Amaliy mashg'ulot.
Mavzu: C++ da ma'lumotlarning asosiy turlari bilan amallar bajarish. C++ da chiziqli dasturlash

C++da oddiy matnni ekranga chiqaruvchi dasturni ko'rib chiqamiz

```
1  // Muallif: Toxirov Feruz
2  // Sana:    27.08.2019
3  // Maqsad:  Matnni ekranga chiqaruvchi dastur
5 #include <iostream> // ekranga ma'lumot chiqarish uchun   6
7  int main()
8  {
9  std::cout << "Assalomu alaykum  bo'lajak programmist!\n";  10
11 return 0;
12 }
```

Har bir satrni o'rganib chiqamiz:

1, 2, 3 - satrlar izoh hisoblanadi. Malakali programmistlar har qanday dastur muallif, dasturning tuzilish sanasi va maqsadini ifodalovchi izoh bilan boshlanishini maslahat berishadi.

4, 6, 10 - satrlar bo'sh satrlar hisoblanadi. Bosh satrlar dastur qismlarini bir - biridan ajratib quyish uchun ishlatiladi. Dastur qismlarining bir - biridan ajralib turishi, dastur o'qilishini osonlashtiradi.

5 - satrda, klaviaturadan ma'lumotlarni kiritish va ekranga chiqarish uchun <iostream> sarlavha fayli dasturga qo'shilyapti. Bu satr klaviatura orqali ma'lumot kirituvchi va ekranga nimadir chiqaruvchi har qanday dasturda bo'lishi shart. Aks xolda xato sodir bo'ladi. Agar sizning kompilyatoringiz eski bo'lsa, unda <iostream.h> yozishingiz lozim bo'ladi.

"// ekranga ma'lumot chiqarish uchun" yozuvi bir satrli izoh hisoblanadi.

7 - satrda butun toifadagi qiymat qaytaruvchi main funksiyasi berilgan. int xizmatchi so'zi butun toifadagi ma'lumotlarni e'lon qilishi uchun ishlatilad i.

8 - satrdagi ochuvchi figirali { funksiya tanasining boshlanganini bildiradi.

12 - satrdagi yopuvchi figirali } funksiya tanasining tugaganini bildiradi.

9 - satrda std::cout << orqali ma'lumotlar ekranga chiqariladi. Qo'shtirnoq ("_") orasida yozilgan ma'lumotlar satr deyiladi. Qo'shtirnoq orasida nima yozilsa, hech qanday o'zgarishsiz ekranga chiqariladi.

9 - satr oxiridagi nuqtali vergul (;) std::cout operatori tugallanganligini bildiradi. ; operatorlarni bir - biridan ajratish uchun xizmat qiladi. Ya'ni operator tugallanganligini bildiradi. 5 - satrdagi kabi preprotsessor amalidan keyin ; quyilmaydi.

11 - satrdagi return xizmatchi so'zi orqali funksiya 0 qiymat qaytaradi va dastur muvoffaqiyatli yakunlanadi.

O'zgaruvchilarni e'lon qilish. Dasturda ishlatilgan barcha o'zgaruvchilarni qaysi toifaga tegishli ekanligini e'lon qilish kerak. Ma'lulotlarni e'lon qilishning umumiy ko'rinishi quyidagicha:
toifa_nomi o'zgaruvchi;
Agar bir nechta o'zgaruvchi bir toifaga mansub bo'lsa, ularni vergul bilan ajratib berish mumkin. Butun sonlarni ifodalash uchun int va haqiqiy sonlarni ifodalash uchun float xizmatchi so'zlaridan foydalaniladi. Bu ma'ruzada shu 2 tasini bilish bizga kifoya qiladi. Keyingi mavzuda butun va haqiqiy sonlar haqida batafsil gaplashamiz.

int x,y; // *butun toifadagi o'zgaruvchilarni e'lon qilish* float a,b,c; // *haqiaiy toifadagi o'zgaruvchilar e'lon qilish*

Kiritish va chiqarish operatorlari. Dasturda klaviatura orqali ma'lumot kiritish va ekranga chiqarish uchun preprotsessor direktivasini, ya'ni #include <iostream> ni dasturga qo'shish shart. Ma'lumotlarni kiritish std::cin >>, ma'lumotlarni chiqarish std::cout << operatori orqali amalga oshiriladi. std::cin >> a;
Bu operator bajarilganda ekranda kursor paydo bo'ladi. Kerakli ma'lumot klaviatura orqali kiritilgandan so'ng Enter tugmasi bosiladi. cout orqali ekranga ixtiyoriy ma'lumotni chiqarish mumkin. Satrli ma'lumotlarni ekranga chiqarish uchun, ularni qo'shtirnoq orasida yozish kerak.

Quyida a va b sonlarining yig'indisini chiqaruvchi dastur berilgan:

```
#include <iostream>
// standart nomlar fazosidan foydalanishni e'lon qilish  using namespace std;
int main()  {  int a, b, c; cout << "a="; cin >> a;      cout << "b="; cin >> b;      c = a + b;      cout << c << endl;
   return 0;
}
```

Ba'zi matematik funksiyalar:
Matemetik funksiyalardan dasturda foydalanish uchun math.h faylini progarmmaga qo'shish kerak. **#include <math.h>**

Funksiyaning C++ da ifodalanishi	Funksiyaning matematik ifodalanishi
abs(x) - butun sonlar uchun fabs(x) - haqiqiy sonlar uchun	/x/
pow(x, y)	x^y
sqrt(X)	$\sqrt{x}$

Matematik funksiyalardan foydalanish

```
#include <iostream>  #include <math.h>  using namespace std;
int main()
{      float a;
   cout << "a="; cin >> a;
   a = sqrt(a);      cout << a << endl;
   return 0;
}
```

Dasturchilar doim dastur ishlashi jarayonida xotiradan kamroq joy talab qilishligi haqida bosh qotirishadi. Bu muammolar dasturdagi o'zgaruvchilar sonini kamaytirish, yoki o'zgaruvchilar saqlanadigan yacheyka hajmini kamaytirish orqali erishiladi. Biz butun va haqiqiy sonlarni e'lon qilishni bilamiz. Bulardan tashqari C++ da butun va haqiqiy sonlarni e'lon qilish uchun bir nechta toifalar mavjud. Ular bir - biridan kompyuter xotirasida qancha hajm egallashi va qabul qiluvchi qiymatlar oralig'i bilan farq qiladi.

Butun sonlar

Toifa ko'rinishi	Qabul qiladigan qiymatlar oralig'i	Kompyuter xotirasida egallagan hajmi
unsigned short int	0..65535	2 bayt
short int	-32768..32767	2 bayt
unsigned long int	0..42949667295	4 bayt
long int	-2147483648..2147483647	4 bayt
int (16 razryadli)	-32768..32767	2 bayt
int (32 razryadli)	-2147483648..2147483647	4 bayt
unsigned int (16 razryadli)	0..65535	2 bayt
unsigned int (32 razryadli)	0..42949667295	4 bayt

Haqiqiy sonlar

Toifa ko'rinishi	Qabul qiladigan qiymatlar oralig'i	Kompyuter xotirasida egallagan hajmi
float	1.2E-38..3.4E38	4 bayt
double	2.2E-308..1.8E308	8 bayt
long double (32 razryadli)	3.4e-4932..-3.4e4932	10 bayt

Boshqa toifalar

Toifa ko'rinishi	Qabul qiladigan qiymatlar oralig'i	Kompyuter xotirasida egallagan hajmi
bool	**true** yoki **false**	1 bayt
char	0..255	1 bayt
void	2 yoki 4	

Har xil toifadagi o'zgaruvchilar kompyuter xotirasida turli xajmdagi baytlarni egallaydi. Xattoki bir toifadagi o'zgaruvchilar ham qaysi kompyuterda va qaysi operatsion sistemada ishlashiga qarab turli o'lchamdagi xotirani egallashi mumkin. C++ da ixtiyoriy toifadagi o'zgaruvchilarning o'lchamini sizeof funksiyasi orqali aniqlash mumkin. Bu funksiyani o'zgarmasga, biror toifaga va o'zaruvchiga qo'llash mumkin.

Toifalarni kompyuter xotirasida egallagam xajmini aniqlash

```
#include <iostream>  using namespace std;
int main()
```

```
{
   cout << "char  = " << sizeof(char)  << endl;      cout << "bool  = " << sizeof(bool)
<< endl;       cout << "int   = " << sizeof(int)   << endl;       cout << "float = " <<
sizeof(float) << endl;      cout << "double= " << sizeof(double)<< endl;       return 0;
}
```

Natija: ***char=1 bool=1 int=4 float=4 double=8***

Matemetik funksiyalardan dasturda foydalanish uchun math.h sarlavha faylini progarmmaga qo'shish kerak. #include <math.h>

Funksiyaning C++ da ifodalanishi	Funksiyaning matematik ifodalanishi
1. abs(x) - butun sonlar uchun 2. fabs(x) - haqiqiy sonlar uchun 3. labs(x) - uzun butun son uchun	/x/
pow(x, y)	x^y
pow10(x)	10^x
sqrt(X)	$\sqrt{X}$
ceil(x)	haqiqiy toifadagi x o'zgaruvchisi qiymatini unga eng yaqin katta butun songa aylantiradi.
floor(x)	haqiqiy toifadagi x o'zgaruvchisi qiymatini unga eng yaqin kichik butun songa aylantiradi
cos(x)	x burchak kosinusini aniqlash. x radian o'lchovida.
sin(x)	x burchak sinusini aniqlash. x radian o'lchovida.
exp(x)	e^x
log(x)	x sonining natural logarifmini qaytaradi.
log10(x)	x sonining 10 asosli logarifmini qaytaradi.

Eslatma: Barcha trigonometrik funksiyalar radian o`lchovida beriladi.

1 - Misol: n va m natural sonlari berilgan. n sonini m soniga bo'lib, qoldiqni aniqlovchi dastur tuzilsin

```
#include <iostream>
int main()
{
   int n, m, qoldiq;
  cout << "n="; cin >> n;
cout << "m="; cin >> m;        // % qoldiqni olishni bildiradi        qoldiq = n % m;
cout << "Qoldiq=" << qoldiq << endl;
 return 0;  }
```

1-Misol: n va m natural sonlari berilgan. n sonini m soniga bo'lib, butun qismini aniqlovchi dastur tuzilsin #include <iostream>

```
int main()
{
```

```
    int n, m, b;

    cout << "n="; cin >> n;
     cout << "m="; cin >> m;
    b = n / m;
    cout << "Butun qismi=" << b << endl;
    return 0;
}
```

2-misol. a sonini b soniga bo`lib 2 xona aniqlikda chiqarish.

```
#include <iostream>
#include <iomanip>
// <iomanip> sarlavha faylini qo'shamiz   using namespace std;
int main()
{
    float a, b;
    cout << "a sonini b soniga bo`lib 2 xona aniqlikda chiqarish"<<endl;        cout <<
"a="; cin >> a;
    cout << "b="; cin >> b;
    a = a / b;       cout <<  a << endl;
    cout << setprecision(2) << fixed << a << endl;       return 0;
}
```

Natija:

```
a sonini b soniga bo`lib 2 xona aniqlikda chiqarish
a=10
b=3
3.33333
3.33

Process returned 0 (0x0)   execution time : 6.723 s
Press any key to continue.
```

3-misol. Bir toifadan boshqasiga o'tish c++ da bir toifadan boshqasiga o'tishning oshkor va oshkormas usullari mavjud. Oshkor ravishda toifaga keltirish uchun qavs ichida boshqa toifa nomi yoziladi.

```
#include <iostream>  using namespace std;
int main()
{       float haqiqiy = 5.57; int oshkor, oshkormas;
     // oshkormas ravishda butun toifaga o'tish       oshkormas = haqiqiy;        oshkor =
(int) haqiqiy; // oshkor holda butun toifaga o'tish        cout << "haqiqiy = " << haqiqiy
<< endl;       cout << "oshkor = "  << oshkor  << endl;
    cout << "oshkormas = " << oshkormas << endl;       return 0;
}
```

4-misol. Butun sonni bo'lish

```
 #include <iostream>  using namespace std;
int main()
{       int bir  = 1;
    int ikki = 2;
```

```
   cout << bir / ikki << endl;
   cout << ((float)bir) / ((float)ikki) << endl;      return 0;
}
```

Natija:

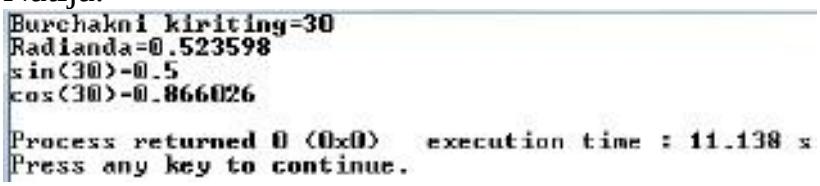
5-misol. Trigonometrik funksiyalar bilan ishlash

```
#include <iostream>  #include <math.h>  using namespace std;
int main()
{      float const pi = 3.14159;
   float burchak, burchak_radian;

   cout << "Burchakni kiriting="; cin >> burchak;      burchak_radian = burchak * pi / 180;

   cout << "Radianda=" << burchak_radian << endl;      cout << "sin(" << burchak << ")=" << sin(burchak_radian) << endl;         cout << "cos(" << burchak << ")=" << cos(burchak_radian) << endl;      return 0;
}
```

Natija:



3-Amaliy mashg’ulot.

Mavzu: C++ tilida shartli va shartsiz o'tish operatorlari. Tanlash operatori.

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

If (ifoda)
1- operator
Else 2- operator eki If (ifoda)
1-operator

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

if(n>0) if(a>b) Z=a; else Z=b; Agar else qismni yuqori if ga mos quyish lozim bo‘lsa, figurali qavslar ishlatish lozim.

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

Misol tariqasida uchta berilgan sonning eng kattasini aniqlash dasturini ko‘ramiz:

```
#include <iostream.h> void( )
{  float a,b,c,max);
   Cout <<—\n a=‖; Cin>>a;
   Cout <<—\n b=‖; Cin>>b;     Cout <<—\n c=‖; Cin>>c;   if (a>b)
   if (a>c) max=a else max=c;   else
    if b>c then max=b else max=c;    Cout <<—\n‖ <<max;
}
```

Keyingi misolda kiritilgan ball va maksimal ball asosida baho aniqlanadi:

```
#include <iostream.h> void main( )
{  float ball,max_ball,baho);
   Cout<< —\n ball=‖; Cin>>(—%f‖,&ball);
   Cout<<—\n max_ball=‖; Cin>>max_ball;
   d=ball/max_ball;   if (d>0.85) baho=5 else      if (d>75) baho=4 else
   if (d>0.55) then baho=3 else baho=2;    Cout<<—\n baho;
}
```

Kalit bo‘yicha tanlash operatori. Kalit bo‘yicha o‘tish switch operatori umumiy ko‘rinishi qo‘yidagicha

```
    Switch(<ifoda>) {
    Case <1-kiymat>:<1-operator>
            …      break;              …
     default: <operator>
            …
     case: <n-operator>;
      }
```

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

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

```
Include <iostream.h>
Int baho;
Cin>> baho;
Switch(baho)
```

{case 2:Cout <<—\n emon‖;break; case 3:Cout <<—\n urta‖;break; case 4:Cout <<—\n yahshi‖;break; case 5:Cout <<—\n a'lo‖;break;
default: Cout <<—\n baho notugri kiritilgan‖; }; }
Keyingi misolimizda kiritilgan simvol unli harf ekanligi aniqlanadi:
Include <iostream.h>
Int baho; Char c; Cin >> c; Switch(c)
{case _a‘: case _u‘: case _o‘: case _i‘:
Cout <<—\n Kiritilgan simvol unli harf‖;break; default: Cout <<—\n Kiritilgan simvol unli harf emas‖;
};
}

Kvadrat tenglama ildizlari topish dasturi

```
#include <iostream.h>
#include <math.h> #include <conio.h>        int main()
                {
         int a,b,c;      float D,x1,x2;
                  cout <<"ax^2+bx+c=0 tenglama ildizini topish dasturi!";
         cout<<"\n a - koeffitsientni kiriting: "; cin>>a;      cout<<"\n b - koeffitsientni kiriting: "; cin>>b;           cout<<"\n c - koeffitsientni kiriting: "; cin>>c;      D = b*b - 4 * a * c;
         if (D<0) { cout << "Tenglama haqiqiy ildizlarga ega emas";       getch(); return 0; }
  if (D==0) { cout << "Tenglama yagona ildizga ega: ";        x1=x2= -b / (2 * a); cout<<"\n  x= "<<x1;       getch(); return 0; }
                   else
      {cout << "Tenglama ikkita ildizga ega: ";          x1 = (- b + sqrt(D)) / (2 * a);
          x2 = (- b - sqrt(D)) / (2 * a);         cout<<"\n x1= "<<x1;
       cout<<"\n x2= "<<x2;
                    }
    getch();       return 0;
 }
```

4-Amaliy mashg’ulot.
Mavzu: C++ tilida takrorlanish operatorlari (while, do while, for).

For strukturasi sanovchi (counter) bilan bajariladigan takrorlashni bajaradi.Boshqa takrorlash bloklarida (while, do/while) takrorlash sonini control qilish uchun ham sanovchini qo'llasa bo'lardi, bu holda takrorlanish sonini o'ldindan bilsa bo'lardi, ham boshqa bir holatning vujudga kelish-kelmasligi orqali boshqarish mumkin edi. Ikkinchi holda ehtimol miqdori katta bo'ladi. Masalan qo'llanuvchi belgilangan sonni kiritmaguncha takrorlashni bajarish kerak bo'lsa biz while li

ifodalar-ni ishlatamiz. for da esa sanovchi ifodaning qiymati oshirilib (kamaytirilib) borilvuradi, va chegaraviy qiymatni olganda takrorlanish tugatiladi. for ifodasidan keyingi bitta ifoda qaytariladi. Agar bir necha ifoda takrorlanishi kerak bo'lsa, ifodalar bloki {} qavs ichiga olinadi.

```
//Ekranda o'zgaruvching qiymatini yozuvchi dastur, for ni ishlatadi.
# include <iostream.h>
int main()
{  for (int i = 0; i < 5; i++){
  cout << i << endl;
 }    return (0); }
```

Ekranda:

```
0
1
2
3 4
```

for strukturasi uch qismdan iboratdir. Ular nuqtavergul bilan bir-biridan ajratiladi. for ning ko'rinishi:

```
for( 1. qism ; 2. qism ; 3. qism ){   takror etiladigan blok
}
```

1. qism - e'lon va initsalizatsiya.
2. qism - shartni tekshirish (oz'garuvchini chegaraviy qiymat bilan solishtirish).

3.qism - o'zgaruvchining qiymatini o'zgartirish.

Qismlarning bajarilish ketma-ketligi quyidagichadir:
Boshida 1. qism bajariladi (faqat bir marta), keyin
2. qismdagi shart tekshiriladi va agar u true bo'lsa takrorlanish bloki ijro ko'radi, va eng ohirda 3. qismda o'zgaruvchilar o'zgartiriladi, keyin yana ikkinchi qismga o'tiladi. for strukturamizni while struktura bilan almashtirib ko'raylik:

```
for (int i = 0; i < 10 ; i++)
  cout << "Hello!"<< endl;
```

Ekranga 10 marta Hello! so'zi bosib chiqariladi. I o'zgaruvchisi 0 dan 9 gacha o'zgaradi.
i 10 ga teng bo'lganda esa i < 10 sharti noto'g'ri (false) bo'lib chiqadi va for strukturasi nihoyasiga yetadi. Buni while bilan yozsak:

```
int i = 0;

while ( i<10 ){   cout << "Hello!" << endl;
  i++;
}
```

Endi for ni tashkil etuvchi uchta qismninig har birini alohida ko'rib chiqsak.Birinchi qismda asosan takrorlashni boshqaradigan sanovchi (counter)o'zgaruvchi- lar e'lon qilinadi va ularga boshlangich qiymatlar beriladi (initsalizatsiya). Yuqoridagi dastur misolida buni int i = 0; deb berganmiz. Ushbu qismda bir necha o'zgaruvchilarni e'lon qilishimiz mumkin, ular vergul bilan ajratilinadi. Ayni shu kabi uchinchi qismda ham bir nechta o'zgaruvchilarning qiyma-tini o'zgartirishimiz mumkin. Undan tashqari birinchi qismda for dan oldin e'lon qilingan o'zgaruvchilarni qo'llasak bo'ladi.
Masalan:

```
int k = 10;  int l;
for (int m = 2, l = 0 ; k <= 30 ; k++, l++, ++m) { cout << k + m + l; }
```

Albatta bu ancha sun'iy misol, lekin u bizga for ifodasining naqadar moslashuvchanligi ko'rsatadi. for ning qismlari tushurib qoldirilishi mumkin. Masalan: for(;;) {} ifodasi cheksiz marta qaytariladi. Bu for dan chiqish uchun break operatorini beramiz. Yoki agar sanovchi sonni takrorlanish bloki ichida o'zgartirsak,for ning 3. qismi kerak emas. Misol:

```
for(int g = 0; g < 10; ){   cout << g;   g++;
}
```

Yana qo'shimcha misollar beraylik.

```
for (int y = 100; y >= 0; y-=5){
...
ifoda(lar);
...
}
```

Bu yerda 100 dan 0 gacha 5 lik qadam bilan tushiladi.

```
for(int d = -30; d<=30; d++){
...
ifoda(lar);
... }
```

60 marta qaytariladi.
for strukrurasi bilan dasturlarimizda yanada yaqinroq tanishamiz. Endi
1. qismda e'lon qilinadigan o'zgaruvchilarning hususiyati haqida bir og'iz aytib o'taylik.
Standartga ko'ra bu qismda e'lon qilingan o'zgaruvchilarning qo'l- lanilish sohasi faqat o'sha for strukturasi bilan chegaralanadi. Yani bitta blokda joylashgan for struk-turalari mavjud bo'lsa, ular ayni ismli o'zgaruvchilarni qo'llana ololmaydilar. Masalan quyidagi hatodir:

```
for(int j = 0; j<20 ; j++){...}
... for(int j = 1; j<10 ; j++){...} //hato!
```

j o'zgaruvchisi birinchi for da e'lon qilinib bo'lindi. Ikkinchi for da ishlatish mumkin emas. Bu masalani yechish uchun ikki hil yo'l tutish mumkin.
Birinchisi bitta blokda berilgan for larning har birida farqli o'zgaruvchilarni qo'llashdir. Ikkinchi yo'l for lar guruhidan oldin sanovchi vazifasini bajaruvchi bir

o'zgaruvchini e'lon qilishdir. Va for larda bu o'zgaruv- chiga faqat kerakli boshlangich qiymat beriladi halos.

for ning ko'rinishlaridan biri, bo'sh tanali for dir. for(int i = 0 ; i < 1000 ; i++);
Buning yordamida biz dastur ishlashini sekinlashtirishimiz mumkin.

while operatori orqali murakkab konstruktsiyalarni tuzish. while operatori shartida murakkab mantiqiy ifodalarni ham qo_llash mumkin. Bunday ifodalarni qo_llashda && (mantiqiy ko_paytirish), || (mantiqiy qo_shish) , hamda !(mantiqiy INKOR) kabi operatsiyalardan foydalaniladi. Quyida while operatori konstruktsiyasida murakkabroq shartlarni quyilishiga misol keltirilgan .
while konstruktsiyasidagi murakkab shartlar.

```
#include <iostream>  using namespace std;
int main()
{
unsigned short kichik;   unsigned long katta;
 const unsigned short  MaxKichik=65535;   cout << "Kichik sonni kiriting:";    cin >> kichik;
  cout << "Katta sonni kiriting:";    cin >> katta;
  cout << "kichik son:" << kichik << "…";       //Xar bir iteratsiyada uchta shart tekshiriladi.
  while (kichik<katta && katta>0 &&
     kichik< MaxKichik )
  {
  if(kichik%5000==0) //Xar 5000 satrdan
   //keyin nuqta chikariladi    cout<<"." ;  kichik++;  katta-=2 ;
}
cout<<"\n kichik son:"<<kichik<<" katta son:"
<<katta << endl ;  return 0 ;  }
```

Natija:
Kichik sonni kirit : 2
Katta sonni kirit : 100000
Kichik son : 2 ……
Kichik son :33335 katta son : 33334

TAHLIL
Dastur quyidagi mantiqiy o_yinni ifodalaydi. Oldin ikkita son – kichik va katta kiritiladi. Undan so_ng toki ular bir biriga teng bo_lmaguncha, ya‘ni «uchrashmaguncha» kichik son birga oshiriladi, kattasi esa ikkiga kamaytiriladi. O_yinni maqsadi qiymatlar «uchrashadigan» sonni topishdir. Qiymatlar kiritilgandan so_ng siklni davom ettirishning quyidagi uchta sharti tekshiriladi:

- kichik o_zgaruvchisi qiymati katta o_zgaruvchisi qiymatidan oshmasligi.
- katta o_zgaruvchisi qiymati manfiy va nolga teng emasligi
- kichik o_zgaruvchisi qiymati MaxKichik qiymatidan oshib ketmasligi

So_ngra kichik soni 5000 ga bo_lingandagi qoldiq hisoblanadi. Agarda kichik 5000 ga qoldiqsiz bo_linsa bu operatsiyaning bajarilishi natijasi 0 ga teng bo_ladi. Bu holatda hisoblash jarayonini vizual ifodasi sifatida ekranga nuqta chiqariladi. Keyin esa kichik qiymati bittaga oshiriladi, katta qiymati esa 2 taga kamaytiriladi. Sikl agarda tekshirish sharti tarkibidagi birorta shart bajarilmasa to_xtatiladi.

do…while konstruktsiyasi yordamida sikl tashkil etish. Ayrim hollarda while operatori yordamida sikllarni tashkil etishda uning tanasidagi amallar umuman bajarilmasligi mumkin.
Chunki siklni davom etish sharti har bir iteratsiyadan oldin tekshiriladi. Agarda boshlang_ich berilgan shart to_g_ri bo_lmasa sikl tanasining birorta operatori ham bajarilmaydi.

do...while konstruktsiyasining qo_llanilishi

```
#include <iostream>  using namespace std;
int main()
{  int counter;
cout<<"How manu hellos ?";
cin >>counter;
do  {
cout << "hello \n";  counter--;
}
while(counter>0);
cout << "Counter is :" << counter <<endl;  return 0 ;
}
```

Natija: how manu hellos ? 2
hello hello Sounter is : 0
How manu hellos ? 0
Hello
Counter is: - 1

5-Amaliy mashg'ulot.
Mavzu: C++ dasturlash tilida massivlar.

Massivlar. Bir o`lchamli massivlar
Massiv - bu bir xil toifali, chekli qiymatlarning tartiblangan to`plamidir. Massivlarga misol qilib matematika kursidan ma`lum bo`lgan vektorlar, matritsalarni ko`rsatish mumkin.
Massiv bir o`lchamli deyiladi, agar uning elementiga bir indeks orqali murojaat qilish mumkin bo`lsa.
Bir o`lchamli massivni e`lon qilish quyidagicha bo`ladi:
<toifa> <massiv_nomi> [elementlar_soni] = { boshlang'ich qiymatlar }; Quyida massivlarni e`lon qilishga bir necha misollar keltirilgan: 1) float a[5]; 2) int m[6];

3) bool b[10]; 1) a elementlari haqiqiy sonlardan iborat bo`lgan, 5 ta elementdan tashkil topgan massiv. Indekslari esa 0 dan 4 gacha bo`lgan sonlar

2) m elementlari butun sonlardan iborat bo`lgan, 6 ta elementdan tashkil topgan massiv. Indekslari esa 0 dan 5 gacha bo`lgan sonlar.
3) b elementlari mantiqiy qiymatlardan (true, false) iborat bo`lgan 10 ta elementdan tashkil topgan massiv. Indekslari esa 0 dan 9 gacha bo`lgan sonlar. Massiv elementlariga murojaat qilish oddiy o`zgaruvchilarga murojaat qilishdan biroz farq qiladi. Massiv elementiga murojaat qilish uning indeksi orqali bo`ladi.

a[1] = 10; a massivining 1 – elementi 10 qiymat o‘zlashtirsin; cin >> a[2]; a massivining 2 – elementi kirtilsin;
cout << a[3]; a massivining 3 – elementi ekranga chiqarilsin;
Massivni e'lon qilishda uning elementlariga boshlang'ich qiymat berish mumkin va buning bir nechta usuli mavjud.

1) O'lchami ko'rsatilgan massivni to'liq initsializatsiyalash.

int k[5] = { 2, 3, 7, 8, 6};
Bu yerda 5 ta elementdan iborat bo'lgan k massivi e'lon qilingan va massivning barcha elemantlariga boshlang'ich qiymat berilgan.

2) O'lchami ko'rsatilgan massivni to'liqmas initsializatsiyalash.

int k[5] = { 2, 3, 7 };
Bu yerda 5 ta elementdan iborat bo'lgan k massivi e'lon qilingan va massivning dastlabki 3 ta elemantlariga boshlang'ich qiymat berilgan.

3) O'lchami ko'rsatilmagan massivni to'liq initsializatsiyalash.

int k[] = { 2, 3, 7, 8, 6};
Shuni takidlash lozimki, agar massiv o'lchami ko'rsatilmasa, uni to'liq initsializatsiyalash shart. Bu xolda massiv o'lchami kompilyatsiya jarayonida massiv elementlari soniga qarab aniqlanadi. Bu yerda massiv o'lchami 5 ga teng.

4) O'lchami ko'rsatilgan massivning barcha elementlariga boshlang'ich qiymat 0 berish:

int k[5] = { 0 }; O'lchami ko'rsatilgan massivning barcha elementlariga boshlang'ich qiymat 0 berish

```
#include <iostream>  using namespace std;
int main()
{
  int a[10] = { 0 };
  //massivning barcha elementlariga 0 qiymat berish
   for (int i = 0; i < 10; i++)
  cout << "a[" << i << "]=" << a[i] << endl;     return 0;  }
```

Agar massiv elementlariga boshlang'ich qiymatlar berilmasa xatolik sodir bo'lishi mumkin.
Elementlari butun sonlardan iborat bo`lgan, n elementdan tashkil topgan massiv elementlarini kirituvchi va ekranga chiqaruvchi dastur tuzilsin. (n <= 10)

```
#include <iostream>  using namespace std;
int main()
{
  int a[10] = { 0 };
  int n;
```

```
 cout << "n="; cin >> n;    for (int i = 0; i < n; i++)
 {
   cout << "a[" << i << "]=";
   cin >> a[i];
 }
 for (int i = 0; i < n; i++)    cout << a[i] << " ";
 return 0;
}
```

n ta elementdan tashkil topgan massiv berilgan. Shu massiv elementlari yig'indisini chiqatuvchi dastur tuzilsin. (n <= 10)

```
#include <iostream>  using namespace std;
int main()
{
 int a[10] = { 0 }; // a massivini e'lon qilish    int n;   // massiv elamentlari soni
 int s = 0; // massiv elementlari yig'indisini hisoblash uchun     cout << "n="; cin >>
n;    for (int i = 0; i < n; i++)
 {
   cout << "a[" << i << "]=";
   cin >> a[i];      s += a[i];
 }
  cout << "Massiv elementlari yig`indisi = " << s <<  endl;    return 0;
}
```

Berilgan a[n] massivning ixtiyoriy tartib raqamda joylashgan elementini ekranga chiqarish dasturini tuzing.

```
#include<iostream.h>
#include<conio.h> int main () {    int i,a[20],n;    for(i=1;i<=5;i++){    cout<<i<<"-
element:";
 cin>>a[i]; }
cout<<"qaysi elementni chiqarishni hohlaysiz:";  cin>>n;   cout<<a[n];   getch();
}
```

Massivda musbat elemenlar soni va summasini hisoblash.

```
#include <iostream> using namespace std; int main()  int s=0,k=0; int x[]={-1,2,5,-
4,8,9}; for(int i=0; i<6; i++)
{
if (x[i]<=0) continue;
k++; s+=x[i];
};
cout<<k<<endl; cout<<s; return 0;
};
```

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

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

```
Void main()
{
Int I,j,n;
Float a,b,d,x[100];
While(1)
{
Cout<<(—\n n=‖); Cin>>(—%i‖,&n); If ( n>0 && n < = 100 ) break;
Cout<<(—\n Hato 0<n<101 bulishi kerak‖); }
Cout<<(—\n elementlar kiymatlarini kiriting:\n‖); For (i=0;i<n;i++) {
Cout<<(—x[%i]=‖,i);Cin>>(—%f‖,&x[i]);} max=x[0];min=x[0];
For (s=0,i=0;i<n;i++)
{ s++;
 If (max<x[i]) max=x[i];
 If (min>x[i]) min=x[i];
};
s/=n;
Cout<<(—\n max=%f‖,max);
Cout<<(—\n min=%f‖,min);
Cout<<(—\n urta kiymat=%f‖,s);
}
```

2. Ikki o`lchovli massivlar matematikada matritsa yoki jadval tushunchasiga mos keladi.

Jadvallarning initsializatsiya qilish qoidasi, ikki o‘lchovli massivning elementlari massivlardan iborat bo‘lgan bir o‘lchovli massiv ta'rifiga asoslangandir. Misol uchun ikki qator va uch ustundan iborat bo‘lgan haqiqiy tipga tegishli d massiv boshlang‘ich qiymatlari quyidagicha ko‘rsatilishi mumkin:

float d[2][3]={(1,-2.5,10),(-5.3,2,14)};

Bu yozuv quyidagi qiymat berish operatorlariga mosdir:

d[0][0]=1;d[0][1]=-2.5;d[0][2]=10;d[1][0]=-5.3;d[1][1]=2;d[1][2]=14;

Bu qiymatlarni bitta ro‘yhat bilan hosil qilish mumkin: float d[2][3]={1,-2.5,10,-5.3,2,14}; Initsializatsiya yordamida boshlang‘ich qiymatlar aniqlanganda massivning hamma elementlariga qiymat berish shart emas.

Misol uchun: int x[3][3]={(1,-2,3),(1,2),(-4)}.

Bu yozuv qo‘yidagi qiymat berish operatorlariga mosdir: x[0][0]=1;x[0][1]=-2;x[0][2]=3;x[1][0]=-1;x[1][1]=2;x[2][0]=-4;

Initsializatsiya yordamida boshlang‘ich qiymatlar aniqlanganda massivning birinchi indeksi chegarasi ko‘rsatilishi shart emas, lekin qolgan indekslar chegaralari ko‘rsatilishi shart. Misol uchun:

Double x[][2]={(1.1,1.5),(-1.6,2.5),(3,-4)}

Bu misolda avtomatik ravishda qatorlar soni uchga teng deb olinadi.

Quyidagi ko‘radigan misolimizda jadval kiritilib har bir qatorning maksimal elementi aniqlanadi va bu elementlar orasida eng kichigi aniqlanadi:

```
#include <iostream.h> void main()
{ double a[4,3]; double s,max=0.0,min=0.0; int i,j;
for(i=0;i<4;i++) { for(j=0;j<3;j++)
{ Cout<<(— a[%d][%d]=‖,i,j);Cin>>(—%f‖,s);a[i,j]=s; if (max<s) max=s;
};
Cout<<(—\n‖); if (max<min) min=max;  }
Cout<<(—\n min=%f‖,min);
}
```

Misol. A(mxn) matritsa berilgan. Shu matritsa elementlarini kirituvchi va ekranga jadval ko`rinishida chiqaruvchi dastur. #include <iostream> using namespace std;

```
int main()  {
 int m, n, a[10][10];   cout << "Satrlar sonini kiriting \nm="; cin >> m;
 cout << "Ustunlar sonini kiriting \nn=";   cin >> n;

 cout <<"Massiv elementlarini kiriting \n";   for(int satr = 0; satr < m ; satr++)
 for(int ustun = 0; ustun < n; ustun++)
 {
  cout << "a[" << satr << "][" << ustun << "]=";
  cin >> a[satr][ustun];
 }
 // matritsani jadval shaklida chiqarish   for(int satr = 0; satr < m; satr++)
 {
  for(int ustun = 0; ustun < n; ustun++)    cout << a[satr][ustun] << "\t";    cout<<"\n";
 }   return 0;
}
```

Matritsaning har bir qatorining maksimal elementi aniqlash dasturi

```
#include <iostream> using namespace std; int main() {  double a[4][3];  double max;
int i,j;
for(i=0;i<4;i++)
{
for(j=0;j<3;j++)
{
cout<<"a["<<i<<"]["<<j<<"]=";
cin>>a[i][j]; } cout<<'\n'; };
for(i=0;i<4;i++)
{ max=a[i][0];
for(j=0;j<3;j++)

if (max<a[i][j]) max=a[i][j];
cout<<max<<endl;
} return 0;
}
```

A(i,j) massivning dioganol elementlari yig`indisini topish dasturi.

```
#include <iostream> #include <conio.h> using namespace std;
int main()
{
  int a[10][10]={0};    int i,j,m,n,sum=0,sum_=0;      cout << "Satr va ustunlar sonini kiriting m="; cin >> m;    for(i=1;i<=m;i++)    {   for(j=1;j<=m;j++)
   {   cout<<"a["<<i<<"]["<<j<<"]=";      cin>>a[i][j]; }   cout<<'\n';   } for(i = 1; i <= m; i++){  for(j = 1; j <= m; j++)
cout << a[i][j] << "\t";  cout<<"\n";}       for(i=1;i<=m;i++)     {     for(j=1;j<=m;j++)
{    if (i==j) sum+=a[i][j];     if (i==m+1-j) sum_+=a[i][j];    }  }
cout<<"1-dioganal= "<<sum<<endl<<"2-dioganal= "<<sum_; getch(); return 0;
}
```

6-Amaliy mashg’ulot.
Mavzu: C++ tilida funksiyalar yaratish.

Foydalanuvchi Funksiyalari.

Funksiyalarni ta'riflash va ularga murojaat qilish. Funksiya ta'rifida funksiya nomi, tipi va formal parametrlar ro`yhati ko‘rsatiladi. Formal parametrlar nomlaridan tashqari tiplari ham ko‘rsatilishi shart. Formal parametrlar ro‘yhati funksiya signaturasi deb ham ataladi. Funksiya ta'rifi umumiy ko`rinishi quyidagichadir:

Funksiya tipi funksiya nomi(formal_parametrlar_ta'rifi)

Formal parametrlarga ta'rif berilganda ularning boshlang‘ich qiymatlari ham ko‘rsatilishi mumkin. Funksiya qaytaruvchi ifoda qiymati funksiya tanasida return <ifoda> ; operatori orqali ko‘rsatiladi.

Misol:

```
Float min(float, float b) { if (a<b) return a;     return b;
}
```

Funksiyaga murojaat qilish quyidagicha amalga oshiriladi:

Funksiya nomi (haqiqiy parametrlar ruyhati)

Haqiqiy parametr ifoda ham bo‘lishi mumkin. Haqiqiy parametrlar qiymati hisoblanib mos formal parametrlar o‘rnida ishlatiladi.

Misol uchun yuqoridagi funksiyaga qo‘yidagicha murojaat qilish mumkin:

Int x=5,y=6,z; z=min(x,y) eki int z=Min(5,6) eki int x=5; int z=min(x,6)

Funksiya ta'rifida formal parametrlar initsializatsiya qilinishi, ya'ni boshlang‘ich qiymatlar ko‘rsatilishi mumkin. Funksiyaga murojaat qilinganda biror haqiqiy parametr ko‘rsatilmasa, uning urniga mos formal parametr ta'rifida ko‘rsatilgan boshlang‘ich qiymat ishlatiladi.

Misol uchun:

```
Float min(float a=0.0, float b)
{ if (a<b) return a;     return b;  }
```

Bu funksiyaga yuqorida ko‘rsatilgan murojaat usullaridan tashqari quyidagicha murojaat qilish mumkin:

Int y=6,z; z=min(,y) eki int z=Min(,6);

Agar funksiya hech qanday qiymat qaytarmasa uning tipi void deb ko‘rsatiladi. Misol uchun:
Void print;
{ Cout<<(—\n Salom!);
};
Bu funksiyaga Print; shaklida murojjat qilish ekranga Salom! Yozilishiga olib keladi.
Qiymat qaytarmaydigan funksiya formal parametrlarga ega bo‘lishi mumkin: Void Pint_Baho(Int baho);
{
Switch(baho)
{case 2:Cout<<(—\n emon‖);break; case 3:Cout<<(—\n urta‖);break; case 4:Cout<<(—\n yahshi‖);break; case 5:Cout<<(—\n a'lo‖);break;
default: Cout<<(—\n baho notugri kiritilgan‖); };
Bu funksiyaga Print_Baho(5) shaklida murojaat qilish ekranga a'lo so‘zi yozilishiga olib keladi.
Agar programmada funksiya ta'rifi murojaatdan keyin berilsa, yoki funksiya boshqa faylda joylashgan bo‘lsa, murojjatdan oldin shu funksiyaning prototipi joylashgan bulishi kerak. Prototip funksiya nomi va formal parametrlar tiplaridan iborat bo‘ladi. Formal parametrlar nomlarini berish shart emas.
Misol uchun y=min(a,b)+2*max(c,d) ifodani hisoblashni kuramaz:
#Include <iostream.h> int max(int a,int b)
{if (a<b) return a;else return b};
void main()
{int a,b,c,d,y; int min(int ,int);
Cin>>(—\n %f%f%f%f‖,&a,&b,&c,&d); y=min(a,b)+2*max(c,d); Cout<<(—\n %f‖,y);
};
int min(int a,int b)
{if (a<b) return b;else return a};
Funksiyaga parametrlar uzatish. Funksiyaga parametrlar qiymat buyicha uzatiladi va quyidagi bosqichlardan iborat bo‘ladi:

1. Funksiya bajarishga tayyorlanganda formal parametrlar uchun hotiradan joy ajratiladi, ya'ni formal parametrlar funksiyalarning ichki parametrlariga aylantiriladi. Agar parametr tipi float bo‘lsa double tipidagi ob'ektlar hosil buladi, char va shortint bulsa int tipidagi ob'ektlar yaratiladi.
2. Haqiqiy parametrlar sifatida ishlatilgan ifodalar qiymatlari hisoblanadi.
3. Haqiqiy parametrlar ifodalar qiymatlari formal parametrlar uchun ajratilgan hotira qismlariga yoziladi. Bu jarayonda float tipi double tipiga, char va shortint tiplari int tipiga keltiriladi.
4. Funksiya tanasi ichki ob'ektlar – parametrlar yordamida bajariladi va qiymat chaqirilgan joyga qaytariladi.
5. Haqiqiy parametrlar qiymatlariga funksiya hech qanday ta'sir o‘tkazmaydi.
6. Funksiyadan chiqishda formal parametrlar uchun ajratilgan hotira qismlari bo‘shatiladi.

C ++ tilida chaqirilgan funksiya chaqiruvchi funksiyadagi o‘zgaruvchi qiymatini uzgartira olmaydi. U faqat o‘zining vaqtinchalik nushasini o‘zgartirishi mumkin holos.
Qiymat bo‘yicha chaqirish qulaylik tug‘diradi. Chunki funksiyalarda kamroq o‘zgaruvchilarni ishlatishga imkon beradi. Misol uchun shu hususiyatni aks ettiruvchi POWER funksiyasi variantini keltiramiz:
power(x,n) /* raise x n-th power; n > 0; version 2 */ int x,n; int p;
for (p = 1; n > 0; --n) p = p * x;
return (p);

Argument N vaqtinchalik o‘zgaruvchi sifatida ishlatiladi. Undan to qiymati 0 bo‘lmaguncha bir ayriladi. N funksiya ichida o‘zgarishi funksiyaga murojjat qilingan boshlang‘ich qiymatiga ta'sir qilmaydi.

Ko‘rsatkichlar ta‘rifi.
C va C++ tillarining asosiy hususiyatlaridan ko‘rsatkichlarning keng qo‘llanilishidir. Ko‘rsatkichlar tilda konstanta ko‘rsatkichlar va o‘zgaruvchi ko‘rsatkichlarga ajratiladi. Ko‘rsatkichlar qiymati konkret tipdagi ob‘ektlar uchun hotirada ajratilgan adreslarga tengdir. Shuning uchun ko‘rsatkichlar ta‘riflanganda ularning adreslarini ko‘rsatish shart. O‘zgaruvchi ko‘rsatkichlar qo‘yidagicha ta‘riflanadi.
<tip> * <ko‘rsatkich nomi> Misol uchun int* lp,lk .
Ko‘rsatkichlarni ta‘riflaganda initsializatsiya qilish mumkindir. Initsializatsiya quyidagi shaklda amalga oshiriladi:
<tip> * <ko‘rsatkich nomi>=<konstanta ifoda> Konstanta ifoda sifatida qo‘yidagilar kelishi mumkin. - Hotira qismining aniq ko‘rsatilgan adresi. Misol uchun: char* comp=(char*) 0xF000FFFE; Bu adresda kompyuter tipi shaklidagi ma‘lumot saqlanadi.

- Qiymatga ega ko‘rsatkich: char c1=comp;
- & simvoli yordamida aniqlangan ob‘ekt adresi. Misol uchun: char c=‘d‘; char* pc=&c;

Borland kompilyatorlarida mahsus NULL kiymat kiritilgan bulib, bu qiymatga e‘ga ko‘rsatkichlar bush ko‘rsatkichlar deyiladi. Bush ko‘rsatkichlar bilan hotirada hech qanday adres bog‘lanmagan bo‘ladi, lekin dasturda konkret obektlar adreslarini qiyjmat sifatida berish mumkin.
Char ca=‘d‘; char* pa(NULL); pa=&ca;
Ko‘rsatkichlar ustida amallar. Yuqorida keltirilgan misollarda & adres olish amalidan keng foydalanilgan. Bu amal nomga va hotirada aniq adresga ega ob‘ektlarga, misol uchun o‘zgaruvchilarga qo‘llaniladi. Bu amalni ifodalarga eki nomsiz konstantalarga qo‘llash mumkin emas. Ya‘ni &3.14 eki &(a+b) ifodalar hato hisoblanadi.
Bundan tashqari ko‘rsatkichlar bilan birga * adres buyjicha kiymat olish eki kiritish amali keng qullaniladi. Misol uchun:
Int i=5; int*pi=&I; int k=*pi; *pi=6.
Bu misolda pi kursatkich I uzgaruvchi bilan boglanadi. *pi=6 amali I uzgaruvchi qiymatini ham uzgartiradi.
Konstanta ko‘rsatkich va konstantaga ko‘rsatkichlar. Konstanta ko‘rsatkich quyidagicha ta‘riflanadi:

<tip>* const<kursatkich nomi>=<konstanta ifoda>
Misol uchun: char* const key_byte=(char*)0x0417. Bu misolda konstanta ko‘rsatkich klaviatura holatini ko‘rsatuvchi bayt bilan bog‘langandir.
Konstanta ko‘rsatkich qiymatini o‘zgartirish mumkin emas lekin * amali yordamida hotiradagi ma‘hlumot qiymatini o‘zgartirish mumkin. Misol uchun *key_byte=‘Yo‘ amali 1047(0x0417) adres qiymati bilan birga klaviatura holatini ham oz‘zgartiradi. Konstantaga ko‘rsatkich quyidagicha ta‘riflanadi:
<tip>const*<kursatkich nomi>=<konstanta ifoda>. Misol uchun const int zero=0; int const* p=&zero;
Bu ko‘rsatkichga * amalini qullash mumkin emas, lekin ko‘rsatkichning qiyjmatini o‘zgartirish mumkin. Qiymati o‘zgarmaydigan konstantaga ko‘rsatkichlar quyidagicha kiritiladi:
<tip>const* const<kursatkich nomi>=<konstanta ifoda>. Misol uchun const float pi=3.141593; float const* const pp=π

7-Amaliy mashg’ulot.
MAVZU: C++ TILIDA STRUKTURALAR VA BIRLASHMALAR.

Sinflarni eng sodda holda quyidagicha tasvirlash mumkin: Sinf-kaliti Sinf-soni {komponentalar ruyhati}
Sinf komponentalari sodda holda tiplangan ma'lumotlar va funksiyalardan iborat bo‘ladi. Figurali qavslarga olingan komponentalar ro‘yhati sinf tanasi deb ataladi. Sinfga tegishli funksiyalar komponenta-funksiyalar yoki sinf funksiyalari deb ataladi. Sinf kaliti sifatida Struct hizmatchi so‘zi ishlatilishi mumkin. Masalan quyidagi konstruktsiya kompleks son sinfini kiritadi.

```
Struct complex 1 { double real;   double imag;
  void define (double re=0.0, double im=0.0)
{ real=re; imag=im;}   void display (void)
{cout<=‖real=‖<<real;
 cout<=‖imag=‖<<imag;
}
};
```

Strukturadan bu sinfning farqi shuki komponenta ma'lumotlardan (real, imag) tashqari ikkita komponenta funksiya (define() va display ()) kiritilgan. Bu kiritilgan sinf o‘zgaruvchilar tipi deb qaralishi mumkin. Bu tiplar yordamida konkret ob'ektlarni quyidagicha tasvirlash mumkin:
Misol uchun:

```
Complex x,y;
Complex dim[8];
Complex *p=1x;
```

Sinfga tegishli ob'ektlar quyidagicha tasvirlanadi;
  Sinf-nomi . ob'ekt-nomi
Dasturda ob'ekt komponentasiga quyidagicha murojaat qilish mumkin:
 Sinf-nomi.ob'ekt-nomi :: komponenta-nomi yoki soddaroq holda ob'ekt-nomi. Elementnomi

Misol uchun: x!=real=1.24; x!=imag=0.0; dim[3]. Real=0.25; dim[3]. Imag=0.0;
Sinfga tegishli funksiyalarga quyidagicha murojaat qilinadi: funksiya-nomi.ob'ekt-nomi; Misol uchun:
X. define.(Bu holda real=0.9 va imag=0.0)
X. define.(Bu holda kompleks son 4.3+i*20.0)
Display funksiyasi ekranda kompleks son qiymatlarini tasvirlaydi. Sinfga tegishli ob'ektga ko‘rsatkich orqali komponentalarga quyidagicha murojat qilinadi:
Ob'ektga-ko‘rsatkich>element-nomi
Yuqorida ko‘rsatilgan P ko‘rsatkich orqali H ob'ekt elementlariga quyidagicha qiymat berish mumkin:
P>real=2.3
P>imag=6.1
Huddi shu shaklda sinfga tegishli funksiyalarga murojat qilinadi:
P>display;
P>define(2.3, 5.4); Kompanenta o‘zgaruvchilar va kompanenta funksiyalar.
Sinf kompanenta o‘zgaruvchilari sifatida o‘zgaruvchilar , massivlar, ko‘rsatkichlar ishlatilishi mumkin . Elementlar ta'riflanganda initsializatsiya qilish mumkin emas . Buning sababi shuki sinf uchun hotiradan joy ajratilmaydi. Kompanenta elementlariga kompanenta funksiyalar orqali murojat qilinganda faqat nomlari ishlatiladi. Sinfdan tashqarida sinf elementlariga emas ob'ekt elementlariga murojaat qilish mumkin. Bu murojaat ikki hil bo‘lishi mumkindir.
Ob'ekt- nomi . Element - nomi.
Ob'ktga – korsatgich – element nomi.
Sinf elementlari sinfga tegishli funksiyalarida ishlatilishidan oldin ta'riflangan bo‘lishi shart emas. Huddi shunday bir funksiyadan hali ta'rifi berilmagan ikkinchi funksiyaga murojaat qilish mumkin. Komponentalarga murojaat huquqlari. Komponentalarga murojaat huquqi murojaat spetsifikatorlari yordamida boshqariladi. Bu spetsifikatorlar :
Protected – himoyalangan;
Private – hususiy;
Public – umumiy;
Himoyalangan kompanentalardan sinflar ierarhiyasi qurilganda foydalaniladi. Oddiy holda Protected spetsifikatori Private spetsifikatoriga ekvivalentdir. Umumiy ya'ni Public tipidagi komponentalarga dasturning ihtiyoriy joyida murojaat qilinishi mumkin. Hususiy ya'ni Private tipidagi komponentalarga sinf tashqarisidan murojaat qilish mumkin emas. Agar sinflar
Struct hizmatchi so‘zi bilan kiritilgan bo‘lsa, uning hamma komponentalari umumiy Public bo‘ladi, lekin bu huquqni murojaat spetsifikatorlari yordamida o‘zgartirish mumkin. Agar sinf Class hizmatchi so‘zi orqali ta'riflangan bo‘lsa, uning hamma komponentalari hususiy bo‘ladi. Lekin bu huquqni murojaat spetsifikatorlari yordamida uzgartirish mumkindir. Bu spetsifikator yordamida Sinflar umumiy holda quyidagicha ta'riflanadi:
class class_name
{
int data_member; // Ma'lumot-element void show_member(int); // Funksiya-element };

Sinf ta'riflangandan so‘ng, shu sinf tipidagi o‘zgaruvchilarni(ob'ektlarni) quyidagicha ta'riflash mumkin:
class_name object_one, object_two, object_three; Quyidagi misolda employee, sinfi kiritilgandir: class employee

```
{    public:
  char name[64] ;   long employee_id;
  float salary;
  void show_employee(void)
  {
   cout << "Imya: " << name << endl;
   cout << "Nomer slujathego: " << employee_id << endl;     cout << "Oklad: " << salary << endl;
  };
};
```

Bu sinf uch o‘zgaruvchi va bitta funksiya-elementga ega. Quyidagi EMPCLASS.CPP dastur ikki employee ob'ektini yaratadi. Nuqta operatordan foydalanib ma'lumot elementlarga qiymat beriladi so‘ngra show_employee elementidapn foydalanib hizmatchi haqidagi ma'lumot ekranga chiqariladi:

```
#include <iostream.h>  #include <string.h>
class employee
{  public:    char name [64];   long employee_id;
  float salary;
  void show_employee(void)
  {
   cout << "Imya: " << name << endl;
   cout << "Nomer slujathego: " << employee_id << endl;     cout << "Oklad: " << salary << endl;
  };
};
void main(void)
{
  employee worker, boss;   strcpy(worker.name, "John Doe");   worker.employee_id = 12345;   worker.salary = 25000;   strcpy(boss.name, "Happy Jamsa"); boss.employee_id = 101;   boss.salary = 101101.00;   worker.show_employee(); boss.show_employee();
}
```

Sinf kompleks ob'ektlari uchun umumiy bo‘lgan elementlar statik elementlar deb ataladi. Yangi ob'ektlar yaratilganda statik elementlarga murojat qilish uchun oldin inizializatsiya qilinishi lozim. Inizializatsiya quyidagicha amalga oshiriladi:
Sinf-nomi:: kompleks-nomi initsializator
Misol uchun skladdagi tovarni kompleks tasvirlovchi sinfni kurib chiqamiz. Bu sinf komponentalari quyidagilardan iborat:

- Tovar nomi
- Olish narhi

- Kushimcha narh foiz ko‘rinishida
- Tovar haqida ma'lumotlar kiritish funksiyasi
- Tovar haqida ma'lumotlar va Tovar narhini chiqaruvchi funksiya;

Sinf ta'rifi :

Goods. Cpp

#include <iostream.h>

Struct goods { char name [40]; float price; Static int percent;

Void input()

{cout <<‖ Tovar nomi:‖; cin>>name; cout<<‖ Tovar narhi:‖;cin>>price; }

Har bir yangi ob'ektning kompanentalari faqat shu ob'ektga tegishli bo‘ladi . Sinf kompanentasi yagona bo‘lib va hamma yaratilgan ob'ektlar uchun umumiy bo‘lishi uchun uni statik element sifatida ta'riflash ya'ni Static atributi orqali ta'riflash lozimdir . S sharning statik kompanentalarini inizializatsiya qilishdan so‘ng dasturda ob'ektlarni kiritmasdan oldin ishlatish mumkin. Agar kompanenta private yoki protected sifatida ta'riflangan bo‘lsa unga kompanenti funksiya yordamida murojat qilish mumkin. Lekin kompaneta funksiyaning chaqirish uchun biror ob'ekt nomini ko‘rsatish lozim. Lekin statik kompanentaga murojat qilinayotgan birorta ob'ekt yaratilmagan bo‘lishi yoki bir nechta ob'ektlar yaratilgan bo‘lishi mumkin shuning uchun sinf statik elemntlariga ob'ekt nomini ko‘rsatmasdan murojat qilish imkoniyatiga ega. Bunday imkoniyatni statik kompanenta funksiyalar yaratadi. Statik kompanenta funksiyaga ob'ekt nomi yoki obe'ktga ko‘rsatkich orqali murojaat qilish mumkin . Shu bilan birga nomi orqali qo‘ydagicha murojaat qilish mumkin.

Sinf - nomi : Statik – funksiya –nomi

Quyidagi dasturda point sinfi uch o‘lchovli fazodagi nuqtani aniqlaydi va shu bilan birga o‘z ichiga shu nuqtalar sonini oladi. # include <iostream. h >

clearr point 3 {double x,y,z; static int N; public

point 3 (double xu=0.o,double yu=0.o, double zu=0.o)

{N + +; x=xn; y=yn; z=zn; } static int & count ( ) {return N; }

};

int point 3: :N=0; void main (void)

{cout < <‖\ n size of (point 3)=‖ < < size of (point 3) ; point 3 A (0: 0, 1. 0, 2. 0);

cout < < — \ nsize of (A)=‖< < size of (A) point 3 B (3.0, 4.0, 5.0)

8-Amaliy mashg’ulot.
MAVZU: C++ TILIDA MULTIMEDIA VA ANIMATSIYALAR.

Ekran bilan ishlovchi funksiyalar. Quyidagi funksiyalar matnli rejimda ekran bilan ishlashga mo’ljallangan.
void clrscr(void) – ekranni tozalash
void gotoxy(int x, int y) – kursorni ko’rsatilgan nuqtaga ko’chirish void textcolor(int c) – text rangini o’rnatish void textbackground (int c) – text foni rangini o’rnatish Bu funksiyalar conio.h modulida joylashgandir.

Grafik rejimda ekran bilan ishlash
Grafik biblioteka. Dev C++ va Borland C++ kompilyatorlarida grafik biblioteka bilan bog’lanish uchun graphic.h – sarlavxali fayl qo’llaniladi. Bu bibliotekaga kiruvchi ba‘zi grafik funksiyalar:
void initgraph(int* graphdriver, int* graphmode, char* pathtodriver)- grafik rejimga o_tkazish void closegraph(void)-grafik rejimdan matnli rejimga o_tkazish.
void putpixel(int x, int y, int color) - Ekranda color rangli(x,y) kordinatali nuqtani tasvirlaydi.
void line (int x1, int y1, int x2, int y2) - Ekranda chiziq chizadi chizadi. void rectangle (int left, int top, int right, int bottom) - Ekranda to_rtburchak chizadi. void circle (int x, int y; int radius) - Ekranda aylana chizadi.
void ellipse (int x, int y, int stangle, int endangle, int xradius, int yradius) - Ekranda ellips
chizadi.
void outtextxy (int x, int y, char* textstring) – Textni berilgan pozitsiyada chiqaradi. void outtext (char* textstring) – Textni joriy pozitsiyada chiqaradi. int getbcolor(void) - Fon rangini qaytaradi int getcolor(void) - Tasvir rangini qaytaradi.
void getimage (int left, int top, int right, int bottom, void* bitmap) - ekran oynasini
xotirada saqlash; putimage (int left, int top, void* bitmap, int op)- xotirada saqlangan tasvirni ekranga
joylash;
Misol. Animatsiyali aylanalar

```
#include<iostream.h>
#include<graphics.h> #include<cmath>
int main()
  {
    int a,b,i,k[900],l[900],j,m;
    cout<<"a="; cin>>a; cout<<"b="; cin>>b;
    cout<<"Nuqtalar soni="; cin>>m; srand(time(NULL));
for(i=1;i<=m;i++) { k[i]=rand()%(a/2)+a/4; l[i]=rand()%(b/2)+b/4; }
initwindow(a,b); for(i=1;i<=m;i++) for(j=1;j<=m;j++) {
setcolor(i%15+1);
      circle(k[i],l[i],round(sqrt(pow(k[i]-k[j],2)+pow(l[i]-l[j],2))));
    }  system("pause");  }
```

Misol. Ichma-ich joylashgan ellipslar #include <graphics.h> main()

```
{
 int gd = DETECT, gm;   int x = 320, y = 240, radius;   initgraph(&gd, &gm, "C:\\TC\\BGI");   for ( radius = 25; radius <= 125 ; radius = radius + 20)
  circle(x, y, radius);   getch();   closegraph();   return 0;
}
```

Misol. Sin, Cos, Tan, Ctg funksiyalari grafiklarini hosil qilish

```
#include <graphics.h>
#include <conio.h>
#include <math.h> #include <stdlib.h> using namespace std; int main()
{
        initwindow(800,600);    setbkcolor(WHITE);      cleardevice();
        setcolor(RED);
   line(0,300, getmaxx(), 300);
        line(400,0, 400, getmaxx());    int x,y;        float                   pi=3.1415;
        setcolor(BLACK);
   outtextxy(10,320,"Sinus");
                setcolor(YELLOW);
   outtextxy(10,340,"Kosinus");         setcolor(GREEN);
   outtextxy(10,360,"Tangens");
                setcolor(BLUE);
   outtextxy(10,380,"Kotangens");    for (float a = -360; a <=360; a=a+0.01)
        {   setlinestyle(2,2,2);            x=400+a;
                y=int(300-sin(a*pi/180)*100);                   putpixel(x,y,BLACK);
        y=int(300-cos(a*pi/180)*100);                   putpixel(x,y,YELLOW);
        y=int(300-tan(a*pi/180)*100);                   putpixel(x,y,GREEN);
        y=int(300-1/tan(a*pi/180)*100);
                        putpixel(x,y,BLUE);
                        }
        getch();        closegraph();
        return 0;
}
```

Gorizontal yo`nalishda harakatlanuvchi uchburchak tasviri.

```
#include <graphics.h>
#include <conio.h> #include <dos.h>
void Figure ( int x, int y, int color )
{   setcolor ( color );   line ( x, y, x+20, y );   line ( x, y, x+10, y-20 );   line ( x+10, y-20, x+20, y );
} int main()
{
 int d = VGA, m = VGAHI;
 int x, y, dx,key;
 initgraph ( &d, &m, "c:\\borlandc\\bgi" );

x = 0; y = 240; dx = 1; while ( 1 )
 {   if ( kbhit() )    if ( getch() == 27 ) break;   Figure ( x, y, YELLOW );   delay ( 10 );
```

```
 Figure ( x, y, BLACK );  if ( x + 20 >= 639 ) dx = - 1;   if ( x <= 0 )    dx = 1;
 x += dx;
 }   closegraph(); return 0;
}
```

Klaviaturaning Up, Down, Right va Left tugmalariga mos ravishda harakatlanuvchi aylana tasviri.

```
#include <graphics.h> #include <conio.h>
void Figure ( int x, int y, int color )
{   setcolor ( color );   circle(x,y,50);
}
int main()
{ int d = VGA, m = VGAHI;
 int x, y, key;   initgraph ( &d, &m, "" );   setbkcolor(WHITE);   cleardevice();    x = 320; y = 240;   while ( 1 )
  {
  Figure ( x, y, RED );     key = getch();     if ( key == 27 ) break;     Figure ( x, y, WHITE );     switch ( key ) {      case 75: x --; break;      case 77: x ++; break;      case 72: y --; break;
   case 80: y ++; break;
   }
  }
 closegraph(); return 0; }
```

Ichma-ich hosil bo`luvchi aylanalar tasviri.

```
#include <graphics.h> #include <conio.h> using namespace std; int main()
{
        initwindow(400,400);      setbkcolor(WHITE);       a:cleardevice();
                for (int i = 1; i <=100; i=i+5)
                {
  setcolor(RED);      circle(200,200,i*2);
  delay(50);
 }    cleardevice();
 for (int i = 100; i >=1; i=i-5)
                {
  setcolor(RED);      circle(200,200,i*2);     delay(50);   }
 goto a;
        getch();        closegraph(); return 0;
}
```

Soat tasviri

```
#include <graphics.h>
#include <iostream.h>
#include <conio.h>
#include <math.h> #include <stdlib.h> using namespace std; int main() {       float fi=0; int k=150;
```

```
        float x,y; int x1,y1;     initwindow(400,400);
        setbkcolor(WHITE);       cleardevice();                    setcolor(BLACK);
outtextxy(190,35,"12"); outtextxy(355,190,"3");
outtextxy(200,350,"6"); outtextxy(35,190,"9");
                do {
      setlinestyle(0,1,2);        fi=fi+6.28/60;          if (fi== 6.28) fi=0;        cout<<"/a";
x1=int(k*cos(fi+3.14));          y1=int(k*sin(fi+3.14));
      setcolor(10); line(200,200, x1+200, y1+200);
        delay(1000);

          setcolor(15);   line(200,200,  x1+200,  y1+200);                  setcolor(10);
circle(200,200,170);}
      while (KEY_END);


getch();
   }
```

LABORATORIYA ISHLANMALARI

1-Laboratoriya mashg'uloti.
Mavzu: C++ tilida chiziqli dasturlar tuzish.

Ishning maqsadi: Talabalarga C++ tilida chiziqli dastrular tuzishni o`rgatish.

Nazariya bo`yicha qisqacha ma'lumot

O'zgaruvchilarni e'lon qilish. Dasturda ishlatilgan barcha o'zgaruvchilarni qaysi toifaga tegishli ekanligini e'lon qilish kerak. Ma'lulotlarni e'lon qilishning umumiy ko'rinishi quyidagicha:
toifa_nomi o'zgaruvchi;
Agar bir nechta o'zgaruvchi bir toifaga mansub bo'lsa, ularni vergul bilan ajratib berish mumkin. Butun sonlarni ifodalash uchun int va haqiqiy sonlarni ifodalash uchun float xizmatchi so'zlaridan foydalaniladi. Bu ma'ruzada shu 2 tasini bilish bizga kifoya qiladi. Keyingi mavzuda butun va haqiqiy sonlar haqida batafsil gaplashamiz.

int x,y; // *butun toifadagi o'zgaruvchilarni e'lon qilish* float a,b,c; // *haqiaiy toifadagi o'zgaruvchilar e'lon qilish*

Kiritish va chiqarish operatorlari. Dasturda klaviatura orqali ma'lumot kiritish va ekranga chiqarish uchun preprotsessor direktivasini, ya'ni #include <iostream> ni dasturga qo'shish shart. Ma'lumotlarni kiritish std::cin >>, ma'lumotlarni chiqarish std::cout << operatori orqali amalga oshiriladi. std::cin >> a;
Bu operator bajarilganda ekranda kursor paydo bo'ladi. Kerakli ma'lumot klaviatura orqali kiritilgandan so'ng Enter tugmasi bosiladi. cout orqali ekranga ixtiyoriy ma'lumotni chiqarish mumkin. Satrli ma'lumotlarni ekranga chiqarish uchun, ularni qo'shtirnoq orasida yozish kerak.

Quyida a va b sonlarining yig'indisini chiqaruvchi dastur berilgan:

```
#include <iostream>
// standart nomlar fazosidan foydalanishni e'lon qilish  using namespace std;
int main()
{
   int a, b, c;
cout << "a="; cin >> a;     cout << "b="; cin >> b;     c = a + b;     cout << c << endl;
   return 0;
}
```

Ba'zi matematik funksiyalar:
Matemetik funksiyalardan dasturda foydalanish uchun math.h faylini progarmmaga qo'shish kerak. **#include <math.h>**

Matematik funksiyalardan foydalanish

```
#include <iostream>  #include <math.h>  using namespace std;
int main()
{      float a;
```

```
    cout << "a="; cin >> a;      a = sqrt(a);
    cout << a << endl;
    return 0;
}
```

Dasturchilar doim dastur ishlashi jarayonida xotiradan kamroq joy talab qilishligi haqida bosh qotirishadi. Bu muammolar dasturdagi o'zgaruvchilar sonini kamaytirish, yoki o'zgaruvchilar saqlanadigan yacheyka hajmini kamaytirish orqali erishiladi. Biz butun va haqiqiy sonlarni e'lon qilishni bilamiz. Bulardan tashqari C++ da butun va haqiqiy sonlarni e'lon qilish uchun bir nechta toifalar mavjud. Ular bir - biridan kompyuter xotirasida qancha hajm egallashi va qabul qiluvchi qiymatlar oralig'i bilan farq qiladi.

Butun sonlar

Toifa ko'rinishi	Qabul qiladigan qiymatlar oralig'i	Kompyuter xotirasida egallagan hajmi
unsigned short int	0..65535	2 bayt
short int	-32768..32767	2 bayt
unsigned long int	0..42949667295	4 bayt
long int	-2147483648..2147483647	4 bayt
int (16 razryadli)	-32768..32767	2 bayt
int (32 razryadli)	-2147483648..2147483647	4 bayt
unsigned int (16 razryadli)	0..65535	2 bayt
unsigned int (32 razryadli)	0..42949667295	4 bayt

Haqiqiy sonlar

Toifa ko'rinishi	Qabul qiladigan qiymatlar oralig'i	Kompyuter xotirasida egallagan hajmi
float	1.2E-38..3.4E38	4 bayt
double	2.2E-308..1.8E308	8 bayt
long double (32 razryadli)	3.4e-4932..-3.4e4932	10 bayt

Boshqa toifalar

Toifa ko'rinishi	Qabul qiladigan qiymatlar oralig'i	Kompyuter xotirasida egallagan hajmi
bool	**true** yoki **false**	1 bayt
char	0..255	1 bayt
void	2 yoki 4	

Har xil toifadagi o'zgaruvchilar kompyuter xotirasida turli xajmdagi baytlarni egallaydi. Xattoki bir toifadagi o'zgaruvchilar ham qaysi kompyuterda va qaysi operatsion sistemada ishlashiga qarab turli o'lchamdagi xotirani egallashi mumkin. C++ da ixtiyoriy toifadagi o'zgaruvchilarning o'lchamini sizeof funksiyasi orqali aniqlash mumkin. Bu funksiyani o'zgarmasga, biror toifaga va o'zaruvchiga qo'llash mumkin.

Toifalarni kompyuter xotirasida egallagam xajmini aniqlash #include <iostream>

```
using namespace std;
int main()
{
   cout << "char  = " << sizeof(char)  << endl;      cout << "bool  = " << sizeof(bool) << endl;       cout << "int   = " << sizeof(int)   << endl;       cout << "float = " << sizeof(float) << endl;
   cout << "double= " << sizeof(double)<< endl;       return 0;
}
```

Natija: ***char=1 bool=1 int=4 float=4 double=8***

Matemetik funksiyalardan dasturda foydalanish uchun math.h sarlavha faylini progarmmaga qo'shish kerak. #include <math.h>

Funksiyaning C++ da ifodalanishi	Funksiyaning matematik ifodalanishi
1. abs(x) - butun sonlar uchun 2. fabs(x) - haqiqiy sonlar uchun 3. labs(x) - uzun butun son uchun	/x/
pow(x, y)	x^y
pow10(x)	10^x
sqrt(X)	$\sqrt{X}$
ceil(x)	haqiqiy toifadagi x o'zgaruvchisi qiymatini unga eng yaqin katta butun songa aylantiradi.
floor(x)	haqiqiy toifadagi x o'zgaruvchisi qiymatini unga eng yaqin kichik butun songa aylantiradi
cos(x)	x burchak kosinusini aniqlash. x radian o'lchovida.
sin(x)	x burchak sinusini aniqlash. x radian o'lchovida.
exp(x)	e^x
log(x)	x sonining natural logarifmini qaytaradi.
log10(x)	x sonining 10 asosli logarifmini qaytaradi.

Eslatma: Barcha trigonometrik funksiyalar radian o`lchovida beriladi.

1 - Misol: n va m natural sonlari berilgan. n sonini m soniga bo'lib, qoldiqni aniqlovchi dastur tuzilsin

```
#include <iostream>
int main()
{
   int n, m, qoldiq;       cout << "n="; cin >> n;       cout << "m="; cin >> m;       // % qoldiqni olishni bildiradi      qoldiq = n % m;
   cout << "Qoldiq=" << qoldiq << endl;      return 0;  }
```

2 - Misol: n va m natural sonlari berilgan. n sonini m soniga bo'lib, butun qismini aniqlovchi dastur tuzilsin

```
#include <iostream>
int main()
{
   int n, m, b;

   cout << "n="; cin >> n;      cout << "m="; cin >> m;
   b = n / m;
   cout << "Butun qismi=" << b << endl;      return 0;
}
```

3 - misol. a sonini b soniga bo`lib 2 xona aniqlikda chiqarish.

```
#include <iostream>
#include <iomanip>
// <iomanip> sarlavha faylini qo'shamiz   using namespace std;
int main()
{
   float a, b;
   cout << "a sonini b soniga bo`lib 2 xona aniqlikda chiqarish"<<endl;       cout << "a="; cin >> a;
   cout << "b="; cin >> b;
   a = a / b;      cout <<  a << endl;
   cout << setprecision(2) << fixed << a << endl;      return 0;
}
```

Natija:

```
a sonini b soniga bo`lib 2 xona aniqlikda chiqarish
a=10
b=3
3.33333
3.33

Process returned 0 (0x0)   execution time : 6.723 s
Press any key to continue.
```

4 - misol. Bir toifadan boshqasiga o'tish c++ da bir toifadan boshqasiga o'tishning oshkor va oshkormas usullari mavjud. Oshkor ravishda toifaga keltirish uchun qavs ichida boshqa toifa nomi yoziladi.

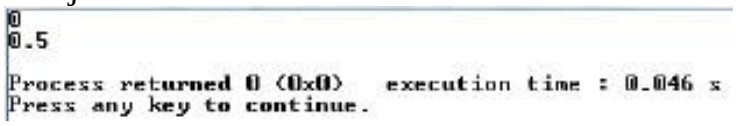
```
#include <iostream>  using namespace std;
int main()
{
   float haqiqiy = 5.57; int oshkor, oshkormas;
    // oshkormas ravishda butun toifaga o'tish      oshkormas = haqiqiy;
```

oshkor = (int) haqiqiy; // oshkor holda butun toifaga o'tish cout << "haqiqiy = " << haqiqiy << endl; cout << "oshkor = " << oshkor << endl;
cout << "oshkormas = " << oshkormas << endl; return 0;
}

5 - misol. Butun sonni bo'lish #include <iostream> using namespace std; int main() { int bir = 1;
int ikki = 2;

cout << bir / ikki << endl;
cout << ((float)bir) / ((float)ikki) << endl; return 0;
}

Natija:

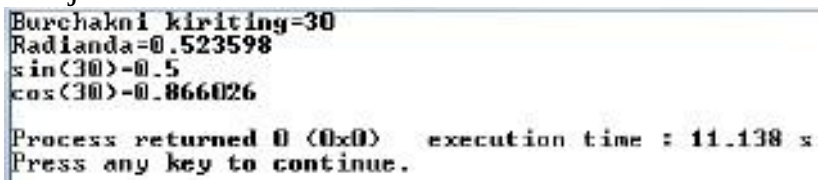


6 - misol. Trigonometrik funksiyalar bilan ishlash
#include <iostream> #include <math.h> using namespace std; int main() { float const pi = 3.14159;
float burchak, burchak_radian;

cout << "Burchakni kiriting="; cin >> burchak; burchak_radian = burchak * pi / 180;

cout << "Radianda=" << burchak_radian << endl; cout << "sin(" << burchak << ")=" << sin(burchak_radian) << endl; cout << "cos(" << burchak << ")=" << cos(burchak_radian) << endl; return 0;
}

Natija:



SAVOLLAR:

1. cin funksiyasining vazifasi nima?
2. cout funksiyasining vazifasi nima?
3. pow funksiyasining vazifasi nima?
4. sqrt funksiyasining vazifasi nima?
5. sin funksiyasining vazifasi nima?
6. cos funksiyasining vazifasi nima?
7. tan funksiyasining vazifasi nima?
8. atan funksiyasining vazifasi nima?
9. exp funksiyasining vazifasi nima?
10. log funksiyasining vazifasi nima?
11. log10 funksiyasining vazifasi nima?
12. ceil funksiyasining vazifasi nima?

13. floor funksiyasining vazifasi nima?
14. getch() funksiyasining vazifasi nima?
15. rename funksiyasining vazifasi nima?
16. Ikki sonning yig`indisi va ko`paytmasini topuvchi dastur tuzing.
17. Ikki sonning ayirmasi va bo`linmasini topuvchi dastur tuzing.
18. A sonini B soniga bo`lgandagi butun qism va qoldiqni toping.
19. A sonining B- darajasini hisoblang.
20. Ikki nuqta orasidagi masofani topuvchi dastur tuzing.

VARIANTLAR

1 - talaba savollari: 20; 1; 18; 2; 14; 19; 4;
2 - talaba savollari: 10; 5; 3; 15; 4; 20; 7;
3 - talaba savollari: 18; 2; 4; 7; 19; 12; 9;
4 - talaba savollari: 15; 11; 3; 5; 12; 18; 10;
5 - talaba savollari: 10; 19; 17; 8; 18; 5; 2;
6 - talaba savollari: 18; 6; 1; 14; 9; 7; 4;
7 - talaba savollari: 6; 7; 16; 3; 11; 2; 20;
8 - talaba savollari: 13; 1; 18; 10; 19; 11; 12;
9 - talaba savollari: 15; 13; 8; 18; 6; 10; 14;
10 - talaba savollari: 4; 10; 14; 20; 2; 7; 19;
11 - talaba savollari: 1; 19; 14; 2; 15; 20; 17;
12 - talaba savollari: 8; 19; 4; 9; 15; 18; 12;
13 - talaba savollari: 20; 18; 13; 2; 19; 16; 7;
14 - talaba savollari: 9; 10; 17; 4; 14; 15; 7;
15 - talaba savollari: 6; 1; 2; 4; 18; 16; 8;

2-Laboratoriya mashg„uloti.
MAVZU: C++ TILIDA IF OPERATORI BILAN ISHLASH.

Ishning maqsadi: Talabalarga C++ tilida if operatoridan foydalanib dasturlar tuzishni o`rgatish.

Nazariya bo`yicha qisqacha ma‘lumot

Shartli operator. Shartli operator ikki ko‘rinishda ishlatilishi mumkin:
If (ifoda) 1- operator Else 2- operator eki
If (ifoda) 1-operator
Shartli operator bajarilganda avval ifoda hisoblanadi ; agar qiymat rost ya'ni nol'dan farqli bo‘lsa 1- operator bajariladi. Agar qiymat yolg‘on ya'ni nol' bo‘lsa va else ishlatilsa 2-operator bajariladi. Else qism har doim eng yaqin if ga mos qo‘yiladi.
if(n>0) if(a>b) Z=a; else Z=b;
Agar else qismni yuqori if ga mos quyish lozim bo‘lsa, figurali qavslar ishlatish lozim. if(n>0) { if(a>b) z=a; }
else z=b;

Misol tariqasida uchta berilgan sonning eng kattasini aniqlash dasturini ko‘ramiz:

```
#include <iostream.h> void( )
 {  float a,b,c,max);
   Cout <<—\n a=‖; Cin>>a;
   Cout <<—\n b=‖; Cin>>b;       Cout <<—\n c=‖; Cin>>c;    if (a>b)        if (a>c)
max=a else max=c;   else      if b>c then max=b else max=c;    Cout <<—\n‖ <<max;
}
```

Keyingi misolda kiritilgan ball va maksimal ball asosida baho aniqlanadi:

```
#include <iostream.h> void main( )
 {  float ball,max_ball,baho);
   Cout<< —\n ball=‖; Cin>>(—%f‖,&ball);
   Cout<<—\n max_ball=‖; Cin>>max_ball;
   d=ball/max_ball;  if (d>0.85) baho=5 else      if (d>75) baho=4 else
   if (d>0.55) then baho=3 else baho=2;    Cout<<—\n baho;
 }
```

Kalit bo‘yicha tanlash operatori. Kalit bo‘yicha o‘tish switch operatori umumiy ko‘rinishi qo‘yidagicha

```
    Switch(<ifoda>) {
    Case <1-kiymat>:<1-operator>
            …       break;
            …
     default: <operator>
            …
     case: <n-operator>;
```

}

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

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

```
Include <iostream.h>
Int baho;
Cin>> baho;
Switch(baho)
{case 2:Cout <<—\n emon‖;break;  case 3:Cout <<—\n urta‖;break;  case 4:Cout <<—\n yahshi‖;break;  case 5:Cout <<—\n a'lo‖;break;
 default: Cout <<—\n baho notugri kiritilgan‖; }; }
```

Keyingi misolimizda kiritilgan simvol unli harf ekanligi aniqlanadi:

```
Include <iostream.h>
Int baho; Char c; Cin >> c; Switch(c)
{case _a‘: case _u‘:  case _o‘:  case _i‘:
Cout <<—\n Kiritilgan simvol unli harf‖;break; default: Cout <<—\n Kiritilgan simvol unli harf emas‖; };
}
```

Kvadrat tenglama ildizlari topish dasturi

```
#include <iostream.h>
#include <math.h> #include <conio.h>        int main()
                {
         int a,b,c;      float D,x1,x2;
                cout <<"ax^2+bx+c=0 tenglama ildizini topish dasturi!";
         cout<<"\n a - koeffitsientni kiriting: "; cin>>a;      cout<<"\n b - koeffitsientni kiriting: "; cin>>b;           cout<<"\n c - koeffitsientni kiriting: "; cin>>c;      D = b*b - 4 * a * c;
         if (D<0) { cout << "Tenglama haqiqiy ildizlarga ega emas";       getch(); return 0; }
                  if (D==0) { cout << "Tenglama yagona ildizga ega: ";
    x1=x2= -b / (2 * a);      cout<<"\n  x= "<<x1;
     getch(); return 0; }
                   else
      {cout << "Tenglama ikkita ildizga ega: ";          x1 = (- b + sqrt(D)) / (2 * a);
          x2 = (- b - sqrt(D)) / (2 * a);
       cout<<"\n x1= "<<x1;          cout<<"\n x2= "<<x2;
                  }
     getch();       return 0;
```

}

SAVOLLAR

1. max(x, y);
2. min(x, y) ;
3. max(x, y)+min(x, y).
4. max(x, y, z);
5. min(x, y, z);
6. max(x+y+z, xyz);
7. min(x+y/2+z/3, x-2y+z, x-y-z);
8. a, b va c haqiqiy sonlar berilgan bo_lsin. a<b<c munosabat o´rinlimi?
9. Uchta o_zaro har xil sonlarning yig_indisi birdan kichik bo_lsa, berilgan sonlarning eng kichigi, aks holda eng kattasi topilsin.
10. Uchta a, b va c haqiqiy sonlar berilgan bo_lsin. Tomonlari shu sonlarga teng uchburchak mavjudmi? Mavjud bo_lsa, uning perimetri va yuzi topilsin.
11. a, b va c haqiqiy sonlar berilgan bo_lsin. ax 4 +bx 2 +c=0 (a≠0) bikvadrat tenglamani to_la tekshiring.
12. Kunning K (k≤86400) soniyasi o_tib bormoqda. Tushlik-kacha qancha vaqt qolganligini soat va minutlarda aniqlang. Tushlik vaqti 12.00.00 hisoblanishi va uni o_tib ketgan bo_lishi mumkinligini nazarda tuting.

VARIANTLAR

1 - talaba savollari: 4; 6; 11; 7; 10; 1; 8;
2 - talaba savollari: 5; 6; 8; 1; 3; 4; 11;
3 - talaba savollari: 5; 2; 8; 6; 11; 10; 1;
4 - talaba savollari: 7; 12; 10; 5; 8; 1; 9;
5 - talaba savollari: 10; 3; 6; 11; 5; 8; 7;
6 - talaba savollari: 2; 11; 10; 12; 9; 8; 1;
7 - talaba savollari: 9; 10; 2; 1; 8; 11; 12;
8 - talaba savollari: 9; 5; 6; 11; 10; 2; 8;
9 - talaba savollari: 7; 12; 4; 5; 2; 1; 10;
10 - talaba savollari: 4; 1; 9; 12; 8; 6; 2;
11 - talaba savollari: 3; 11; 2; 4; 12; 5; 10;
12 - talaba savollari: 4; 3; 12; 1; 11; 5; 8;
13 - talaba savollari: 11; 3; 2; 6; 1; 8; 4;
14 - talaba savollari: 6; 1; 10; 4; 3; 11; 9; 15 - talaba savollari: 2; 9; 1; 5; 6; 10; 3;

3-Laboratoriya mashg„uloti.
Mavzu: C++ tilida For operatori bilan ishlash.

Ishning maqsadi: Talabalarga C++ tilida for operatoridan foydalanib dasturlar tuzishni or`gatish.
Nazariya bo`yicha qisqacha ma‘lumot

For strukturasi sanovchi (counter) bilan bajariladigan takrorlashni bajaradi.Boshqa takrorlash bloklarida (while, do/while) takrorlash sonini control qilish uchun ham sanovchini qo'llasa bo'lardi, bu holda takrorlanish sonini o'ldindan bilsa bo'lardi, ham boshqa bir holatning vujudga kelish-kelmasligi orqali boshqarish mumkin edi. Ikkinchi holda ehtimol miqdori katta bo'ladi. Masalan qo'llanuvchi belgilangan sonni kiritmaguncha takrorlashni bajarish kerak bo'lsa biz while li ifodalar-ni ishlatamiz. for da esa sanovchi ifodaning qiymati oshirilib (kamaytirilib) borilvuradi, va chegaraviy qiymatni olganda takrorlanish tugatiladi. for ifodasidan keyingi bitta ifoda qaytariladi. Agar bir necha ifoda takrorlanishi kerak bo'lsa, ifodalar bloki {} qavs ichiga olinadi.

```
 //Ekranda o'zgaruvching qiymatini yozuvchi dastur, for ni ishlatadi.
# include <iostream.h>
int main()
{  for (int i = 0; i < 5; i++){
  cout << i << endl;
 }    return (0); }
```

Ekranda:

```
0
1
2
3 4
```

for strukturasi uch qismdan iboratdir. Ular nuqtavergul bilan bir-biridan ajratiladi. for ning ko'rinishi:

```
 for( 1. qism ; 2. qism ; 3. qism ){   takror etiladigan blok
}
```

1. qism - e'lon va initsalizatsiya.
2. qism - shartni tekshirish (oz'garuvchini chegaraviy qiymat bilan solishtirish).

3.qism - o'zgaruvchining qiymatini o'zgartirish.

Qismlarning bajarilish ketma-ketligi quyidagichadir:
Boshida 1. qism bajariladi (faqat bir marta), keyin
2. qismdagi shart tekshiriladi va agar u true bo'lsa takrorlanish bloki ijro ko'radi, va eng ohirda 3. qismda o'zgaruvchilar o'zgartiriladi, keyin yana ikkinchi qismga o'tiladi. for strukturamizni while struktura bilan almashtirib ko'raylik:

```
for (int i = 0; i < 10 ; i++)
  cout << "Hello!"<< endl;
```

Ekranga 10 marta Hello! so'zi bosib chiqariladi. I o'zgaruvchisi 0 dan 9 gacha o'zgaradi.
i 10 ga teng bo'lganda esa i < 10 sharti noto'g'ri (false) bo'lib chiqadi va for strukturasi nihoyasiga yetadi.
Endi for ni tashkil etuvchi uchta qismninig har birini alohida ko'rib chiqsak.Birinchi qismda asosan takrorlashni boshqaradigan sanovchi (counter)o'zgaruvchilar e'lon qilinadi va ularga boshlangich qiymatlar beriladi (initsalizatsiya). Yuqoridagi dastur misolida buni int i = 0; deb berganmiz. Ushbu qismda bir necha o'zgaruvchilarni e'lon qilishimiz mumkin, ular vergul bilan ajratilinadi. Ayni shu kabi uchinchi qismda ham bir nechta o'zgaruvchilarning qiyma-tini o'zgartirishimiz mumkin. Undan tashqari birinchi qismda for dan oldin e'lon qilingan o'zgaruvchilarni qo'llasak bo'ladi.
Masalan:

```
int k = 10;  int l;
for (int m = 2, l = 0 ; k <= 30 ; k++, l++, ++m) { cout << k + m + l; }
```

Albatta bu ancha sun'iy misol, lekin u bizga for ifodasining naqadar moslashuvchanligi ko'rsatadi. for ning qismlari tushurib qoldirilishi mumkin. Masalan: for(;;) {} ifodasi cheksiz marta qaytariladi. Bu for dan chiqish uchun break operatorini beramiz. Yoki agar sanovchi sonni takrorlanish bloki ichida o'zgartirsak,for ning 3. qismi kerak emas. Misol:

```
for(int g = 0; g < 10; ){   cout << g;   g++;
}
```

Yana qo'shimcha misollar beraylik.

```
for (int y = 100; y >= 0; y-=5){
...
ifoda(lar);
...
}
```

Bu yerda 100 dan 0 gacha 5 lik qadam bilan tushiladi.

```
for(int d = -30; d<=30; d++){
...
ifoda(lar);
...
}
```

60 marta qaytariladi.
for strukrurasi bilan dasturlarimizda yanada yaqinroq tanishamiz. Endi
1. qismda e'lon qilinadigan o'zgaruvchilarning hususiyati haqida bir og'iz aytib o'taylik. Standartga ko'ra bu qismda e'lon qilingan o'zgaruvchilarning qo'llanilish sohasi faqat o'sha for strukturasi bilan chegaralanadi. Yani bitta blokda joylashgan for struk-turalari mavjud bo'lsa, ular ayni ismli o'zgaruvchilarni qo'llana ololmaydilar. Masalan quyidagi hatodir:

```
for(int j = 0; j<20 ; j++){...}
```

... for(int j = 1; j<10 ; j++){...} //hato! j o'zgaruvchisi birinchi for da e'lon qilinib bo'lindi. Ikkinchi for da ishlatish mumkin
emas. Bu masalani yechish uchun ikki hil yo'l tutish mumkin.
Birinchisi bitta blokda berilgan for larning har birida farqli o'zgaruvchilarni qo'llashdir. Ikkinchi yo'l for lar guruhidan oldin sanovchi vazifasini bajaruvchi bir o'zgaruvchini e'lon qilishdir. Va for larda bu o'zgaruvchiga faqat kerakli boshlangich qiymat beriladi halos.

for ning ko'rinishlaridan biri, bo'sh tanali for dir. for(int i = 0 ; i < 1000 ; i++);
Buning yordamida biz dastur ishlashini sekinlashtirishimiz mumkin.

SAVOLLAR:

1. Raqamlari yigindisining kubiga teng bo`lgan barcha uch xonali sonlarni toping.
2. 2,3,4,5,6,7,8,9 sonlariga ko`paytirilganda raqmlarining yig`indisi o`zgamaydigan ikki xonali sonlarni toping.
3. Raqamlari yig`indisi berilgan butun songa teng uch xonali barcha sonlarni toping.
4. Barcha shunday uch xonali sonlarni toping-ki, ularni kvadratga ko`targanda uchata raqami bir xil bo`lsin, nol bundan mustasno.
5. Do`konda 16 kgli, 17 kgli va 21 kgli pol buyoqlari yashiklarda mavjud. Ularni ochib ko`rmasdan 185 kg pol buyoq tanlahg. Barcha variantlarni ko`rib chiqing.
6. 42*4* yulduzchalar o`rniga shunday raqamlarni tanlang, hosil bo`lgan besh xonali son 72ga bo`linsin.
7. O`zi va raqamlari yig`indisi 7ga karrali uch xonali sonlarni toping.
8. Shunday to`rt xonali sonlarni toping-ki, 133ga bo`lganda qoldiq 125ga va 134ga bo`lganda qoldiq 111ga teng bo`lsin.
9. Raqamlari yig`indisi berilgan butun songa teng to`rt xonali barcha sonlarni toping.
10. O`zi va raqamlari yig`indisi 11ga karrali to`rt xonali sonlarni toping.
11. 42*4* yulduzchalar o`rniga shunday raqamlarni tanlang, hosil bo`lgan olti xonali son 11ga bo`linsin.
12. Berilgan natural sonni bo`luvchilari yig`indisini toping.
13. Berilgan N sonidan oshmaydigan barcha pifagor natural sonnlarni toping. (Agar berigan uch xonali sonning dastlabki ikkita raqami yig`indisining kvadrati uchinchi raqamiga teng bo`lsa, bunday natural son pifagor soni deb ataladi)
14. Birinchi ikkita raqamlari yig`indisi oxirgi ikkita raqamlari yig`indisiga teng bo`lgan barcha to`rt xonali sonlarni toping.

VARIANTLAR

1 - talaba savollari: 14; 10; 8; 7; 5;
2 - talaba savollari: 10; 11; 14; 9; 5;
3 - talaba savollari: 1; 9; 4; 13; 12;
4 - talaba savollari: 7; 4; 11; 5; 1;
5 - talaba savollari: 4; 9; 14; 10; 12;
6 - talaba savollari: 13; 8; 1; 6; 2;
7 - talaba savollari: 14; 3; 6; 12; 2;

8 - talaba savollari: 9; 1; 13; 2; 11;
9 - talaba savollari: 7; 9; 5; 10; 12;
10 - talaba savollari: 14; 2; 8; 12; 11;
11 - talaba savollari: 6; 11; 12; 14; 1;
12 - talaba savollari: 5; 14; 9; 2; 1;
13 - talaba savollari: 6; 13; 8; 2; 11;
14 - talaba savollari: 14; 3; 6; 12; 1;
15 - talaba savollari: 11; 7; 6; 10; 5;

4-Laboratoriya mashg`uloti

Mavzu: C++ tilida While operatori bilan ishlash.

Ishning maqsadi: Talabalarga C++ tilida While operatoridan foydalanib dasturlar tuzishni o`rgatish.

Nazariya bo`yicha qisqacha ma‘lumot

While operatori orqali murakkab konstruktsiyalarni tuzish. while operatori shartida murakkab mantiqiy ifodalarni ham qo_llash mumkin. Bunday ifodalarni qo_llashda && (mantiqiy ko_paytirish), || (mantiqiy qo_shish) , hamda !(mantiqiy INKOR) kabi operatsiyalardan foydalaniladi. Quyida while operatori konstruktsiyasida murakkabroq shartlarni quyilishiga misol keltirilgan .

while konstruktsiyasidagi murakkab shartlar.

```
#include <iostream>  using namespace std;
int main()
{
unsigned short kichik;  unsigned long katta;  const unsigned short MaxKichik=65535;  cout << "Kichik sonni kiriting:";  cin >> kichik;
 cout << "Katta sonni kiriting:";  cin >> katta;  cout << "kichik son:" << kichik << "…";  //Xar bir iteratsiyada uchta shart tekshiriladi.
 while (kichik<katta && katta>0 &&
   kichik< MaxKichik )
 {  if(kichik%5000==0) //Xar 5000 satrdan
  //keyin nuqta chikariladi  cout<<"." ;  kichik++;  katta-=2 ;
}
cout<<"\n kichik son:"<<kichik<<" katta son:"
<<katta << endl ;
return 0 ; }
```

Natija:

```
Kichik sonni kirit : 2
Katta sonni kirit : 100000
Kichik son : 2 ……
Kichik son :33335  katta son : 33334
```

TAHLIL

Dastur quyidagi mantiqiy o_yinni ifodalaydi. Oldin ikkita son – kichik va katta kiritiladi. Undan so_ng toki ular bir biriga teng bo_lmaguncha, ya‘ni «uchrashmaguncha» kichik son birga oshiriladi, kattasi esa ikkiga kamaytiriladi.

O‗yinni maqsadi qiymatlar «uchrashadigan» sonni topishdir. Qiymatlar kiritilgandan so‗ng siklni davom ettirishning quyidagi uchta sharti tekshiriladi:

- kichik o‗zgaruvchisi qiymati katta o‗zgaruvchisi qiymatidan oshmasligi.
- katta o‗zgaruvchisi qiymati manfiy va nolga teng emasligi
- kichik o‗zgaruvchisi qiymati MaxKichik qiymatidan oshib ketmasligi

So‗ngra kichik soni 5000 ga bo‗lingandagi qoldiq hisoblanadi. Agarda kichik 5000 ga qoldiqsiz bo‗linsa bu operatsiyaning bajarilishi natijasi 0 ga teng bo‗ladi. Bu holatda hisoblash jarayonini vizual ifodasi sifatida ekranga nuqta chiqariladi. Keyin esa kichik qiymati bittaga oshiriladi, katta qiymati esa 2 taga kamaytiriladi. Sikl agarda tekshirish sharti tarkibidagi birorta shart bajarilmasa to‗xtatiladi.

SAVOLLAR:

1. u=ln(1+x)= x-x2/2+x3/3-...+(-1)n-1xn/n+...(|x|<1) .
2. u=arctgx= x-x3/3+x5/5-...+(-1)n x2n+1/(2n+1)+...(|x|<1).
3. Bir-biridan farqli, uchtadan kam bo'lmagan natural sonlar ketma-ketligi berilgan bo'lib, u 0 bilan tugallanadi. Shu sonlar ichida uchta eng kattasi topilsin.
4. Nol bilan tugaydigan, noldan farqli butun sonlar ketma-ketligida ishora o'zgarishlari sonini aniqlaydigan dastur tuzilsin. (Masalan, 1,-34,8,14,-5,0 kesmalar kesishmasida ishora 3 marta o'zgaradi).
5. Berilgan 10 ta natural sonlarning eng katta umumiy bo'luvchisini topadigan dastur tuzilsin.
6. 7 so‘mdan katta bo‘lgan har qanday tiyinsiz pul miqdorini 3 va 5 so‘mliklar yig‘indisi bilan qaytimsiz to‘lash mumkinligi isbotlansin. Berilgan n>7 uchun shartni qanoatlantiruvchi, musbat va butun a,b sonlar juftliklari topilsin.
7. Hadlar soni ikkitadan kam bo'lmagan nol bilan tugaydigan natural sonlar ketmaketligi berilgan. Tartib nomerlari tub sonlar bo'lgan hadlarining yig'indisini aniqlaydigan dastur tuzilsin.
8. Hadlar soni ikkitadan kam bo'lmagan nol bilan tugaydigan natural sonlar ketmaketligi berilgan. Tartib nomerlari tub sonlar bo'lgan hadlarining yig'indisini aniqlaydigan dastur tuzilsin.
9. Berilgan natural son raqamlarining yig'indisini hisoblaydigan dastur tuzilsin.
10. Berilgan natural sonning raqamlarini teskari tartibda yozishdan hosil bo'ladigan sonni aniqlaydigan dastur tuzilsin.
11. Berilgan natural sonning palindrom ekanligini, ya'ni o'ngdan o'qiganda ham, chapdan o'qiganda ham bir xil son bo'lgan natural sonlarni aniqlaydigan dastur tuzilsin.
12. Quyida berilgan ketma-ketliklarning k-raqamini chop etadigan dastur tuzilsin:
    a) 12345678910111213...-ketma-ket yozilgan natural sonlar;
    b) 1123581321...- Fibonachchi sonlari.

VARIANTLAR

1 - talaba savollari: 10; 5; 1; 2; 9;
2 - talaba savollari: 4; 10; 3; 6; 5;
3 - talaba savollari: 4; 11; 12; 9; 1;

4 - talaba savollari: 2; 5; 11; 3; 6;
5 - talaba savollari: 12; 2; 9; 1; 7;
6 - talaba savollari: 5; 10; 4; 6; 8;
7 - talaba savollari: 5; 3; 10; 12; 2;
8 - talaba savollari: 10; 1; 7; 4; 12;
9 - talaba savollari: 10; 7; 4; 5; 3;
10 - talaba savollari: 3; 11; 10; 8; 9;
11 - talaba savollari: 6; 4; 11; 9; 10;
12 - talaba savollari: 12; 2; 6; 1; 7;
13 - talaba savollari: 11; 9; 1; 5; 8;
14 - talaba savollari: 3; 11; 2; 8; 4;
15 - talaba savollari: 3; 6; 7; 2; 10;

5-Laboratoriya mashg`uloti
Mavzu: C++ tilida Do-while operatori bilan ishlash.

Ishning maqsadi: Talabalarga C++ tilida Do-while operatoridan foydalanib dasturlar tuzishni o`rgatish
Nazariya bo`yicha qisqacha ma‘lumot

do - while operatorining umumiy ko'rinishi : do { operator; } while (shart);
Bu yerda do va while xizmatchi so`zlar. ( shart ) sikl tanasi bajarilgandan so`ng, sikldan chiqish uchun tekshiriladigan shart. (mantiqiy ifoda).
do - while operatorning ishlash tartibi:
do xizmatchi so`zidan keyingi operatorlar bajariladi, keyin while xizmatchi so'zidan keyingi shart tekshiriladi. Agar shart rost (true) natija bersa do xizmatchi so`zidan keyingi operatorlar qayta bajariladi. Shart qayta tekshiriladi, bu jarayon shart yolg'on (false) natija berguncha takrorlanadi. Qachon while xizmatchi so'zidan keyingi shart yolg'on (false) qiymatga ega bo'lsa, boshqarilish do - while operatoridan keyingi operatorga uzatiladi.
do - while sikl operatorida sikllanib qolish
DIQQAT: do - while sikl operatoridan, qachon while xizmatchi so'zidan keyingi (shart) false (yolg'on) qiymat qabul qilsa chiqiladi. Ya'ni boshqarilish do - while operatoridan keyingi operatorga uzatiladi. Agar (shart) false qiymat qabul qilmasa, do - while sikl operatoridan chiqib ketilmaydi va bu jarayon sikllanib qolish deyiladi.
1 dan 10 gacha bo'lgan sonlarni chiqaruvchi dastur tuzilsin.

```
#include <iostream>  using namespace std;
int main()
{    int i = 1;    do {
   cout << i << endl;      i++;
} while ( i <= 10);
```

```
  return 0;
}
```

Misol. Quyidagi yig`indini hisoblovchi dastur tuzilsin.

Bu dastur parametrli sikl operatoridan foydalangan holda oldingi darsda tuzilgan edi. Endi do - while sikl operatori orqali dastur tuzamiz va sikl operatorlarini farqini ko`rib olamiz.

```
#include <iostream>  using namespace std;
int main()
{    float i = 1; // i - sikl uchun
  float s = 0; // s - yig'indi

  do {      s += 1 / i;      i++;
  } while ( i <= 50);

  cout << "yig`indi = " << s << endl;    return 0;
}
```

do…while konstruktsiyasi yordamida sikl tashkil etish. Ayrim hollarda while operatori yordamida sikllarni tashkil etishda uning tanasidagi amallar umuman bajarilmasligi mumkin. Chunki siklni davom etish sharti har bir iteratsiyadan oldin tekshiriladi. Agarda boshlang_ich berilgan shart to_g_ri bo_lmasa sikl tanasining birorta operatori ham bajarilmaydi.

do...while konstruktsiyasining qo_llanilishi

```
#include <iostream>  using namespace std;
int main()
{  int counter;
cout<<"How manu hellos ?";
cin >>counter;
do  {
cout << "hello \n";  counter--;
}
while(counter>0);
cout << "Counter is :" << counter <<endl;  return 0 ;
}
```

Natija: how manu hellos ? 2
hello hello Sounter is : 0
How manu hellos ? 0
Hello
Counter is: - 1

SAVOLLAR:

13. u=ln(1+x)= x-x2/2+x3/3-...+(-1)n-1xn/n+...(|x|<1) .

14. u=arctgx= x-x3/3+x5/5-...+(-1)n x2n+1/(2n+1)+...(|x|<1).
15. Bir-biridan farqli, uchtadan kam bo'lmagan natural sonlar ketma-ketligi berilgan bo'lib, u 0 bilan tugallanadi. Shu sonlar ichida uchta eng kattasi topilsin.
16. Nol bilan tugaydigan, noldan farqli butun sonlar ketma-ketligida ishora o'zgarishlari sonini aniqlaydigan dastur tuzilsin. (Masalan, 1,-34,8,14,-5,0 kesmalar kesishmasida ishora 3 marta o'zgaradi).
17. Berilgan 10 ta natural sonlarning eng katta umumiy bo'luvchisini topadigan dastur tuzilsin.
18. 7 so‘mdan katta bo‘lgan har qanday tiyinsiz pul miqdorini 3 va 5 so‘mliklar yig‘indisi bilan qaytimsiz to‘lash mumkinligi isbotlansin. Berilgan n>7 uchun shartni qanoatlantiruvchi, musbat va butun a,b sonlar juftliklari topilsin.
19. Hadlar soni ikkitadan kam bo'lmagan nol bilan tugaydigan natural sonlar ketmaketligi berilgan. Tartib nomerlari tub sonlar bo'lgan hadlarining yig'indisini aniqlaydigan dastur tuzilsin.
20. Hadlar soni ikkitadan kam bo'lmagan nol bilan tugaydigan natural sonlar ketmaketligi berilgan. Tartib nomerlari tub sonlar bo'lgan hadlarining yig'indisini aniqlaydigan dastur tuzilsin.
21. Berilgan natural son raqamlarining yig'indisini hisoblaydigan dastur tuzilsin.
22. Berilgan natural sonning raqamlarini teskari tartibda yozishdan hosil bo'ladigan sonni aniqlaydigan dastur tuzilsin.
23. Berilgan natural sonning palindrom ekanligini, ya'ni o'ngdan o'qiganda ham, chapdan o'qiganda ham bir xil son bo'lgan natural sonlarni aniqlaydigan dastur tuzilsin.
24. Quyida berilgan ketma-ketliklarning k-raqamini chop etadigan dastur tuzilsin:
    a) 12345678910111213...-ketma-ket yozilgan natural sonlar;
    b) 1123581321...- Fibonachchi sonlari.

VARIANTLAR

1 - talaba savollari: 10; 5; 1; 2; 9;
2 - talaba savollari: 4; 10; 3; 6; 5;
3 - talaba savollari: 4; 11; 12; 9; 1;
4 - talaba savollari: 2; 5; 11; 3; 6;
5 - talaba savollari: 12; 2; 9; 1; 7;
6 - talaba savollari: 5; 10; 4; 6; 8;
7 - talaba savollari: 5; 3; 10; 12; 2;
8 - talaba savollari: 10; 1; 7; 4; 12;
9 - talaba savollari: 10; 7; 4; 5; 3;
10 - talaba savollari: 3; 11; 10; 8; 9;
11 - talaba savollari: 6; 4; 11; 9; 10;
12 - talaba savollari: 12; 2; 6; 1; 7;
13 - talaba savollari: 11; 9; 1; 5; 8;
14 - talaba savollari: 3; 11; 2; 8; 4;
15 - talaba savollari: 3; 6; 7; 2; 10;

6-Laboratoriya mashg’uloti.
Mavzu: C++ tilida bir o ‘lchovli, ikki o ‘lchovli massivlar.

Ishning maqsadi: Talabalarga C++ tilida bir o‗lchovli va ikki o‗lchovli massivlar ustida amallar bajarishni o`rgatish.

Nazariya bo`yicha qisqacha ma‘lumot

Massivlar. Bir o`lchamli massivlar

Massiv - bu bir xil toifali, chekli qiymatlarning tartiblangan to`plamidir. Massivlarga misol qilib matematika kursidan ma`lum bo`lgan vektorlar, matritsalarni ko`rsatish mumkin.

Massiv bir o`lchamli deyiladi, agar uning elementiga bir indeks orqali murojaat qilish mumkin bo`lsa.

Bir o`lchamli massivni e`lon qilish quyidagicha bo`ladi:

<toifa> <massiv_nomi> [elementlar_soni] = { boshlang'ich qiymatlar }; Quyida massivlarni e`lon qilishga bir necha misollar keltirilgan: 1) float a[5];

2) int m[6];
3) bool b[10]; 1) a elementlari haqiqiy sonlardan iborat bo`lgan, 5 ta elementdan tashkil topgan massiv. Indekslari esa 0 dan 4 gacha bo`lgan sonlar

2) m elementlari butun sonlardan iborat bo`lgan, 6 ta elementdan tashkil topgan massiv. Indekslari esa 0 dan 5 gacha bo`lgan sonlar.
3) b elementlari mantiqiy qiymatlardan (true, false) iborat bo`lgan 10 ta elementdan tashkil topgan massiv. Indekslari esa 0 dan 9 gacha bo`lgan sonlar. Massiv elementlariga murojaat qilish oddiy o`zgaruvchilarga murojaat qilishdan biroz farq qiladi. Massiv elementiga murojaat qilish uning indeksi orqali bo`ladi.

a[1] = 10; a massivining 1 – elementi 10 qiymat o‘zlashtirsin; cin >> a[2]; a massivining 2 – elementi kirtilsin;

cout << a[3]; a massivining 3 – elementi ekranga chiqarilsin;

Massivni e'lon qilishda uning elementlariga boshlang'ich qiymat berish mumkin va buning bir nechta usuli mavjud.

1) O'lchami ko'rsatilgan massivni to'liq initsializatsiyalash. int k[5] = { 2, 3, 7, 8, 6};

Bu yerda 5 ta elementdan iborat bo'lgan k massivi e'lon qilingan va massivning barcha elemantlariga boshlang'ich qiymat berilgan.

2) O'lchami ko'rsatilgan massivni to'liqmas initsializatsiyalash.

int k[5] = { 2, 3, 7 };

Bu yerda 5 ta elementdan iborat bo'lgan k massivi e'lon qilingan va massivning dastlabki 3 ta elemantlariga boshlang'ich qiymat berilgan.

3) O'lchami ko'rsatilmagan massivni to'liq initsializatsiyalash.

int k[] = { 2, 3, 7, 8, 6};

Shuni takidlash lozimki, agar massiv o'lchami ko'rsatilmasa, uni to'liq initsializatsiyalash shart. Bu xolda massiv o'lchami kompilyatsiya jarayonida massiv elementlari soniga qarab aniqlanadi. Bu yerda massiv o'lchami 5 ga teng.

4) O'lchami ko'rsatilgan massivning barcha elementlariga boshlang'ich qiymat 0 berish:

```
int k[5] = { 0 };
```

O'lchami ko'rsatilgan massivning barcha elementlariga boshlang'ich qiymat 0 berish

```
#include <iostream>  using namespace std;
int main()
{
  int a[10] = { 0 };
  //massivning barcha elementlariga 0 qiymat berish
   for (int i = 0; i < 10; i++)
  cout << "a[" << i << "]=" << a[i] << endl;     return 0;  }
```

Agar massiv elementlariga boshlang'ich qiymatlar berilmasa xatolik sodir bo'lishi mumkin.

Elementlari butun sonlardan iborat bo`lgan, n elementdan tashkil topgan massiv elementlarini kirituvchi va ekranga chiqaruvchi dastur tuzilsin. (n <= 10)

```
#include <iostream>  using namespace std;
int main()
{
  int a[10] = { 0 };
  int n;
  cout << "n="; cin >> n;    for (int i = 0; i < n; i++)    {
    cout << "a[" << i << "]=";
    cin >> a[i];
  }
  for (int i = 0; i < n; i++)    cout << a[i] << " ";
  return 0;
}
```

n ta elementdan tashkil topgan massiv berilgan. Shu massiv elementlari yig'indisini chiqatuvchi dastur tuzilsin. (n <= 10)

```
#include <iostream>  using namespace std;
int main()
{
  int a[10] = { 0 }; // a massivini e'lon qilish    int n;   // massiv elamentlari soni
  int s = 0; // massiv elementlari yig'indisini hisoblash uchun     cout << "n="; cin >>
n;    for (int i = 0; i < n; i++)
  {
    cout << "a[" << i << "]=";
    cin >> a[i];      s += a[i];
  }
   cout << "Massiv elementlari yig`indisi = " << s <<  endl;    return 0;
}
```

Berilgan a[n] massivning ixtiyoriy tartib raqamda joylashgan elementini ekranga chiqarish dasturini tuzing.

```
#include<iostream.h> #include<conio.h> int main ()
{   int i,a[20],n;   for(i=1;i<=5;i++){   cout<<i<<"-element:";
  cin>>a[i];
}
 cout<<"qaysi elementni chiqarishni hohlaysiz:";  cin>>n;   cout<<a[n];   getch();
}
```

Massivda musbat elemenlar soni va summasini hisoblash.

```
#include <iostream> using namespace std; int main()
int s=0,k=0; int x[]={-1,2,5,-4,8,9}; for(int i=0; i<6; i++)
{
if (x[i]<=0) continue;
k++; s+=x[i];
};
cout<<k<<endl; cout<<s; return 0;
};
```

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

```
#include <iostream.h>
Void main()
{
Int I,j,n;
Float a,b,d,x[100];
While(1)
{
Cout<<(—\n n=‖); Cin>>(—%i‖,&n);
If ( n>0 && n < = 100 ) break;
Cout<<(—\n Hato 0<n<101 bulishi kerak‖); }
Cout<<(—\n elementlar kiymatlarini kiriting:\n‖); For (i=0;i<n;i++) {
Cout<<(—x[%i]=‖,i);Cin>>(—%f‖,&x[i]);} max=x[0];min=x[0];
For (s=0,i=0;i<n;i++)
{ s++;
 If (max<x[i]) max=x[i];
 If (min>x[i]) min=x[i];
}; s/=n;
Cout<<(—\n max=%f‖,max);
Cout<<(—\n min=%f‖,min);
Cout<<(—\n urta kiymat=%f‖,s);
}
```

C++tilida ko'p o'lchovli massivlar.

Ikki o`lchovli massivlar matematikada matritsa yoki jadval tushunchasiga mos keladi.

Jadvallarning initsializatsiya qilish qoidasi, ikki o‘lchovli massivning elementlari massivlardan iborat bo‘lgan bir o‘lchovli massiv ta'rifiga asoslangandir. Misol uchun ikki qator va uch ustundan iborat bo‘lgan haqiqiy tipga tegishli d massiv boshlang‘ich qiymatlari quyidagicha ko‘rsatilishi mumkin:
float d[2][3]={(1,-2.5,10),(-5.3,2,14)};
Bu yozuv quyidagi qiymat berish operatorlariga mosdir:
d[0][0]=1;d[0][1]=-2.5;d[0][2]=10;d[1][0]=-5.3;d[1][1]=2;d[1][2]=14; Bu qiymatlarni bitta ro‘yhat bilan hosil qilish mumkin: float d[2][3]={1,-2.5,10,-5.3,2,14};
Initsializatsiya yordamida boshlang‘ich qiymatlar aniqlanganda massivning hamma elementlariga qiymat berish shart emas.
Misol uchun: int x[3][3]={(1,-2,3),(1,2),(-4)}.
Bu yozuv qo‘yidagi qiymat berish operatorlariga mosdir: x[0][0]=1;x[0][1]=-2;x[0][2]=3;x[1][0]=-1;x[1][1]=2;x[2][0]=-4;
Initsializatsiya yordamida boshlang‘ich qiymatlar aniqlanganda massivning birinchi indeksi chegarasi ko‘rsatilishi shart emas, lekin qolgan indekslar chegaralari ko‘rsatilishi shart. Misol uchun:
Double x[][2]={(1.1,1.5),(-1.6,2.5),(3,-4)}
Bu misolda avtomatik ravishda qatorlar soni uchga teng deb olinadi.

Quyidagi ko‘radigan misolimizda jadval kiritilib har bir qatorning maksimal elementi aniqlanadi va bu elementlar orasida eng kichigi aniqlanadi:

```
#include <iostream.h> void main()
{ double a[4,3]; double s,max=0.0,min=0.0; int i,j;
for(i=0;i<4;i++) { for(j=0;j<3;j++) { Cout<<(—
a[%d][%d]=‖,i,j);Cin>>(—%f‖,s);a[i,j]=s; if (max<s) max=s;
};
Cout<<(—\n‖); if (max<min) min=max; }
Cout<<(—\n min=%f‖,min);
}
```

Misol. A(mxn) matritsa berilgan. Shu matritsa elementlarini kirituvchi va ekranga jadval ko`rinishida chiqaruvchi dastur. #include <iostream> using namespace std;

```
int main()
{
 int m, n, a[10][10];
 cout << "Satrlar sonini kiriting \nm="; cin >> m;
 cout << "Ustunlar sonini kiriting \nn=";   cin >> n;

 cout <<"Massiv elementlarini kiriting \n";   for(int satr = 0; satr < m ; satr++)
 for(int ustun = 0; ustun < n; ustun++)
 {
  cout << "a[" << satr << "][" << ustun << "]=";
  cin >> a[satr][ustun];
 }
 // matritsani jadval shaklida chiqarish   for(int satr = 0; satr < m; satr++)
```

```
{
 for(int ustun = 0; ustun < n; ustun++)    cout << a[satr][ustun] << "\t";    cout<<"\n";
 }   return 0;
}
```

Matritsaning har bir qatorining maksimal elementi aniqlash dasturi

```
#include <iostream> using namespace std; int main() {  double a[4][3];  double max;
int i,j;
for(i=0;i<4;i++)
{
for(j=0;j<3;j++)
{
cout<<"a["<<i<<"]["<<j<<"]=";
cin>>a[i][j]; } cout<<'\n'; };
for(i=0;i<4;i++)
{ max=a[i][0];
for(j=0;j<3;j++)

if (max<a[i][j]) max=a[i][j];
cout<<max<<endl;
} return 0;
}
```

A(i,j) massivning dioganol elementlari yig`indisini topish dasturi.

```
#include <iostream> #include <conio.h> using namespace std; int main() {
  int a[10][10]={0};    int i,j,m,n,sum=0,sum_=0;     cout << "Satr va ustunlar sonini kiriting m="; cin >> m;    for(i=1;i<=m;i++)    {   for(j=1;j<=m;j++)
   {  cout<<"a["<<i<<"]["<<j<<"]=";      cin>>a[i][j]; }  cout<<'\n';  } for(i = 1; i <= m;  i++){    for(j  =  1;  j  <=  m;  j++)    cout  <<  a[i][j]  <<  "\t";    cout<<"\n";} for(i=1;i<=m;i++)     {     for(j=1;j<=m;j++)        {     if (i==j) sum+=a[i][j];      if (i==m+1-j) sum_+=a[i][j];    }  } cout<<"1-dioganal= "<<sum<<endl<<"2-dioganal= "<<sum_; getch(); return 0;
}
```

SAVOLLAR:

1. A va B ikkita vektorning elementlari butun sonlarda iborat. A vektorning toq elementlardan tashkil topgan, lekin B vektorning elementi bo`lmagan C vektorni hosil qiling.
2. Qo`shimcha, massivdan foydalanmasdan turib, massiv elementlarini teskari tartibda joylashtiring.
3. X(N) massivning eng kichik va eng katta elementlarini ruyxatdan chiqarib, massivning o`rta arifmetigini hisoblang.
4. B(N) massiv elementlarini, B massiv elementlari yig`indisiga bo`lib, yangi elementlardan tashkil topgan B massivni hosil qiling.
5. X(M) vektorning eng katta elementigacha bo`lgan barcha elementlarini nol bilan almashtiring.

6. R(K) massivning musbat elementlari orasidan eng kichigini va manfiy elementlari orasidan eng kattasini toping.
7. Y(N) vektorning eng kichik elementidan keyingi barcha elementlarini nol bilan almashtiring.
8. A va B vektorlar berilgan. C vektorni shunday tashkil qiling, unung elementlari A da ham, B da ham mavjud bo`lsin.
9. Y(M) vektorning eng katta elementi va eng kichik elementlari o`rnini almashtirsh dasturini tuzing.
10. Y(K) vektorning uchga karrali elementlarining o`rta geometrigini hisoblang.
11. A(X) massivning elementlari yig`indisi 7ga karraligini aniqlang.
12. D(M) massivning juft elementlari sonini aniqlang.
13. Y(K) vektorning birinchi elementi bilan eng kichik elementi o`rnini almashtiring.
14. C(N) vektorning juft elementlaridan A va toq elementlaridan B vektorni hosil qiling 15. A(K) massivning juft elementlari indeksini aniqlang.
16. Berilgan N musbat butun son uchun A(N,N) matritsa hosil qiling, uning diagonal elementlari birdan, qolganlari noldan iborat bo`lsin.
17. Berilgan N musbat butun son uchun A(N,N) matritsa hosil qiling, uning diagonal elementlari satrlarining tartib raqamiga, qolgan elementlari noldan iborat bo`lsin.
18. A(N,M) matritsaning oxirgi satridan oldinda joylashgan satrini o`chirib yangi matritsa hosil qiling.
19. X(K,L) matritsaning eng katta va eng kichik elementlari o`rnini almashtiring.
20. A(N,N) matritsa elementlarini quyidagi tartibda tanlang: diagonal elementlari birdan, diagonaldan yuqoridagi elementlar noldan, diagonaldan pastdagi elementlar mos indekslarning yig`indisiga teng bo`lsin.
21. V(3,5) matritsa berilgan. B matritsaning eng kichik elementi joylashgan satr va ustunni yo`qotish yo`li bilan V matritsani hosil qiling.
22. A(M,N) martitsa berilgan. Matritsani (M+1) satr va (N+1) ustun bilan to`ldiring, bu satr va ustunning elementlari dastlabki matritsaning mos satr va ustunlarinig yig`indisidan tashkil topsin.
23. X(N,M) matritsani transponerlang.
24. A(3,4) matritsaning satr elementlari ko`paytmasidan B massivni hosil qiling.
25. Z(3,4) matritsaning har bir ustunidagi manfiy elementlar sonidan tashkil topgan M vektorni hosil qiling.
26. 1-ustun elementlari birdan, 2-ustun elementlari ikkidan va h.k., n-ustun elementlari ndan iborat bo`lgan Y(N,N) matritsani hosil qiling.
27. A(M,N) matritsa berilgan. Har bir satrdagi eng kichik elementlar orasidan eng kattasini va u joylashgan tartib raqamini aniqlang.
28. Diagonal elementlaridan tashqari barcha elementlari nolga teng bo`lgan C(M,M) matritsa tashkil eting.
29. K(3,4) matritsaning musbat elementlaridan tashkil topgan L vektorni hosil qiling.
30. K(M,M) matritsaning asosiy diagonal elementlaridan bir o`lchovli massiv hosil qiling va shu massiv elementlari yig`indisini hosil qiling.

31. Quyidagi qoida asosida Z(N,N) matritsani hosil qiling: diagonaldan yuqoridagi elementlar nolga teng, qolganlar esa ixtiyoriy qiymatlarini qabul qiladi.
32. A(N,M) matritsa elementlarini shunday ixtiyoriy butun sonlardan tanlang-ki, har bir satr va har bir ustundagi element oldingisidan kichik bo`lmasin.
33. X(M,M) matritsa diagonalidagi eng katta elementni toping va u joylashgan satrni chop eting.
34. C(5,5) matritsaning ikkita asosiy diagonal elementlari yig`indisini hisoblang.
35. Y(4,5) matritsaning juft elementlari o`rta arifmetigini hisoblang.

VARIANTLAR

1 - talaba savollari: 6; 25; 16; 34; 14; 12; 8; 29; 10; 31; 27; 24;
2 - talaba savollari: 15; 20; 23; 11; 4; 16; 10; 32; 8; 35; 17; 31;
3 - talaba savollari: 3; 20; 30; 33; 31; 27; 25; 9; 24; 14; 18; 6;
4 - talaba savollari: 18; 28; 14; 17; 11; 32; 25; 31; 22; 12; 7; 15;
5 - talaba savollari: 31; 12; 32; 30; 34; 15; 35; 29; 9; 26; 6; 14;
6 - talaba savollari: 21; 8; 33; 26; 32; 13; 11; 24; 25; 9; 4; 15;
7 - talaba savollari: 11; 26; 17; 5; 24; 20; 1; 21; 31; 35; 28; 19;
8 - talaba savollari: 23; 3; 14; 4; 7; 5; 28; 16; 34; 22; 21; 30;
9 - talaba savollari: 15; 7; 30; 27; 18; 3; 5; 34; 13; 21; 9; 10;
10 - talaba savollari: 11; 12; 15; 8; 7; 30; 23; 29; 34; 24; 32; 2;
11 - talaba savollari: 10; 2; 30; 32; 12; 27; 16; 26; 1; 5; 11; 24;
12 - talaba savollari: 31; 34; 2; 7; 24; 21; 29; 30; 9; 13; 11; 10;
13 - talaba savollari: 33; 25; 26; 8; 20; 2; 3; 5; 9; 34; 12; 17;
14 - talaba savollari: 24; 33; 10; 17; 5; 12; 27; 6; 26; 28; 8; 14; 15 - talaba savollari: 14; 5; 17; 26; 23; 29; 16; 20; 4; 7; 3; 18;

7-Laboratoriya mashg'uloti.
Mavzu: C++ tilida satriy kattaliklar.

Ishning maqsadi: Talabalarga C++ tilida satriy miqdorlar ustida amallar bajarishni o`rgatish.

Nazariya bo`yicha qisqacha ma'lumot

Kiritilgan satrni katta harflar bilan chiqaruvchi dastur tuzilsin

```
#include <iostream>  #include <ctype.h>  using namespace std;
int main()
{
   char c[20];       cout << "satr kiriting\n";       cin.getline(c, sizeof(c));       for (int i =
0; i < strlen(c); i++)      c[i] = toupper(c[i]);       cout << c << endl;
   return 0;
}
```

Toifalarni o'zgartirish funksiyalari

Quyidagi funksiyalardan foydalanish uchun **stdlib.h** sarlavha faylini progarmmaga qo'shish kerak.

Funksiya prototipi	Funksiya tavsifi
`double atof(const char *c)`	c satrini **double** toifasiga o'zgartiradi.
`int atoi(const char *c)`	c satrini **int** toifasiga o'zgartiradi.
`int atol(const char *c)`	c satrini **long int** toifasiga o'zgartiradi.
`double strtod(const char *c, char **endPtr)`	c satrini **double** toifasiga o'zgartiradi.
`char * itoa(int n, char *satr, int radix)`	n sonini radix sanoq sistemasida satr o'zgaruvchisiga o'zlashtiradi.

Satrni butun va haqiqiy songa aylantirish

```
#include <iostream>  #include <stdlib.h>  using namespace stds;
int main ()
{
  char c[] = "3.1415";
  double f;     int n;     f = atof(c);     n = atoi(c);     cout << f << endl;     cout << n << endl;
  return 0;
}
```

Satrlar bilan ishlovchi asosiy funksiyalar bilan tanishib chiqamiz.

Satr xususiyatlarini aniqlash uchun quyidagi funksiyalardan foydalanish mumkin:
unsigned int size() const; // satr o‘lchami unsigned int length() const; // satr elementlar soni unsigned int max_size() const; // satrning maksimal uzunligi unsigned int capacity() const; // satr egallagan xotira hajmi bool empty() const; // satrning bo‘shligini aniqlash
Satrning uzunligini aniqlash uchun length() yoki size() funksiyalaridan foydalanish mumkin.

```
#include <iostream>  #include <string>  using namespace std;
int main()
{      string s;
    cout << "Satr kiriting" << endl;
    getline(cin, s);
    cout << "Siz kiritgan satr " << s.length() << " ta belgidan iborat";       cout << "Siz kiritgan satr " << s.size() << " ta belgidan iborat";         return 0;
}
```

Satr qismini almashtirish

Satrning biror qismini almashtirish kerak bo'lsa, replace funksiyasidan foydalanish mumkin. replace (unsigned int pos1, unsigned int n1, const string &str);

replace (unsigned int pos1, unsigned int n1, const string & str, unsigned int pos2, unsigned int n2);

replace (unsigned int pos1, unsigned int n1, const char *str, int n);

replace funksiyasi insert kabi ishlaydi, faqat qo'shilishi kerak bo'lan satrni pos1 - o'rindan boshlab n1 ta belgi o'rniga qo'shadi.

2 ta satrni to'la almashtirish uchun swap funksiyasi ishlatiladi.

```
#include <iostream>  #include <string>
using namespace std;

int main()
{
   string s = "Assalomu alaykum do'stlar";
   string c = "Merhibon va muhtarama ayol";

   cout << s << endl;

   // 17 - belgidan boshlab 5 ta belgi o'rniga c satrni qo'shish      s.replace(17, 5, c);
   cout << s << endl;

   s.swap(c); // 2 ta satrni to'la almashtirish      cout << s << endl;

   s.replace(0, 0, c, 0, 17);
   s.erase(25);
   cout << s << endl;

   return 0;
}
```

Natija:

Assalomu alaykum do'stlar
Assalomu alaykum Merhibon va muhtarama ayollar
Merhibon va muhtarama ayol
Assalomu alaykum Merhibon

Satrdan qidirish funksiyalari

unsigned int find(const string &str, unsigned int pos=0) const;
Bu funksiyani chaqirgan satrning pos o'zgaruvchisida ko‘rsatilgan joyidan boshlab str satrni qidiradi.

Agar qidirilayotgan satr (str) topilsa, mos keluvchi satr qismining boshlanish indeksini javob sifatida qaytaradi, aks holda (satrning maksimal uzunligi qiymati) npos sonini qaytaradi. (npos=4294967295)
Agar pos ko'rsatilmasa, satr boshidan boshlab izlanadi.
unsigned int rfind(const string &str, unsigned int pos=npos) const;
Bu funksiyani chaqirgan satrdan pos o'ringacha str satri qidiriladi. Agar str topilsa, oxirgi uchragan indeks qaytariladi.
Agar pos ko'rsatilmasa, satr oxirigacha izlanadi. Ya'ni oxirgi uchragan indeks qaytariladi. Agar topilmasa, npos qaytariladi.

```
#include <iostream>  #include <string>  using namespace std;
int main()
{
   string s = "Assalomu alaykum";
   string c = "alaykum";      cout << "s=" << s << endl;      cout << "c=" << c << endl;
   cout << "s.find(c)=" << s.find(c) << endl; // 9
   cout << "c.find(s)=" << c.find(s) << endl; // 4294967295        // birinchi uchragan "a" harfining o'rnini aniqlash        cout << "s.find('a')=" << s.find("a") << endl; // 3 // oxirgi uchragan "a" harfining o'rnini aniqlash          cout << "s.rfind('a')=" << s.rfind("a") << endl; // 11      return 0;
}
```

Satr qismini almashtirishga misol

```
#include <iostream>
#include <stdlib.h>  #include <string>  using namespace std;
int main()
{ string c;      string s, s1;      size_t index;
   cout << "Satr kiriting" << endl;
   getline(cin, s);          cout << "Qidiriladigan satrni kiriting=";        getline(cin, c); cout << "Almashtiriladigan satrni kiriting="; getline(cin,s1);          index = s.find(c); while (index < s.length())
   {
      s.replace(index, c.length(), s1);
      index = s.find(c, index + s1.length());
   }
   cout << s << endl;      system("pause");
   return 0;
}
```

SAVOLLAR

1. S matn berilgan. Shu matnda ―b‖ harfi necha marta uchraydi?
2. n ta belgidan iborat bo_lgan S matn berilgan. Shu matnning nechanchi pozitsiyasidan boshlab ―a‖ belgisi ikki marta ketma-ket keladi? Agar kelmasa, natija deb nol olinsin.
3. Berilgan satrli ma‘lumot palindrom satrmi, yani o_ngdan va chapdan bir xil o_qiladimi?
4. Matnda mavjud ―,‖ larni o_chiring va ular sonini aniqlang.

5. Berilgan matnda ochildan qavslar va yopilgan qavslar nisbati mosligini tekshiring.
6. Gapdagi so_zlar sonini aniqlang.
7. Berilgan gapdagi ko_p nuqtali belgilarni tahrirlang.
8. Berilgan gapdagi har bir so_zda mavjud undosh sonini aniqlang.
9. Berilgan gapda unli ko_p bo_lgan so_zni aniqlang.
10. Matnda mavjud bo_sh joylarni o_chiring va ular sonini aniqlang.
11. n ta belgidan iborat bo_lgan S matn berilgan. Undagi barcha ―abcd‖ ko_rinishidagi belgilar guruhini o_chiring.
12. n ta belgidan iborat S matn berilgan bo_lsin. Bu matnga kirgan barcha raqamlar yig_indisi 3 ga bo_linadimi?
13. Ikki xonali son berilgan bo_lsin. Bu sonni so_zlar orqali ifodalang.

VARIANTLAR

1 - talaba savollari: 13; 2; 4; 12; 9; 6;
2 - talaba savollari: 7; 4; 5; 8; 13; 11;
3 - talaba savollari: 3; 13; 8; 4; 5; 2;
4 - talaba savollari: 7; 8; 2; 10; 4; 12;
5 - talaba savollari: 2; 11; 3; 12; 5; 4;
6 - talaba savollari: 4; 3; 11; 12; 10; 2;
7 - talaba savollari: 5; 9; 3; 7; 1; 11;
8 - talaba savollari: 4; 3; 1; 11; 8; 6;
9 - talaba savollari: 12; 8; 6; 10; 9; 1;
10 - talaba savollari: 2; 1; 7; 3; 9; 11;
11 - talaba savollari: 6; 2; 12; 5; 10; 11;
12 - talaba savollari: 2; 8; 1; 13; 12; 9;
13 - talaba savollari: 2; 13; 12; 10; 8; 4;
14 - talaba savollari: 4; 9; 10; 12; 11; 8;
15 - talaba savollari: 11; 7; 6; 13; 5; 1;

8-Laboratoriya mashg’uloti.
Mavzu: C++ tilida sinflar

Ishning maqsadi: Talabalarga C++ tilida sinflar yaratishni o`rgatish.

Nazariya bo`yicha qisqacha ma‘lumot

Sinflarni eng sodda holda quyidagicha tasvirlash mumkin:
Sinf-kaliti Sinf-soni {komponentalar ruyhati}
Sinf komponentalari sodda holda tiplangan ma'lumotlar va funksiyalardan iborat bo‘ladi. Figurali qavslarga olingan komponentalar ro‘yhati sinf tanasi deb ataladi. Sinfga tegishli funksiyalar komponenta-funksiyalar yoki sinf funksiyalari deb ataladi. Sinf kaliti sifatida Struct hizmatchi so‘zi ishlatilishi mumkin. Masalan quyidagi konstruktsiya kompleks son sinfini kiritadi.
Struct complex 1 { double real; double imag; void define (double re=0.0, double im=0.0)

{ real=re; imag=im;} void display (void)
{cout<=‖real=‖<<real; cout<=‖imag=‖<<imag; }
};
Strukturadan bu sinfning farqi shuki komponenta ma'lumotlardan (real, imag) tashqari ikkita komponenta funksiya (define() va display ()) kiritilgan. Bu kiritilgan sinf o‘zgaruvchilar tipi deb qaralishi mumkin. Bu tiplar yordamida konkret ob'ektlarni quyidagicha tasvirlash mumkin:
Misol uchun:
Complex x,y;
Complex dim[8];
Complex *p=1x;
Sinfga tegishli ob'ektlar quyidagicha tasvirlanadi;
 Sinf-nomi . ob'ekt-nomi
Dasturda ob'ekt komponentasiga quyidagicha murojaat qilish mumkin:
 Sinf-nomi.ob'ekt-nomi :: komponenta-nomi yoki soddaroq holda ob'ekt-nomi. Elementnomi
Misol uchun: x!=real=1.24; x!=imag=0.0; dim[3]. Real=0.25; dim[3]. Imag=0.0;
Sinfga tegishli funksiyalarga quyidagicha murojaat qilinadi: funksiya-nomi.ob'ekt-nomi; Misol uchun:
X. define.(Bu holda real=0.9 va imag=0.0)
X. define.(Bu holda kompleks son 4.3+i*20.0)
Display funksiyasi ekranda kompleks son qiymatlarini tasvirlaydi. Sinfga tegishli ob'ektga ko‘rsatkich orqali komponentalarga quyidagicha murojat qilinadi:
Ob'ektga-ko‘rsatkich>element-nomi
Yuqorida ko‘rsatilgan P ko‘rsatkich orqali H ob'ekt elementlariga quyidagicha qiymat berish mumkin:
P>real=2.3
P>imag=6.1
Huddi shu shaklda sinfga tegishli funksiyalarga murojat qilinadi:
P>display;
P>define(2.3, 5.4); Kompanenta o‘zgaruvchilar va kompanenta funksiyalar.
Sinf kompanenta o‘zgaruvchilari sifatida o‘zgaruvchilar , massivlar, ko‘rsatkichlar ishlatilishi mumkin . Elementlar ta'riflanganda initsializatsiya qilish mumkin emas . Buning sababi shuki sinf uchun hotiradan joy ajratilmaydi. Kompanenta elementlariga kompanenta funksiyalar orqali murojat qilinganda faqat nomlari ishlatiladi. Sinfdan tashqarida sinf elementlariga emas ob'ekt elementlariga murojaat qilish mumkin. Bu murojaat ikki hil bo‘lishi mumkindir.
Ob'ekt- nomi . Element - nomi.
Ob'ktga – korsatgich – element nomi.
Sinf elementlari sinfga tegishli funksiyalarida ishlatilishidan oldin ta'riflangan bo‘lishi shart emas. Huddi shunday bir funksiyadan hali ta'rifi berilmagan ikkinchi funksiyaga murojaat qilish mumkin. Komponentalarga murojaat huquqlari. Komponentalarga murojaat huquqi murojaat spetsifikatorlari yordamida boshqariladi. Bu spetsifikatorlar :
Protected – himoyalangan;
Private – hususiy;

Public – umumiy;
Himoyalangan kompanentalardan sinflar ierarhiyasi qurilganda foydalaniladi. Oddiy holda Protected spetsifikatori Private spetsifikatoriga ekvivalentdir. Umumiy ya'ni Public tipidagi komponentalarga dasturning ihtiyoriy joyida murojaat qilinishi mumkin. Hususiy ya'ni Private tipidagi komponentalarga sinf tashqarisidan murojaat qilish mumkin emas. Agar sinflar
Struct hizmatchi so‘zi bilan kiritilgan bo‘lsa, uning hamma komponentalari umumiy Public bo‘ladi, lekin bu huquqni murojaat spetsifikatorlari yordamida o‘zgartirish mumkin. Agar sinf
Class hizmatchi so‘zi orqali ta'riflangan bo‘lsa, uning hamma komponentalari hususiy bo‘ladi. Lekin bu huquqni murojaat spetsifikatorlari yordamida uzgartirish mumkindir. Bu spetsifikator yordamida Sinflar umumiy holda quyidagicha ta'riflanadi:

```
class class_name
{
  int data_member; // Ma'lumot-element
  void show_member(int); // Funksiya-element  };
```

Sinf ta'riflangandan so‘ng, shu sinf tipidagi o‘zgaruvchilarni(ob'ektlarni) quyidagicha ta'riflash mumkin:
class_name object_one, object_two, object_three; Quyidagi misolda employee, sinfi kiritilgandir:

```
class employee
{    public:
  char name[64] ;   long employee_id;
  float salary;
  void show_employee(void)
  {
   cout << "Imya: " << name << endl;
   cout << "Nomer slujathego: " << employee_id << endl;     cout << "Oklad: " << salary << endl;
  };
};
```

Bu sinf uch o‘zgaruvchi va bitta funksiya-elementga ega. Quyidagi EMPCLASS.CPP dastur ikki employee ob'ektini yaratadi. Nuqta operatordan foydalanib ma'lumot elementlarga qiymat beriladi so‘ngra show_employee elementidapn foydalanib hizmatchi haqidagi ma'lumot ekranga chiqariladi:

```
#include <iostream.h>  #include <string.h>
class employee
{  public:
  char name [64];   long employee_id;    float salary;
  void show_employee(void)
  {   cout << "Imya: " << name << endl;
   cout << "Nomer slujathego: " << employee_id << endl;     cout << "Oklad: " << salary << endl;
  }; };
```

```
void main(void)
{   employee worker, boss;   strcpy(worker.name, "John Doe");   worker.employee_id
= 12345;   worker.salary = 25000;   strcpy(boss.name, "Happy Jamsa");
  boss.employee_id = 101;   boss.salary = 101101.00;   worker.show_employee();
  boss.show_employee(); }
```

Sinf kompleks ob'ektlari uchun umumiy bo‘lgan elementlar statik elementlar deb ataladi. Yangi ob'ektlar yaratilganda statik elementlarga murojat qilish uchun oldin inizializatsiya qilinishi lozim. Inizializatsiya quyidagicha amalga oshiriladi:

Sinf-nomi:: kompleks-nomi initsializator

Misol uchun skladdagi tovarni kompleks tasvirlovchi sinfni kurib chiqamiz. Bu sinf komponentalari quyidagilardan iborat:

- Tovar nomi
- Olish narhi
- Kushimcha narh foiz ko‘rinishida
- Tovar haqida ma'lumotlar kiritish funksiyasi
- Tovar haqida ma'lumotlar va Tovar narhini chiqaruvchi funksiya;

Sinf ta'rifi :

Goods. Cpp

```
#include <iostream.h>
Struct goods
{ char name [40];
float price;  Static int percent; Void input()
{cout <<‖ Tovar nomi:‖; cin>>name;  cout<<‖ Tovar narhi:‖;cin>>price; }
```

Har bir yangi ob'ektning kompanentalari faqat shu ob'ektga tegishli bo‘ladi . Sinf kompanentasi yagona bo‘lib va hamma yaratilgan ob'ektlar uchun umumiy bo‘lishi uchun uni statik element sifatida ta'riflash ya'ni Static atributi orqali ta'riflash lozimdir . S sharning statik kompanentalarini inizializatsiya qilishdan so‘ng dasturda ob'ektlarni kiritmasdan oldin ishlatish mumkin. Agar kompanenta private yoki protected sifatida ta'riflangan bo‘lsa unga kompanenti funksiya yordamida murojat qilish mumkin. Lekin kompaneta funksiyaning chaqirish uchun biror ob'ekt nomini ko‘rsatish lozim. Lekin statik kompanentaga murojat qilinayotgan birorta ob'ekt yaratilmagan bo‘lishi yoki bir nechta ob'ektlar yaratilgan bo‘lishi mumkin shuning uchun sinf statik elemntlariga ob'ekt nomini ko‘rsatmasdan murojat qilish imkoniyatiga ega. Bunday imkoniyatni statik kompanenta funksiyalar yaratadi. Statik kompanenta funksiyaga ob'ekt nomi yoki obe'ktga ko‘rsatkich orqali murojaat qilish mumkin . Shu bilan birga nomi orqali qo‘ydagicha murojaat qilish mumkin.

Sinf - nomi : Statik – funksiya –nomi

Quyidagi dasturda point sinfi uch o‘lchovli fazodagi nuqtani aniqlaydi va shu bilan birga o‘z ichiga shu nuqtalar sonini oladi. # include <iostream. h > clearr point 3

```
{double x,y,z;  static int N; public
point 3 (double xu=0.o,double yu=0.o, double zu=0.o)
{N + +; x=xn; y=yn; z=zn; } static int & count ( ) {return N; }
};
int point 3: :N=0; void main (void)
{cout < <‖\ n size of ( point 3)=‖ < < size of (point 3) ; point 3 A (0: 0, 1. 0, 2. 0);
```

cout < < — \ nsize of (A)=‖< < size of (A)
point 3 B (3.0, 4.0, 5.0)
SAVOLLAR:

1. Sinflar nima maqsadda ishlab chiqiladi?
2. Bazaviy sinflar qanday e‘lon qilinadi?
3. Konstruktorlar va destruktorlar haqida ma‘lumot bering.
4. Konstruktorlarning nechta turi mavjud?
5. Static funksiya-elementlardan foydalanish.
6. Hosila sinflar qanday e‘lon qilinadi?
7. Quyidagilarga mos sinf yarating: bunda doiraning r — radiusi va silindrning h — balandligi o_zgaruvchilarining ichki qiymatlari yaratilayotgan obyektlar parametrlarini aniqlashi kerak. Circle bazaviy sinfi doirani modellashtiradi, Cylinder hosila sinfi esa silindrni modellashtiradi:
8. Polimorfizm haqida ma‘lumot bering.
9. Shablonlar qanday yaratiladi?

VARIANTLAR

1 - talaba savollari: 8; 6; 4; 1;
2 - talaba savollari: 1; 7; 9; 4;
3 - talaba savollari: 7; 2; 3; 4;
4 - talaba savollari: 9; 7; 5; 1;
5 - talaba savollari: 9; 7; 1; 6;
6 - talaba savollari: 2; 5; 7; 8;
7 - talaba savollari: 6; 3; 9; 5;
8 - talaba savollari: 7; 6; 2; 8;
9 - talaba savollari: 6; 7; 4; 3;
10 - talaba savollari: 9; 6; 4; 5;
11 - talaba savollari: 3; 9; 4; 5;
12 - talaba savollari: 8; 4; 7; 1;
13 - talaba savollari: 7; 9; 4; 3;
14 - talaba savollari: 5; 2; 8; 3;
15 - talaba savollari: 2; 4; 8; 5;

9-Laboratoriya mashg’uloti.
Mavzu: C++ tilida grafika, multimedia va animatsiyalar.

Ishning maqsadi: Talabalarga C++ tilida grafiklarni chizishni hamda multimedia va animatsiyalar yaratishni o`rgatish.
Nazariya bo`yicha qisqacha ma‘lumot

Ekran bilan ishlovchi funksiyalar. Quyidagi funksiyalar matnli rejimda ekran bilan ishlashga mo_ljallangan.
void clrscr(void) – ekranni tozalash
void gotoxy(int x, int y) – kursorni ko_rsatilgan nuqtaga ko_chirish void textcolor(int c) – text rangini o_rnatish void textbackground (int c) – text foni rangini o_rnatish Bu funksiyalar conio.h modulida joylashgandir.

Grafik rejimda ekran bilan ishlash
Grafik biblioteka. Dev C++ va Borland C++ kompilyatorlarida grafik biblioteka bilan bog_lanish uchun graphic.h – sarlavxali fayl qo_llaniladi. Bu bibliotekaga kiruvchi ba‘zi grafik funksiyalar:
void initgraph(int* graphdriver, int* graphmode, char* pathtodriver)- grafik rejimga o_tkazish void closegraph(void)-grafik rejimdan matnli rejimga o_tkazish.
void putpixel(int x, int y, int color) - Ekranda color rangli(x,y) kordinatali nuqtani tasvirlaydi.
void line (int x1, int y1, int x2, int y2) - Ekranda chiziq chizadi chizadi. void rectangle (int left, int top, int right, int bottom) - Ekranda to_rtburchak chizadi. void circle (int x, int y; int radius) - Ekranda aylana chizadi.
void ellipse (int x, int y, int stangle, int endangle, int xradius, int yradius) - Ekranda ellips
chizadi.
void outtextxy (int x, int y, char* textstring) – Textni berilgan pozitsiyada chiqaradi. void outtext (char* textstring) – Textni joriy pozitsiyada chiqaradi. int getbcolor(void) - Fon rangini qaytaradi int getcolor(void) - Tasvir rangini qaytaradi.
void getimage (int left, int top, int right, int bottom, void* bitmap) - ekran oynasini xotirada saqlash; putimage (int left, int top, void* bitmap, int op)- xotirada saqlangan tasvirni ekranga
joylash;
Misol. Animatsiyali aylanalar

```
#include<iostream.h>
#include<graphics.h> #include<cmath>
int main()
  {
    int a,b,i,k[900],l[900],j,m;
    cout<<"a="; cin>>a; cout<<"b="; cin>>b;
    cout<<"Nuqtalar soni="; cin>>m; srand(time(NULL));
for(i=1;i<=m;i++) { k[i]=rand()%(a/2)+a/4; l[i]=rand()%(b/2)+b/4; }
```

```
initwindow(a,b);  for(i=1;i<=m;i++)  for(j=1;j<=m;j++)  {
setcolor(i%15+1);
    circle(k[i],l[i],round(sqrt(pow(k[i]-k[j],2)+pow(l[i]-l[j],2))));  }
system("pause");  }
```

Misol. Ichma-ich joylashgan ellipslar

```
#include <graphics.h> main()
{
 int gd = DETECT, gm;  int x = 320, y = 240, radius;  initgraph(&gd, &gm,
"C:\\TC\\BGI");  for ( radius = 25; radius <= 125 ; radius = radius + 20)
  circle(x, y, radius);  getch();  closegraph();  return 0;
}
```

Misol. Sin, Cos, Tan, Ctg funksiyalari grafiklarini hosil qilish

```
#include <graphics.h>
#include <conio.h>
#include <math.h> #include <stdlib.h> using namespace std; int main()
{
      initwindow(800,600);  setbkcolor(WHITE);  cleardevice();
      setcolor(RED);
   line(0,300, getmaxx(), 300);
      line(400,0, 400, getmaxx());  int x,y;
float pi=3.1415;  setcolor(BLACK);
   outtextxy(10,320,"Sinus");
            setcolor(YELLOW);
   outtextxy(10,340,"Kosinus");
            setcolor(GREEN);
   outtextxy(10,360,"Tangens");
            setcolor(BLUE);
   outtextxy(10,380,"Kotangens");
 for (float a = -360; a <=360; a=a+0.01)
      {  setlinestyle(2,2,2);
      x=400+a;
            y=int(300-sin(a*pi/180)*100);
putpixel(x,y,BLACK);
y=int(300-cos(a*pi/180)*100);
      putpixel(x,y,YELLOW);
      y=int(300-tan(a*pi/180)*100);
      putpixel(x,y,GREEN);
y=int(300-1/tan(a*pi/180)*100);
                  putpixel(x,y,BLUE);
                  }
      getch();  closegraph();
      return 0;
}
```

Gorizontal yo`nalishda harakatlanuvchi uchburchak tasviri.

```
#include <graphics.h>
#include <conio.h>
#include <dos.h>
void Figure ( int x, int y, int color )
{   setcolor ( color );   line ( x, y, x+20, y );
line ( x, y, x+10, y-20 );   line ( x+10, y-20, x+20, y );
} int main()
{
  int d = VGA, m = VGAHI;
  int x, y, dx,key;
  initgraph ( &d, &m, "c:\\borlandc\\bgi" );

x = 0; y = 240; dx = 1; while ( 1 )
  {   if ( kbhit() )    if ( getch() == 27 ) break;
   Figure ( x, y, YELLOW );   delay ( 10 );
  Figure ( x, y, BLACK );  if ( x + 20 >= 639 ) dx = - 1;
  if ( x <= 0 )    dx = 1;
  x += dx;
  }   closegraph(); return 0;
}
```

Klaviaturaning Up, Down, Right va Left tugmalariga mos ravishda harakatlanuvchi aylana tasviri.

```
#include <graphics.h> #include <conio.h>
void Figure ( int x, int y, int color )
{   setcolor ( color );   circle(x,y,50);
} int main()
{ int d = VGA, m = VGAHI;
  int x, y, key;   initgraph ( &d, &m, "" );   setbkcolor(WHITE);
  cleardevice();    x = 320; y = 240;   while ( 1 )
   {
   Figure ( x, y, RED );    key = getch();    if ( key == 27 ) break;
 Figure ( x, y, WHITE );    switch ( key ) {     case 75: x --; break;
    case 77: x ++; break;     case 72: y --; break;
    case 80: y ++; break;
    }
   }
  closegraph(); return 0;
}
```

Ichma-ich hosil bo`luvchi aylanalar tasviri.

```
#include <graphics.h> #include <conio.h> using namespace std; int main()
{
        initwindow(400,400);     setbkcolor(WHITE);     a:cleardevice();     for (int i
= 1; i <=100; i=i+5)
```

```
                {
   setcolor(RED);      circle(200,200,i*2);
   delay(50);
  }    cleardevice();
  for (int i = 100; i >=1; i=i-5)
                {
   setcolor(RED);      circle(200,200,i*2);     delay(50);   }
  goto a;
        getch();       closegraph();
        return 0;
}
```

SAVOLLAR

1. Aylana tasvirini hosil qiling.
2. Ellips tasvirini hosil qiling.
3. To`rtburchak tasvirini hosil qiling.
4. Kvadrat tasvirini hosil qiling.
5. Futbol maydoni tasvirini hosil qiling.
6. Uy tasvirini hosil qiling.
7. Monitor tasvirini hosil qiling.
8. Ko`pburchak tasvirini hosil qiling.
9. Avtomobil tasvirini hosil qiling.
10. Robot tasvirini hosil qiling.
11. O`zbekiston Respublikasi davlat bayrog`i tasvirini hosil qiling.
12. Harflarni yozish usullaridan biri ularni kesmalarning birlashmasi oqrali ifodalashdir. Ekranda ana shu usul bilan ―TECT‖ so_zini hosil qiling.
13. Ekranning (320,240) koordinatali nuqtasida ―↑― ko_rinishidagi kursorni hosil qiling.
14. Yulduzli osmon va oy tasvirini hosil qiling.
15. Markazi ekran markazida joylashgan, radiusi 125 piksel bo_lgan doira tasvirini yasang va bo_yang.
16. Aylana va unga ichki chizilgan muntazam oltiburchak tasvirini yasang.
17. Ekranda doimiy tezlik bilan gorizontal yo_nalishda chapdan o_ngga va o_ngdan chapga qarab harakat qilayotgan nuqta tasvirini hosil qiling.
18. Ekranda aylana bo_ylab bir xil tezlikda harakatlanayotgan nuqta tasvirini yasang.
19. O_zining markaziy nuqtasi atrofida aylanayotgan muntazan uchburchakni yasang.
20. Strelkasi aylanib turuvchi soat tasvirini hosil qiling.
21. Klaviatura yordamida harakatlantirish mumkin bo`lgan kvadrat tasvirini hosil qiling.
22. Ekranda aylana bo_ylab harakatlanayotgan nuqta tasvirini yasang. U ―<‖ tugmasi bosilganda tezligini kamaytirsin, ―>‖ tugmasida esa tezlatsin.
23. Ekranda ichma-ich joylashgan ikki aylana bo_ylab qarama-qarshi yo_nalishda haralatlanayotgan ikkita nuqtani ifodalang. Ichki nuqtaning tezligi tashqi nuqta tezligidan kichik bo_lsin.

24. O‗zining markaziy nuqtasi atrofida aylanayotgan muntazan uchburchakni yasang.
25. Mayatnikning o‗zgarmas tezlik bilan tebranishini ifodalang.
26. Ekranda uzoqdan yaqinlasib kelayotgan shar tasvirini hosil qiling. Shar vaqt o‗tishi bilan kattalashishi qaysi qonun bilan aniqlanadi?

VARIANTLAR:

1 - talaba savollari: 10; 17; 22; 6; 23; 8; 13; 15; 24; 19;
2 - talaba savollari: 22; 24; 6; 12; 10; 25; 15; 9; 7; 23;
3 - talaba savollari: 1; 8; 12; 3; 26; 24; 17; 7; 25; 9;
4 - talaba savollari: 18; 20; 3; 12; 15; 8; 4; 11; 25; 23;
5 - talaba savollari: 9; 19; 17; 25; 6; 13; 24; 15; 23; 12;
6 - talaba savollari: 12; 2; 26; 16; 10; 25; 24; 15; 8; 5;
7 - talaba savollari: 13; 15; 5; 10; 12; 8; 7; 1; 11; 3;
8 - talaba savollari: 21; 23; 15; 17; 6; 26; 13; 18; 22; 14;
9 - talaba savollari: 17; 15; 6; 4; 26; 19; 21; 10; 22; 20;
10 - talaba savollari: 5; 19; 13; 12; 14; 22; 15; 25; 4; 10;
11 - talaba savollari: 15; 11; 19; 1; 24; 3; 14; 9; 23; 12;
12 - talaba savollari: 26; 2; 6; 17; 5; 3; 23; 1; 24; 22;
13 - talaba savollari: 7; 19; 4; 25; 5; 26; 2; 23; 17; 20;
14 - talaba savollari: 21; 6; 11; 26; 19; 15; 12; 2; 1; 23;
15 - talaba savollari: 14; 17; 15; 24; 21; 25; 23; 1; 13; 3;

10-Laboratoriya mashg`uloti.
Mavzu: C++ tilida fayllar bilan ishlash.
Ishning maqsadi: Talabalarga C++ tilida fayllar ustida amallar bajarishni o`rgatish.

Nazariya bo`yicha qisqacha ma‘lumot

Dastur ishlashi natijasida olingan ma`lumotlarni, saqlab qo`yish uchun, CD, DVD disklar, qattiq disklar va boshqa har xil tashqi qurilmalardan foydalaniladi. Ma`lumotlarni saqlab qo`yish uchun tashqi qurilmalardan foydalanish qulay va ishonchlidir.

Ma`lumotlarni saqlab qo`yish uchun, tashqi xotiraning nomlangan qismiga fayl deyiladi. Bunday fayllar fizik fayllar deyiladi. Mantiqiy fayllar. Fizik fayllar bilan ishlash uchun, dasturlash tillarida maxsus strukturalashgan, toifalangan fayllar kiritilgan. Bunday fayllar mantiqiy (logicheskiy) fayllar deyiladi. Mantiqiy fayllar, hech qanday fizik xotirani band qilmasdan ma`lumotlarning mantiqiy modelini o`zida saqlaydi.

Fizik va mantiqiy fayllar bir - biri bilan fopen funksiyasi orqali bog'lanadi. Fayl bir nechta elementdan tashkil topgan bo`lganligi uchun, faqat fayl ko`rsatkichi ko`rsatayotgan elementga murojaat qilish mumkin. Fayldan o'qish yoki yozish mumkin bo'lgan o'rinni ko'rsatuvhi elementga fayl ko'rsatkichi deyiladi. Fayldan ma'lumot o'qiganda yoki yozganda fayl ko'rsatkichi avtomat ravishda o'qilgan yoki

yozilgan bayt miqdoricha siljiydi. Fayl ko'rsatkichini magnitafon galovkasiga o'xshatish mumkin.
Binar fayl - har xil ob'ektlarni ifodalovchi baytlar ketma - ketligidir. Ob'ektlar faylda qanday ketma - ketlikda joylashganini programmaning o'zi aniqlashi lozim.

Fayllar bilan ishlovchi funksiyalardan foydalanish uchun <stdio.h> sarlavha faylini programmaga qo'shish kerak bo'ladi. Fayldan ma'lumotlarni o'qish yoki yozish uchun ochish fopen funksiyasi orqali amalga oshiriladi.
FILE * fopen (const char * filename, const char * mode); filename - o'zgaruvchisi char toifasidagi satr bo'lib, faylning to'liq nomini ko'rsatishi lozim (filename = "D:\Qudrat_c++\Namuna\file\file.txt"). Agar faylning faqat nomi ko'rsatilgan bo'lsa, fayl joriy katalogdan qidiriladi (filename = "file.txt"). mode - o'zgaruvchisi ham char toifasidagi satr bo'lib, faylni qaysi xolatda ochish lozimligini bildiradi.

mode qiymati	Faylning ochilish xolati
"w"	Faylni yozish uchun ochish. falename o'zgaruvchisida ko'rsatilgan fayl hosil qilinadi va unga ma'lumot yozish mumkin bo'ladi. Agar fayl oldindan bor bo'lsa (ya'ni oldin hosil qilingan bo'lsa), faylning ma'lumotlari o'chiriladi va yangi bo'sh fayl faqat yozish uchun ochiq holda bo'ladi.
"r"	Fayl o'qish uchun ochiladi. Agar fayl oldindan mavjud bo'lmasa, xatolik sodir bo'ladi. Ya'ni ochilishi lozim bo'lgan fayl oldindan hosil qilingan bo'lishi shart.
"a"	Faylga yangi ma'lumotlar qo'shish - kiritish uchun ochiladi. Yangi kiritilgan ma'lumotlar fayl oxiriga qo'shiladi. Agar fayl oldindan mavjud bo'lmasa, yangi fayl hosil qilinadi.
"w+"	Yozish va o'qish uchun faylni ochish. Agar fayl oldindan bor bo'lsa (ya'ni oldin hosil qilingan bo'lsa), faylning ma'lumotlari o'chiriladi va yangi bo'sh fayl yozish va o'qish uchun ochiqholda bo'ladi.
"r+"	Oldindan mavjud bo'lgan faylni o'qish va yozish uchun ochish.
"a+"	Fayl ma'lumotlarni o'qish va yangi ma'lumot qo'shish uchun ochiladi. fseek, rewind

Faylni ochishda xatolik sodir bo'lsa, fopen funksiyasi NULL qiymat qaytaradi.
Ochilgan faylni yopish uchun fclose funksiyasi ishlatilad i.
int fclose (FILE * stream);
Faylni yopishda xato sodir bo'lmasa, fclose funksiyasi nol qiymat qaytaradi. Xato sodir bo'lsa, EOF - fayl oxiri qaytariladi.

Faylga ma'lumot yozish va o'qish
size_t fread (void * ptr, size_t size, size_t n, FILE * stream);
fread funksiyasi, fayldan ptr ko'rsatkichi adresiga size xajmdagi ma'lumotdan n tani o'qishni amalga oshiradi. Agar o'qish muvoffaqiyatli amalga oshsa fread funksiyasi o'qilgan bloklar soni n ni qaytaradi. Aksholda nol qaytariladi
size_t fwrite (const void * ptr, size_t size, size_t n, FILE * stream); fwrite funksiyasi, faylga ptr ko'rsatkichi adresidan boshlab size xajmdagi ma'lumotdan n tani yozishni amalga oshiradi.

fread va fwrite funksiyalarining qo'llanilishi

```
#include <iostream>  #include <stdio.h>  using namespace std;
int main()
{
   int    n = 5;      double d = 10.77;      char s[20] = "dastur.uz";
   FILE *f;
   // binar faylni yozish uchun ochamiz          f = fopen("my_file.dat", "wb");
fwrite(&n, sizeof(int), 1, f); // n sonini faylga yozish      fwrite(&d, sizeof(double), 1,
f); // d sonini faylga yozish
   // satrni faylga yozish
   fwrite(s,  sizeof(char), strlen(s) + 1, f);      fclose(f); // faylni yopish      n = 0; d = 0;
   // binar faylni o'qish uchun ochamiz          f = fopen("my_file.dat", "rb");
fread(&n, sizeof(int), 1, f); // n sonini fayldan o'qish      fread(&d, sizeof(double), 1,
f); // d sonini fayldan o'qish
   // satrni fayldan o'qish
   fread(s,  sizeof(char), strlen(s) + 1, f);      fclose(f); // faylni yopish      cout << n <<
endl;      cout << d << endl;      cout << s << endl;
   return 0;
}
```

yuqoridagi misolda satrni yozish va o'qish uchun quyidagicha kod ishlatildi:

```
fwrite(s, sizeof(char), strlen(s) + 1, f);
fread (s, sizeof(char), strlen(s) + 1, f);
```

Buning kamchiligi s satridagi har bir belgi alohida - alohida faylga yozildi va o'qildi. Bu masalani quyidagicha hal qilish mumkin edi:

```
fwrite(s, sizeof(s), 1, f);   fread (s, sizeof(s), 1, f);
```

Lekin bu usulning ham kamchiligi bor. Ya'ni s satri belgilari soni massiv o'lchamidan kam bo'lgan holda, keraksiz ma'lumotlarni saqlash va o'qish sodir bo'ladi.

Fayl ko'rsatkichi bilan ishlovchi funksiyalar

Fayldan ma'lumot o'qiganda yoki yozganda fayl ko'rsatkichi avtomat ravishda o'qilgan yoki yozilgan bayt miqdoricha siljiydi. Fayl ko'rsatkichining kelgan joyini aniqlash uchun ftell funksiyasi ishlariladi. long int ftell (FILE * stream);

Fayl ko'rsatkichini siljitish uchun fseek funksiyasi ishlatilad i. int fseek (FILE * stream, long int offset, int whence);

Bu funksiya offset da ko'ratilgan bayt miqdoricha siljishni amalga oshiradi. whence o'zgaruvchisi quyidagi qiymatlarni qabul qilishi mumkin:

O'zgarmas	whence	Izoh
SEEK_SET	0	Fayl boshiga nisbatan siljitish
SEEK_CUR	1	Fayl ko'rsatkichining joriy xolatiga nisbatan siljitish
SEEK_END	2	Fayl oxiriga nisbatan siljitish

Agar whence = 1 bo'lsa (SEEK_CUR), offset musbat (o'ngga siljish) yoki manfiy (chapga siljish) bo'lishi mumkin.
Fayl ko'rsatkichini faylning boshiga o'rnatish uchun rewind funksiyasi ishlatiladi. void rewind (FILE * stream);
Bu amalni fayl ko'rsatkichini siljitish orqali ham amalga oshirish mumkin.

fseek (f, 0, SEEK_SET);
Agar faylda faqat butun sonlar yozilgan bo'lsa, uning k - elementiga murojaat quyidagicha bo'ladi.
fseek (f, sizeof(int) * (k - 1), SEEK_SET);
fread (&n, sizeof(int), 1, f);
Fayl oxirini aniqlash uchun feof funksiyasi ishlatiladi. int feof (FILE * stream); feof funksiyasi fayl ko'rsatkichi fayl oxirida bo'lsa, noldan farqli qiymat qaytaradi. Boshqa hollarda nol qaytaradi.

Misol. n natural soni berilgan. Elementlari n ta butun sondan iborat bo`lgan faylni hosil qiluvchi va ekranga chiqaruvchi programma tuzilsin.

```
#include <iostream>  #include <stdio.h>  using namespace std;
int main()
{     int n, k;      FILE *f;
   f = fopen("binar", "wb+");
   // binar faylni yozish va o'qish uchun ochish      if (f == NULL)
   {
      cout << "Faylni hosil qilishda xato bo'ldi";          return 1;
   }
   cout << "n="; cin >> n;      for (int i = 0; i < n; i++)
   {          cin >> k;
      fwrite(&k, sizeof(k), 1, f);
   }
   // fayl ko'rsatkichini fayl boshiga quyish      rewind(f);
   while (fread(&k, sizeof(k), 1, f))
   {
  //fayl boshidan fayl ko'rsatkichi turgan o'ringacha bo'lgan baytlar            int bayt =
ftell(f);
      cout << k <<" ftell(f)=" << bayt << endl;
   }      fclose(f);      return 0;
}
```

Misol. n natural soni berilgan. Elementlari n ta butun sondan iborat bo`lgan faylni hosil qiluvchi va juft elementlarini 2 marta orttiruvchi programma tuzilsin.

```
#include <iostream>  #include <stdio.h>  using namespace std;
int main()
{     int n, k;
   FILE *f;
```

```
   // binar faylni yozish va o'qish uchun ochish      f = fopen("binar", "wb+");      if (f
== NULL)
   {
      cout << "Faylni hosil qilishda xato bo'ldi";         return 1;
   }
   cout << "n="; cin >> n;      for (int i = 0; i < n; i++)
   {         cin >> k;
      fwrite(&k, sizeof(k), 1, f);
   }
   // fayl ko'rsatkichini fayl boshiga quyish      rewind(f);
   while (!feof(f)) // fayl oxiri uchramasa bajar
   {
      fread(&k, sizeof(k), 1, f);         if (k % 2 == 0 )
      {
         k *= 2;
         // fayl ko'rsatkichini sizeof(int) bayt chapga surish                fseek(f, -
sizeof(int), SEEK_CUR);                fwrite(&k, sizeof(int), 1, f);                // fayl
ko'rsatkichini o'rnatish
         fseek(f, ftell(f), SEEK_SET);
      }
   }
   cout << "fayl elementlari\n";
   rewind(f);
   while (fread(&k, sizeof(k), 1, f))      cout << k << endl;      fclose(f);      return 0;
}
```

Matnli faylga ma'lumot yozish

```
#include <iostream>   include <fstream>   using namespace std;  int main ()  {
   ofstream yozish; // faylga yozish oqimini hosil qilish
yozish.open("namuna.txt");
   // yangi namuna.txt nomli fayl hosil qilinadi.
   // agar namuna.txt fayli oldindan bo'lsa,
   // uning eski qiymatlari o'chiriladi      // va yangi fayl hosil qilinadi
   yozish << "Matnli faylga ma'lumot yozish" << endl;      yozish << "Juda oson!" <<
endl;      yozish.close(); // faylni yopish
   return 0;
}
```

Matnli fayldan o'qish

```
#include <iostream>
#include <fstream>  #include <string>  using namespace std;  int main ()  {
   ifstream oqish; // fayldan o'qish oqimini hosil qilish      string satr;
   oqish.open("namuna.txt");          // faylni ochishda xatolik sodir bo'lsa          if
(!oqish.is_open())
   {
```

```
    cout << "Faylni ochishda xatolik sodir bo'ldi." << endl;
    exit(1); // dasturni tugatish
  }
  while (!oqish.eof())
  {
    // fayldan o'qish  getline(oqish, satr);  // ekranga chiqarish  cout
<< satr << endl;  }
  // namuna.txt fayli bilan oqish oqimi aloqasini uzish
  oqish.close();
  return 0;  }
```

Fayldan nusxa olish

```
//ushubu dastur orqali ixtiyoriy fayldan nusxa olish mumkin
#include <iostream>  #include <fstream>  using namespace std;
int main ()
{
  int length;
  char * buffer, fayl[] = "matn.txt", yangi[]="yangi_fayl.txt";
 // fayl - nusxalanadigan fayl nomi
  // yangi - yangi nusxalangan fayl nomi
  // o'qish oqimi
  ifstream fromfile(fayl, ios::binary );  if (!fromfile.is_open())
  {
    cout << "faylni o'qishda xatolik sodir bo'ldi\n";  exit(1);
  }
  // yozish oqimi  ofstream tofile(yangi, ios::binary );  // fayl xajmini aniqlash:
fromfile.seekg (0, ios::end); // fayl oxiriga o'tish  length = fromfile.tellg();
fromfile.seekg (0, ios::beg); // fayl boshiga o'tish  // xotira ajratish:
  buffer = new char [length];  // blokka ma'lumotlarni o'qish:  fromfile.read
(buffer, length);  fromfile.close();
 // nusxalanishi kerak bo'lgan faylga yozish
  tofile.write (buffer, length);  // xotirani bo'shatish  delete[] buffer;
  cout << "fayl nusxalandi\n";
  return 0;
}
```

SAVOLLAR

1. Hafta kunlarining nomlarini kiritib, ularni «HAFTA.TXT» faylida saqlab qo'yadigan dastur tuzing
2. «HAFTA.TXT» faylida berilgan hafta kunlarining nomlarini ekranga chiqaruvchi dastur tuzing
3. y=sin2x funksiyasining [-π;π] oraliqdagi qiymatlarini 0,01 qadam bilan hisoblang. Natijalarni «sinus.out» faylida saqlab quying.
4. «sinf.txt» faylida berilgan 9-sinf o'quvchilarning familiyalari ichidan «M» harfi bilan boshlanadiganlarini ekranga chiqaruvchi dastur tuzing.

5. «sinf.txt» faylida berilgan 9-sinf o'quvchilarining familiyalari ichidan «B» harfi bilan boshlanadiganlarini ajratib olib, ulardan «bsinf.txt» faylini hosil qiluvchi dastur tuzing.
6. «massiv.in» fayli 12 ta satrdan iborat. Uning har bir satrida 9 tadan son o'zaro probel bilan ajratib yozilgan. A(12;9) - ikki o'lchamli massiv elementlarining qiymatlarini «massiv.in» faylidan o'qib oluvchi dastur tuzing.
7. y=cos2x funksiyasining [-π;π] oraliqdagi qiymatlarini 0,01 qadam bilan hisoblang. Natijalarni «sinus.out» faylida saqlab quying.
8. y=cosx funksiyasining [-π;π] oraliqdagi qiymatlarini 0,01 qadam bilan hisoblang. Natijalarni «sinus.out» faylida saqlab quying.
9. y=sinx funksiyasining [-π;π] oraliqdagi qiymatlarini 0,01 qadam bilan hisoblang. Natijalarni «sinus.out» faylida saqlab quying.
10. y=tg2x funksiyasining [-π;π] oraliqdagi qiymatlarini 0,01 qadam bilan hisoblang. Natijalarni

«sinus.out» faylida saqlab quying.

VARIANTLAR:

1 - talaba savollari: 5; 7; 3; 9; 1;
2 - talaba savollari: 8; 2; 10; 4; 9;
3 - talaba savollari: 4; 2; 9; 3; 6;
4 - talaba savollari: 1; 7; 5; 2; 4;
5 - talaba savollari: 7; 2; 9; 1; 10;
6 - talaba savollari: 4; 3; 7; 1; 2;
7 - talaba savollari: 2; 5; 7; 10; 3;
8 - talaba savollari: 5; 10; 7; 9; 1;
9 - talaba savollari: 8; 5; 7; 1; 4;
10 - talaba savollari: 1; 9; 2; 6; 10;
11 - talaba savollari: 4; 6; 3; 7; 2;
12 - talaba savollari: 2; 4; 1; 9; 8;
13 - talaba savollari: 7; 6; 2; 5; 1;
14 - talaba savollari: 9; 5; 6; 10; 8;
15 - talaba savollari: 9; 7; 3; 10; 5;

Asosiy va qo‘shimcha o‘quv adabiyotlar hamda axborot manbaalari

Asosiy adabiyotlar

1. Peter Gottschling. Discovering Modern C++. An Intensive Course for Scientists, Engineers, and Programmers. “Addison-Wesley”, 2015 y.

2. M.Ashurov, N.Mirzahmedova, N.Xaytullayeva. Algoritmlash va dasturlash asoslari. Uslubiy qo‘llanma. T. : “Bayoz”, 2016 y.

3. A. R. Azamatov, B. Boltayev. Algoritmlash va dasturlash asoslari. O‘quv qo‘llanma. T. : “Cho‘lpon”, 2010 y.

4. A. R. Azamatov, B. Boltayev. Algoritmlash va dasturlash asoslari. O‘quv qo‘llanma. T. : “Cho‘lpon”, 2013 y.

5. Sh. I. Razzoqov, M. J. Yunusova. Dasturlash: Kasb-hunar kollejlari uchun o‘quv qo‘llanma. T. : “Ilim Ziyo”, 2011y.

6. М. Ашуров, М. Мирмахмудов, Ш. Сапаев. Замонавий дастурлаш тиллари фанидан лаборатория ишлари. Т. : ТДПУ, 2008 й.

7. Меняев Михаил Федорвич. Информационные технология управления. Москва, «Издательский Омегал», 2003 г.

Qo‘shimcha adabiyotlar

1. Мирзиёев Шавкат Миромонович. Эркин ва фаровон, демократик Ўзбекистон давлатини биргаликда барпо этамиз. Ўзбекистон Республикаси Президенти лавозимига киришиш тантанали маросимига бағишланган Олий Мажлис палаталарининг қўшма мажлисидаги нутқ / Ш.М. Мирзиёев. – Тошкент : Ўзбекистон, 2016. - 56 б.

2. Мирзиёев Шавкат Миромонович. Танқидий таҳлил, қатъий тартиб-интизом ва шахсий жавобгарлик – ҳар бир раҳбар фаолиятининг кундалик қоидаси бўлиши керак. Мамлакатимизни 2016 йилда ижтимоий-иқтисодий ривожлантиришнинг асосий якунлари ва 2017 йилга мўлжалланган иқтисодий дастурнинг энг муҳим устувор йўналишларига бағишланган Вазирлар Маҳкамасининг кенгайтирилган мажлисидаги маъруза, 2017 йил 14 январ / Ш.М. Мирзиёев. – Тошкент : Ўзбекистон, 2017. – 104 б.

3. Мирзиёев Шавкат Миромонович**.** Қонун устуворлиги ва инсон манфаатларини таъминлаш – юрт тараққиёти ва халқ фаровонлигининг гарови. Ўзбекистон Республикаси Конституцияси қабул қилинганининг 24 йиллигига бағишланган тантанали маросимдаги маъруза. 2016 йил 7 декабр /Ш.М.Мирзиёев. – Тошкент: “Ўзбекистон”, 2017. – 48 б.

4. Мирзиёев Шавкат Миромонович. Буюк келажагимизни мард ва олижаноб халқимиз билан бирга қурамиз. Мазкур китобдан Ўзбекистон Республикаси Президенти Шавкат Мирзиёевнинг 2016 йил 1 ноябрдан 24 ноябрга қадар Қорақалпоғистон Республикаси, вилоятлар ва Тошкент шаҳри сайловчилари вакиллари билан ўтказилган сайловолди учрашувларида сўзлаган нутқлари ўрин олган. /Ш.М.Мирзиёев. – Тошкент: : “Ўзбекистон”, 2017. – 488 б

5. Ўзбекистон Республикаси Президентининг Фармони. Ўзбекистон республикасини янада ривожлантириш бўйича ҳаракатлар стратегияси тўғрисида. *(Ўзбекистон Республикаси қонун ҳужжатлари тўплами, 2017 й., 6-сон, 70-модда)*

6. Ўзбекистон Республикаси Конституцияси. Т.: Ўзбекистон. 2014. -46 б.

7. П. Дарахвелидзе, Э. Марков. Программирование в Delphi7. Учебник. Санкт-Петербург, “БХВ-Петербург” 2003 г.

8. В. М. Пестиков, А. Н. Маслобоев. Turbo PASCAL 7. 0. Изучаем на примерах. Санкт-Петербург. : “БХВ-Петербург”, 2004 г.

9. Фаронов В. В. Программирование на языке высокого уровня Delphi. Учебник. М. : “Питер”, 2003 г.

10. В.Т.Безручко. Практикум по курсу информатики. М. : «Финансы и статистика», 2004 г.

11. Nazirov Sh.A., Qobulov R.V., Bobojanov M.R., Raxmanov Q.S. C va C++ tili. “Voris-nashriyot” MCHJ, Toshkent 2013. 488 b.

12. П. Дарахвелидзе, Э. Марков. Программирование в Delphi7. Учебник. Санкт-Петербург, “БХВ-Петербург” 2003 г.

13. Фаронов В. В. Программирование на языке высокого уровня Delphi. Учебник. М. : “Питер”, 2003 г.

14. В. Т. Безручко. Практикум по курсу информатики. М. : «Финансы и статистика», 2004 г.

Internet saytlari

1. **www.ziyonet.uz** – Axborot ta’lim portali
2. **www.edu.uz** – Oliy va o‘rta maxsus ta’lim vazirligi portali
3. www.tdpu.uz **– Nizomiy nomidagi TDPU rasmiy sayti**
4. http://acm.tuit.uz/ **- dasturiy yechim to‘g‘riligini avtomatik testlovchi tizim.**
5. http://acm.tuit.uz/forum/**, http://acm.timus.ru/ – dasturlarni testlovchi tizim.**

G L O S S A R I Y

Atamaning ingliz tilida nomlanishi	Atamaning o’zbek tilida nomlanishi	Atamaning rus tilida nomlanishi	Atamaning nomlanishi
Algorithm	Algoritm	Алгоритм	1. Vazifani bajarishga qaratilgan aniq belgilangan qoidalarning tartiblangan chekli to`plami. 2. Dastlabki ma’lumotlarni oxirgi natijaga o`tkazuvchi hisoblash jarayoni orqali masala yechimini aniq ko`rsatuvchi amallar mazmuni va ketma-ketligi.
Allocation	Joylashtirish	Размещение	1. Biror narsani joylashtirish, ishlatish tartibi. 2. Informatikada- sahifalarning o`lchovlarini va matnlarning sahifasini, tasvir tuzulmasini belgilaydi. 3. Dasturlashda - ma’lumotlarni kiritish-chiqarish tartibi va ketma-ketligi. 4. Apparat ta’minotini loyihalashda-platalar, integral sxemalar va tarkibiy qismlarni joylashtirish. 5. Tashqi xotirada faylni yozish uchun makon ajratish.
Animation	Animatsiya	Анимация	Bir necha tasvir yoki kadrlarni ko`rsatish orqali yaratiladigan harakat taqlidi. Televideniyedagi multfilmlar animatsiyaning bir turidir.
Application	qo‘llanma	Приложение	Ma’lum foydalanish sohasida ma’lumotlarga ishlov berishni amalga oshiruvchi jami dasturlar.
Architecture	Arxitektura	Архитектура	Murakkab obyektning tuzilishi, bajarilayotgan vazifalari va tarkibiy bo`laklarining o`zaro bog`liqligini belgilovchi konsepsiya. Tarmoq me’moriy tuzilmasi uning asosiy elekmentlari va ularning o`zaro ishlash tavsifi va topologiyasini belgilaydi.
Archive	Arxiv	Архив	Arxivator yordamida ochish mumkin bo`lgan, tarkibida bir yoki ko`p (odatda kompressiyalangan) fayllar va axborot bo`lgan fayl. Arxivlar odatda dasturiy mahsulotlar yoki rezerv nusxalarni tarqatish uchun yaratiladi. tar, gzip formatidagi arxivlar UNIX; zip, rar, arj formatidagi arxivlar esa Windows amaliy tizimlarida ishlatiladi
archive document	arxiv hujjati	архивный документ	1. Axborot tashuvchisi turidan qat’iy nazar davlat va jamiyat uchun ahamiyatliligi sababli saqlanayotgan yoki saqlanishi lozim bo`lgan, hamda mulkdori uchun tarixiy, ilmiy, badiiy, madaniy qiymatga ega bo`lgan hujjat.
Archiver	Arxivator	Архиватор	Tashqi qurilmada ixcham va uzoq muddatli saqlash uchun fayllarni zichlash (arxivlash) va zichlangan fayllarni dastlabki shaklga qaytarish (arxivsizlash) uchun mo`ljallangan dastur yoki dasturlar majmui. Shaxsiy kompyuterlarda eng keng tarqalgan axrivatorlar – PKZIP, ARJ, RAR

array processor	matritsaviy protsessor	матричный процессор	Sonli massivlarni, masalan matritsalarni qayta ishlash uchun mo`ljallangan arxitekturaga ega bo`lgan markaziy protsessorning ham protsessori.
Attribute	Atribut	Атрибут	Xususiyat, sifat yoki miqdor belgisi. U makondagi obyektni ta'riflovchi (biroq uning qayerda joylashganligini ko`rsatish bilan bog`liq bo`lmagan) va uning noyob soni ya'ni aniqlovchisi bilan bog`liqlikda tasavvur qilinadi.
Attack	Hujum	Атака	Kompyuter muhofazasini buzishga qaratilgan harakat.
backup copy	zahira nusxa	резервная копия	Ma'lumotlar ko`chirilgan nusxasini o`z ichiga olgan magnit disk yoki tasma.
backup procedure	zahiraviy nusxalash	резервное копирование	Kompyuter disklari, ma'lumotlar bazalari, veb-serverlari mazmunidan davriy ravishda to`la yoki qisman nusxa ko`chirish.
beta testing	beta testlash	бета-тестирование	Dasturiy mahsulotni bozorga chiqarishdan avval sinash uchun ishlatib ko`rish.
bibliographic description	bibliografik ta'rif	библиографическое описание	Hujjat haqidagi bibliografik ma'lumotlar majmui.
blended portal	aralash portal	смешанный портал	O`zida elektron savdo vazifalari va an'anaviy ma'lumotnoma xizmatlarini mujassamlantirgan portal.
Brightness	Ravshanlik	Яркость	Kompyuter grafikasida rang tavsiflanadigan uch tavsifnomadan (to`yinganlik va ta'sirchanlik bilan bir qatorda) biri.
Brush	mo'yqalam	Кисть	Tasvirlarni chizishda va bo`yashda, aniq o`lcham, rang va fakturadagi yo`llarni o`tkazish texnologiyasi.
catalogue	Katalog	Каталог	1. Izlab topish qulayligini hisobga olib tartibga solingan obyektlar ro`yxati. 2. Informatikada, bir xil turdagi obyektlar orasidan qidirishni ta'minlaydigan ma'lumotlarning tuzilmasini aniqlovchi ma'lumotnoma
Cell	Uya	Ячейка	1. Jadvaliy qo`llanmalarda – ma'lumotlar elementini (matn, son qiymati, formula) kiritish uchun mo`ljallangan to`g`riburchak shaklli katak.
Character	Ramz	Знак	Biror bir tushunchani, hodisani, jarayonni shartli ifodalashda xizmat qiluvchi alomat.
chief editor	bosh muharrir	главный редактор	Tahririyatni (qanday atalishidan qat'iy nazar) boshqaradigan va ommaviy axborot vositasini ishlab chiqarish va nashr qilish bo`yicha yakuniy qarorni qabul qiladigan shaxs.

Coding	Kodlash	Кодирование	1. Dastlabki alifboni obyektli alifboga o`zgartirish jarayoni. 2. Ma'lumotlarni ramzlar ketma-ketligi bilan ifodalash jarayoni
Colour	Rang	Цвет	Muayyan elektromagnit spektrli yorug`likni ko`z bilan sezish. Kompyuter grafikasida rang uch tavsifnoma bilan tavsiflanadi: - ta'sirchanlik, yorug`lik nuri chastotasi bilan belgilanadigan sifat; - to`yinganlik, rangni berilgan ta'sirchanlik bilan ifodalanish darajasi, odatda foizlarda belgilanadi (0 dan 100 gacha); - ravshanlik, nurlanish energiyasi darajasi (yorug`lik oqimining zichligi).,
colour print	rangli choplash	цветная печать	Matn va grafikani rangli choplash imkoniyati.
Compression	Taxlam	Упаковка	Tashuvchi imkoniyatlariga ko`ra kattaroq ma'lumotlar hajmlarini uzatish (yoki xotirlash) imkonini beruvchi signallarni kodlash/dekodlash uslubi.
Computer	Kompyuter	Компьютер	Hisoblarni bajarish, shu jumladan elektron shakldagi axborotni oldindan belgilangan algoritm bo`yicha qabul qilish, qayta ishlash, saqlash va ishlov berish uchun mo`ljallangan mashina.
computer architecture	kompyuter arxitekturasi	архитектура компьютера	Kompyuter tarkibiy bo`laklarining texnik va dasturiy vositalarining o`zaro aloqalarini o`z ichiga oluvchi kompyuterning mantiqiy tuzilishi va funksional tavsifnomalari.
computer graphics	kompyuter grafikasi	компьютерная графика	Kompyuterlar yordamida tasvirlarni yaratish va ishlov berish texnologiyasi.
computer language	kompyuter tili	компьютерный язык	Kompyuterlar va kompyuter texnikasi bilan bog`liq, odatda tillarga tegishli tushuncha.
computer literacy	kompyuter savodxonligi	компьютерная грамотность	Shaxsiy kompyuterda ishlash uchun zarur bilim va ko`nikmalarning eng kam to`plamini egallash.
computer program	kompyuter dasturi	компьютерная программа	1. Masalani yechish algoritmining tavsifi. Dasturlash tilida beriladigan, dasturchi tomonidan tuziladigan va kompyuter bajaradigan ko`rsatmalar yig`masi.
computer protection	kompyuter muhofazasi	защита компьютера	Ma'lumotlar va tizim resurslarini, odatda tasodifiy va qasddan qilingan harakatlarga qarshi qo`llanadigan tegishli tadbirlar tizimi bilan

			muhofazalash.
computer virus	kompyuter virusi	компьютерный вирус	1. Boshqa dasturlarni turlab o`z-o`zini tarqatadigan dastur. U iloji boricha, o`z o`zgartirilgan nusxalarini ham va kasallangan dasturni chaqirilganda bajariladigan dasturlarni ham o`z ichiga oladi.
Data	ma’lumotlar	Данные	1. Rasmiylashtirilgan, ya’ni uzatish, izohlash va qayta ishlash uchun mos shaklda taqdim etilgan axborot. 2. Kompyuterda qayta ishlanishi jarayonida aylanayotgan hujjatlashtirilgan axborot.
data model	ma’lumotlar modeli	модель данных	Ma’lumotlarni saqlash, uzatish va qayta ishlash sohalarida tarkibiy qismlar turi va ularning aloqalari to`g`risidagi tasavvur.
data security	ma’lumotlarning xavfsizligi	безопасность данных	Dasturlarni va ma’lumotlarni tasodifiy yoki qasddan o`zgartirish, yo`q qilish, oshkor qilish, hamda ruxsatsiz foydalanishdan muhofazalash tamoyillar to`plami.
data search	ma’lumotlar izlash	поиск данных	Axborot massivida oldindan belgilangan izlash sharti (so`rovi) talabini qondiruvchi yozuvlar borligini aniqlash jarayoni va agar ular mavjud bo`lsa bunday yozuvlar joylashishini aniqlash jarayoni.
Disk	Disk	Диск	Bitta yoki ikkita tomonida ma’lumotlarni o`qish yoki yozishni amalga oshirish uchun aylanuvchi yassi dumaloq plastinadan iborat ma’lumotlar tashuvchisi.
document processing	hujjatga ishlov berish	обработка документов	Hujjatlarni yaratish va o`zgartirish jarayoni. Hujjatlarga ishlov berish tasniflash, saralash, zarur bo`lgan shaklga o`zgartirish, ma’lumotlar bazasida joylashtirish, izlash va foydalanuvchilarga berishdan iborat.
File	Fayl	Файл	Yagona yaxlit deb qaraladigan ma’lumotlar yoki dasturlar majmuasi. Fayl o`z nomiga ega bo`lgan va tizimda saqlanadigan ma’lumotlarning asosiy elementi bo`lgan obyektdir
file name extension	fayl ismi kengaytmasi	расширение имени файла	Nuqtadan keyin joylashadigan, fayl ismining bir qismi. Masalan, “def.exe” fayl ismidagi “exe” qismi kengaytma bo`lib hisoblanadi.
Font	Shrift	Шрифт	Alifbo ramzlarining to`plam shakli. Shrift garnitura (imlo elementlari)ning birlashmasi, shakl, o`lchamlar, interval bilan ajralib turadi.

O‘ZBEKISTON RESPUBLIKASI

OLIY VA O‘RTA MAXSUS TA’LIM VAZIRLIGI

TOSHKENT DAVLAT PEDAGOGIKA UNIVERSITETI

"TASDIQLAYMAN"

SH.SHaripov.

2019 yil "29" 06

"KELISHILDI"

Oliy va o‘rta maxsus ta’lim vazirligi

2019 yil "17" 10

Ro‘yhatga olindi: BD -5110700-2.05

2019 - yil "17" 08

DASTURLASH TILLARI

FAN DASTURI

Bilim sohasi:	100000	– Gumanitar
Ta’lim sohasi:	110000	– Pedagogika
Ta’lim yo‘nalishi:	5110700	– Informatika o‘qitish metodikasi

TOSHKENT – 2019

O'zbekiston Respublikasi Oliy va o'rta maxsus ta'lim vazirligining 2019-yi
"4" 10 dagi 892-sonli buyrug'i bilan ma'qullangan fan dasturlarini tayanch
oliy ta'lim muassasasi tomonidan tasdiqlashga rozilik berilgan.

Fan dasturi Oliy va o'rta maxsus, kasb-hunar ta'limi yo'nalishlari bo'yicha
O'quv-uslubiy birlashmalar faoliyatini Muvofiqlashtiruvchi Kengashning 2019-yi
"17" 08 dagi 4 -sonli bayonnomasi bilan ma'qullangan.

Fan dasturi Nizomiy nomidagi Toshkent Davlat pedagogika universitetid
ishlab chiqildi.

Tuzuvchilar:

M.O'.Ashurov	– "Informatika va uni o'qitish metodikasi" kafedrasi katta o'qituvchisi
N.S.Xaytullayeva	– "Informatika va uni o'qitish metodikasi" kafedrasi katta o'qituvchisi, pedagogika fanlari bo'yicha falsafa doktori (PhD).

Taqrizchilar:

Ziyadullayev D.SH.	– Muhammad al-Xorazmiy nomidagi Toshkent axborot texnologiyalari universiteti «Axborot ta'lim texnologiyalari» kafedrasi mudiri, texnika fanlari nomzodi
Yuldasheva U.T.	– TTESI qoshidagi akademik litsey direktori, texnika fanlari nomzodi

Fan dasturi Nizomiy nomidagi Toshkent davlat pedagogika universiteti O'q
-uslubiy Kengashida ko'rib chiqilgan va tavsiya qilingan (2019- yil "29
06 dagi 11. -sonli bayonnoma).

I. O‘quv fanining dolzarbligi va oliy kasbiy ta’limdagi o‘rni

Mustaqil Respublikamizda yuz berayotgan siyosiy, iqtisodiy, ilmiy-texnikaviy va madaniy o‘zgarishlar Oliy ta’lim tizimida ham o‘z aksini topmoqda. O‘zbekistonda uzluksiz ta’lim-tarbiya tizimini yaratish, shu asosida ta’lim sifatini jaxon andozalari darajasiga etkazish ta’lim sistemasining eng dolzarb vazifasiga aylandi. Bu esa barcha mutaxassisliklar qatori Informatika va dasturlash bo‘yicha kadrlar tayyorlash sifatini oshirishni ham taqozo etadi. Bu maqsad vazifalar ushbu fan dasturi mazmunini ham belgilaydi. Algoritm konsepsiyasining vujudga kelishi bilan algebra, sonlar nazariyasi, geometriya va matematikaning boshqa sohalariga tegishli bir qator muammolarning echimli yoki echimli emasligini aniqlashtirish imkonini berdi. Algoritmlar nazariyasi faoliyat sohasi EHMlar vujudga kelishi bilan yanada kengaydi. Yuqoridagi fikrlar “Dasturlash tillari” fanining asosiy mazmunini belgilashga yordam beradi.

“Dasturlash tillari” fani umumkasbiy fanlar blokiga kiritilgan kurs hisoblanib, 2- va 3-kurslarda o‘qitilishi maqsadga muvofiq. “Dasturlash tillari” fani “Informatika o‘qitish metodikasi” ta’lim yo‘nalishida o‘qitiladi. Mazkur fan Algoritmlar fanining nazariy va uslubiy asosini tashkil qilib, o‘z rivojida aniq va tabiiy fanlar uchun zamin bo‘lib xizmat qiladi.

II. O‘quv fanining maqsadi va vazifasi

“Dasturlash tillari” fanini o‘qitishdan maqsad – talabalarga dasturlashning ilmiy-nazariy asoslarini, informatika o‘qituvchisining kasbiy sohasida egallashi lozim bo‘lgan bilimlar, amalda qo‘llash uchun ko‘nikma va makalalarni shakllantirish hamda rivojlantirisbdan iborat.

Ushbu maqsadga erishish uchun fan talabalarni ob’ektga yo‘naltirilgan dasturlash tillarida ishlash, amaliy masalalarga dasturlar tuzishga oid nazariy bilimlar, amaliy ko‘nikma va malakalarini shakllantirish vazifalarini bajaradi.

Fan bo‘yicha talabalarning bilim, ko‘nikma va malakalariga quyidagi talablar qo‘yiladi. ***Talaba:***

- ob’yektga yo‘naltirilgan dasturlash tillarining nazariy asoslari, ob’yektlarni loyihalash, matematik va interfeys ob’yektlari, voqealar va xabarlar, ob’yektga yo‘naltirilgan muhitlarda xabarlarni uzatish, ularga ishlov berish mexanizmlari, ob’yektlar iyerarxiyasi asosida dasturlarni loyihalash, muayyan ob’yektga yo‘naltirilgan dasturlash tillari to‘g‘risida ***tasavvurga ega bo‘lishi;***
- ob’yektga yo‘naltirilgan dasturlash tillarida chiziqli, tarmoqlanuvchi va takrorlanuvchi va modulli dasturlar tuza olishni, dasturlashning ob’yektga yo‘naltirilgan paradigmasini, ob’yektga yo‘naltirilgan muhitlarda dasturlarni loyihalashni ***bilishi va ulardan foydalana olishi;***

I. O‘quv fanining dolzarbligi va oliy kasbiy ta’limdagi o‘rni

Mustaqil Respublikamizda yuz berayotgan siyosiy, iqtisodiy, ilmiy-texnikaviy va madaniy o‘zgarishlar Oliy ta’lim tizimida ham o‘z aksini topmoqda. O‘zbekistonda uzluksiz ta’lim-tarbiya tizimini yaratish, shu asosida ta’lim sifatini jaxon andozalari darajasiga etkazish ta’lim sistemasining eng dolzarb vazifasiga aylandi. Bu esa barcha mutaxassisliklar qatori Informatika va dasturlash bo‘yicha kadrlar tayyorlash sifatini oshirishni ham taqozo etadi. Bu maqsad vazifalar ushbu fan dasturi mazmunini ham belgilaydi. Algoritm konsepsiyasining vujudga kelishi bilan algebra, sonlar nazariyasi, geometriya va matematikaning boshqa sohalariga tegishli bir qator muammolarning echimli yoki echimli emasligini aniqlashtirish imkonini berdi. Algoritmlar nazariyasi faoliyat sohasi EHMlar vujudga kelishi bilan yanada kengaydi. Yuqoridagi fikrlar “Dasturlash tillari” fanining asosiy mazmunini belgilashga yordam beradi.

“Dasturlash tillari” fani umumkasbiy fanlar blokiga kiritilgan kurs hisoblanib, 2- va 3-kurslarda o‘qitilishi maqsadga muvofiq. “Dasturlash tillari” fani “Informatika o‘qitish metodikasi” ta’lim yo‘nalishida o‘qitiladi. Mazkur fan Algoritmlar fanining nazariy va uslubiy asosini tashkil qilib, o‘z rivojida aniq va tabiiy fanlar uchun zamin bo‘lib xizmat qiladi.

II. O‘quv fanining maqsadi va vazifasi

“Dasturlash tillari” fanini o‘qitishdan maqsad – talabalarga dasturlashning ilmiy-nazariy asoslarini, informatika o‘qituvchisining kasbiy sohasida egallashi lozim bo‘lgan bilimlar, amalda qo‘llash uchun ko‘nikma va malakalarni shakllantirish hamda rivojlantirishdan iborat.

Ushbu maqsadga erishish uchun fan talabalarni ob’ektga yo‘naltirilgan dasturlash tillarida ishlash, amaliy masalalarga dasturlar tuzishga oid nazariy bilimlar, amaliy ko‘nikma va malakalarini shakllantirish vazifalarini bajaradi.

Fan bo‘yicha talabalarning bilim, ko‘nikma va malakalariga quyidagi talablar qo‘yiladi. ***Talaba:***

- ob‘yektga yo‘naltirilgan dasturlash tillarining nazariy asoslari, ob‘yektlarni loyihalash, matematik va interfeys ob‘yektlari, voqealar va xabarlar, ob‘yektga yo‘naltirilgan muhitlarda xabarlarni uzatish, ularga ishlov berish mexanizmlari, ob‘yektlar iyerarxiyasi asosida dasturlarni loyihalash, muayyan ob‘yektga yo‘naltirilgan dasturlash tillari to‘g‘risida ***tasavvurga ega bo‘lishi;***
- ob‘yektga yo‘naltirilgan dasturlash tillarida chiziqli, tarmoqlanuvchi va takrorlanuvchi va modulli dasturlar tuza olishni, dasturlashning ob‘yektga yo‘naltirilgan paradigmasini, ob‘yektga yo‘naltirilgan muhitlarda dasturlarni loyihalashni ***bilishi va ulardan foydalana olishi;***

– ob'yektga yo'naltirilgan dasturlash tillari muhitida ishlash, masalalarni tahlil qila olish, muayyan dasturlash tillari yordamida masalalarning dasturini tuzish va natijalarni taqqoslay olish ***ko'nikmalariga ega bo'lishi lozim.***

III. Asosiy nazariy qism (ma'ruza mashg'ulotlari)

Fanning nazariy mashg'ulotlari mazmuni

1-MODUL. OB'EKTGA YO'NALTIRILGAN DASTURLASH TILLARI

1-mavzu. "Dasturlash tillari" faniga kirish

Dasturlash tillari va ularning klassifikasiyasi. Mashinaga mo'ljallangan va proseduraga mo'ljallangan dasturlash tillari. Yuqori darjali dasturlash tillari. Interpretatorlar va kompilyatorlar. Dasturlarni translyasiyalash. Muyyan dasturlash tilining alifbosi, buruqlar tizimi va operatorlari.

2-mavzu. Ob'ektga yo'naltirilgan dasturlash tillari

Ob'ektga yo'naltirilgan dasturlash tillari. Dasturlashning ob'ektga yo'naltirilgan paradigmasi.

3-mavzu. Ob'ektga yo'naltirilgan loyihalash.

Ob'ektlarni loyihalash: satrlar, steklar, ro'yxatlar, navbatlar, daraxtlar. Matematik ob'ektlar: rasional va kompleks sonlar, vektorlar, matrisalar. Ob'ektlar kutubxonasi. Interfeys ob'ektlari: boshqarish elementlari, oynalar, dialoglar.

4-mavzu: Voqealar va habarlar.

Voqealar va habarlar. Ob'ektga yo'naltirilgan muhitlarda habarlarni uzatish va ularga ishlov berish mexanizmlari. Ob'ektlar ierarxiyasi asosida dasturlarni loyihalash. Muayyan ob'ektga yo'naltirilgan dasturlash tili va unda dastur tuzish asoslari

2-MODUL. DELPHI DASTURLASH TILI

5-mavzu. Delphi dasturlash tili ishchi muhiti

Delphi dasturlash tilining ishchi muhiti, undagi oynalar (Ob'ektlarning daraxtsimon ko'rinish oynasi, ob'ektlar inspektori oynasi, kod brauzeri oynasi, asosiy oyna, forma oynasi, dastur kodi oynasi), u o'rnatilishi zarur bo'lgan kompyuterga qo'yiladigan texnik talablar va instrumental tugmalar. Komponentlar palitrasi. Palitra bo'limlari va ayrim komponentlar xossalari bilan tanishish.

6-mavzu. Standard komponentlar palitrasi

Frame komponenti va uning xossalari. MainMenu, PopupMenu komponentlari. Label, Edit, Button, Memo, Panel komponentlari va ularning xossalari. CheckBox, ListBox, ComboBox, ListBox, RadioGroup, RadioButton, ScrollBar komponentlari.

7-mavzu. Additional komponentlar palitrasi

BitBtn, MaskEdit, StringGrid, MaskEdit, CheckListBox, DrawGrid, Image, Shape va boshqa komponentlaridan foydalanish. Komponentlar xossalalri.

3-MODUL. DELPHIDA DASTURLASH

8-mavzu. Delphi dasturlari strukturasi. Loyiha va modul.

Bo'sh forma va uning modifikasiyasi. Delphida nomlanishlar, forma xossalarini o'zgartirish. Formaga yangi komponent joylashtirish va unda komponent xossalaridan foydalanish, Xodisa tushunchasi. Loyiha va modul strukturasi. Dastur elementlari (alfavit, identifikatorlar, doimiyliklar, ifodalar va amallar).

9-mavzu. Delphida tiplar, o'zgarmaslar, o'zgaruvchilar va standart funksiyalar.

Delphida tiplar, ularning ahamiyati. Butun tiplar: sodda (tartib va xaqiqiy) tiplar, mantiqiy va simvolli tiplar, tip-diapazon, vaqt-sana tipi. Delphida simvolli va satriy tiplar. Simvolli va satriy tiplarning berilishi, ular bilan bajariladigan amallar. Simvolli va satriy kattaliklar,

Delphi dasturlash tilida o'zgarmaslar, o'zgaruvchilar va standart funksiyalar

4-MODUL. DELPHI DASTURLASH TILI OPERATORLARI.

10-mavzu. Delphi dasturlash muxitida tarmoq operatorlari.

Tarkibiy va bo'sh operatorlar. Shart va mantiqiy ifodalar. If…Then…else shartli operatori. Tanlash (case) operatori. Goto o'tish operatori. Label (belgilar) xizmatchi so'zidan foydalanish qoidalari bilan tanishish.

11-mavzu. Delphi dasturlash muxitida siklik operatorlar.

Delphi dasturlash tilida sikllar. For sikli. While sikli. Repeat sikli. Murakkab sikllar.

12-mavzu. Delphida massivlar.

Delphi dasturlash tilida massivlar. Massivlarni tavsiflash, e'lon qilish. Massivlarni berilish usullari. Massiv elementlarini kiritish va chiqarish. Random (max) funksiyasi bilan tanishtirish. Ko'p o'lchovli massivlar. Massiv elementlarini saralash va tartiblash.

13-mavzu. Prosedura va funksiyalar

Prosedura va funksiyalar Delphi dasturlash tilining muhim instrumenti sifatida. Prosedura tarifi, uning nomi, undan foydalanish yo'llari. Funksiya tarifi, uning nomlanishi, undan dasturda foydalanish va uning proseduradan farqi.

14-mavzu. Delphi dasturlash tilining grafik vositalari.

Delphi dasturlash tilining grafik imkoniyatlari. Delphidagi maxsus TCanvas, TFont, TPen, Tbrush klasslari. TFont klassi xossalari: Color, Name, Size, Style. TPen klassi xossalari: Color, Mode, Width, Style. TBrush klassi xossalari: Bitmap, Color, Style.

5-MODUL. C++ DASTURLASH TILIGA KIRISH

15-mavzu. C++ tilining leksik asoslari

C++ dasturlash tiliga kirish. C++ dasturlash tili alifbosi va xizmatchi so'zlari. Amallar. Izohlar satrini tavsiflash. C++ tilida operatorlar. Standart funksiyalar va ularning yozilishi. Konsol orqali muloqot qilish. Chiqarish operatori. Kiritish operatori

16-mavzu. O'zgaruvchi va o'zgarmas tipli kattaliklar.

O'zgaruvchilar. Identifikatorlar. Ma'lumotlar tipi. O'zgaruvchilarni e'lon qilish. O'zgaruvchilarni initsializatsiya qilish.

17-mavzu. Dasturlash operatorlari

C++ dasturlash tilidagi operatsiyalar. Arifmetik operatorlar. Qiymatni bir birlikka o'zgartiruvchi operatorlar. Taqqoslash operatorlari.

18-mavzu. Shartli operatorlar

C++ dasturlash tilida o'tish operatori. Shartli operatorning qisqa ko'rinishi. Shartli operatorning uzun ko'rinishi. Tanlash operatorlari. Shartsiz o'tish operatori. Ko'p tarmoqlanishlar va variant tanlash operatorlari.

19-mavzu. C++ dasturlash tilida takrorlanuvchi jarayonlar

Sikl operatorlari. Oldingi shartli While takrorlash operatori. Keyingi shartli do-while takrorlash operatori. For takrorlash operatori. Uzish break operatori. Umumiy takrorlanish algoritmlari va ichma-ich takrorlanishlar. Continue operatori.

20-mavzu. C++ dasturlash tilida funksiyalar

Funsiyalar haqida tushuncha va ularni yaratish. Funksiyalarning tuzilishi. Funksiya parametrlari. Lokal va global o'zgaruvchilar. Tipsiz funksiyalar. Void funksiyasi. Funksiyalardan foydalanish. Qiymat qaytarmaydigan funksiyalar va ular yordamida masala yechish.

21-mavzu. C++ dasturlash tilida massivlar

Massivlar haqida tushuncha. Massivlarni tavsiflash va ulardan foydalanish. Bir o'lchovli massivlar. Ko'p o'lchovli (indeksli) massivlar. Massivlarni navlarga ajratish usullari.

2-mavzu. C++ da ko`rsatkichlar

Adres (manzil) operatori. Jo'natish operatori. Ko'rsatkich tipidagi o'zgaruvchilarni e'lon qilish. Ko'rsatkichga boshlang'ich qiymat berish. Ko'rsatkich ustida amallar. Adresni olish amali. Ko'rsatkichlar va adres oluvchi o'zgaruvchilar funksiya parametri sifatida. Ko'rsatkichlar va massivlar.

23-mavzu. C++ da satrlar va ular ustida amallar

Satr uzunligini aniqlash funktsiyalari. Satrlarni nusbalash, ulash, solishtirish. Satrdagi harflar registrini almashtirish. Satrda izlash funktsiyalari. Turlarni o'zgartirish funktsiyalari.

24-mavzu. C++ da strukturalar va birlashmalar

Strukturalar. Ma'lumot strukturalari. Struktura ko'rsatkichlari. Strukturalar bilan ko'rsatgich a'zolar. Birlashmalar va ular ustida amallar. Foydalanuvchi tomonidan aniqlangan berilganlar turi. Sinflar.

25-mavzu. C++ da fayllar bilan ishlash

Matn fayllarini o'qish va yozish. Oqimni ochish. Faylдan o'qish. Faylga yozish. Binary fayllar bilan ishlash operatorlari. Matn va binar fayllar. O'qish-yozish oqimlari. Standart oqimlar. Belgilarni o'qish-yozish funksiyalari. Satrlarni o'qish - yozish funksiyalari. Faylдan o'qish-yozish funksiyalari. Formatli o'qish va yozish funksiyalari. Fayl ko'rsatkichini boshqarish funksiyalari.

6-MODUL. VIZUAL DASTURLAR TUZISH

26-mavzu. Borland C++ Builder dasturlash muhiti

Borland C++ Builder dasturlash muhitiga kirish, ishchi muhit, oynalar. C++ Bulder komponentlari va ularning hossalari. Komponentlar hodisalari va metodlari. Komponentlar tarkibi. Hodisalar. Uslublar. Loyihalar menejeri. C++ Builder da ilova dastur yaratish. Oddiy ilova dasturini yararish.

27-mavzu. Borland C++ Builderda tiplar

Borland C++ Builder da butun va haqiqiy sonlar. Tiplarni almashtirish. Borland C++ Builderda simvolli va satriy tiplar. Simvolli va satriy tiplarning berilishi, ular bilan bajariladigan amallar.

28-mavzu. Borland C++ Builderda amallar, matematik funktsiyalar va taqlash operatorlari

Matematik funktsiyalar va doimiyliklardan foydalanish. Amallar va ularning bajarilish tartiblari. Mantiqiy amallar. If va Switch operatorlari.

29-mavzu. Borland C++ Builderda sikllar

For sikli va uning qo'llanilishi. While va do_while sikllari. Ichma-ich joylashgan sikllar.

30-mavzu. Borland C++ Builder komponentlarini o'rganish

Borland C++ Builder komponentlar palitrasi va komponentlar xossalari. Guruhli operatsiyalar uchun komponentlarni tanlash. Komponentlar o'lchovlarini

o'zgartirish. Matn muharriri ilova dasturini yaratish. Hodisa jarayonlariini yaratish. Menyu yaratish. Panel va menyu yaratuvchi komponentlar: Panel, GroupBox, Bevel, ScroolBox, ToolBar, StatusBar.

31-mavzu. Borland C++ Builder Standard komponentlar palitrasi

Frame komponenti va uning xossalari. MainMenu, PopupMenu komponentlari. Label, Edit, Button, Memo, Panel komponentlari va ularning xossalari, CheckBox, ListBox, ComboBox, ListBox, RadioGroup, RadioButton, ScrollBar komponentlari.

32-mavzu. Borland C++ Builder Additional komponentlar palitrasi

BitBtn, MaskEdit, StringGrid, MaskEdit, CheckListBox, DrawGrid, Image, Shape va boshqa komponentlaridan foydalanish. Komponentlar xossalalri.

33-mavzu. Borland C++ Builder Dialogs komponentlar palitrasi

Borland C++ Builderdagi Dialogs komponentlar palitrasi. OpenDialog, SaveDialog, FindDialog, ColorDialog, FontDialog va hokazo komponentlar va ularning xossalari.

34-mavzu. Borland C++ Builderda komponentlar xodisalari va metodlari

Komponentlar xodisalari: OnClick, OnDblClick, OnKeyDown, OnKeyPress, OnKeyUp, OnEnter, OnExit, OnMouseDown va OnMouseUp, OnChange va hokazo.

Komponentlar metodlari: Add, Hide, Show, Delete, CanFocus, ChangeScale, TextOut, MoveTo, LineTo va hokazo.

35-mavzu. Borland C++ Builderda massivlar.

Borland C++ Builderda massivlarni tavsiflash, e'lon qilish. Borland C++ Builderda massiv elementlarini kiritish va chiqarish. Borland C++ Builderda massiv elementlarini saralash va tartiblash.

36-mavzu. Borland C++ Builderda grafik tasvirlar yaratish

Tayyor grafik fayllardan foydalanish. Grafik fayllar formati. Image Editor grafik muharriri. Tugmalar uchun piktogrammalar yaratish.

37-mavzu. Borland C++ Builderning grafik vositalari

Borland C++ Builderning maxsus TCanvas, TFont, TPen, Tbrush klasslari. TFont klassi xossalari: Color, Name, Size, Style. TPen klassi xossalari: Color, Mode, Width, Style. TBrush klassi xossalari: Bitmap, Color, Style.

38-mavzu. Borland C++ Builderda multimedia va animatsiyalar.

Borland C++ Builderda multimedia va animatsiyalar. C++ tilida ko'p formali loyihalar yaratish.

IV. Amaliy mashg'ulotlar bo'yicha ko'rsatma va tavsiyalar

Amaliy mashg'ulotlarda talabalar muayyan masala bo'yicha mavjud bo'lgan yoki mustaqil tarzda kichik ishchi guruhlari yordamida hosil qilingan algoritmlarni

muhokama qiladilar. Mazkur mavzularga oid test masalalar tuzib, ular asosida tuzilgan dasturlar majmuasini tuzadilar va kompyuterda olingan natijalarni birgalikda tahlil qiladilar.

Amaliy mashg'ulotlar uchun quyidagi mavzular tavsiya etiladi:

1. Komponent xosslarini dinamik va statik o'zgartirish.
2. Standard va Additional bo'limi komponentlaridan foydalanish yo'llari
3. Delphidagi dasturlarda tiplardan foydalanish.
4. Delphi dasturlash tilida tarkibiy operatorlar va tanlash operatori.
5. Delphi dasturlash tilida sikl operatorlari.
6. Delphi dasturlash tilida massivlar va satriy kattaliklar.
7. Delphi dasturlash tilida protsedura va funksiyalar
8. Delphida modullar va ulardan foydalanish
9. Delphi dasturlash tilining Office dasturlari bilan hamkorligi
10. Delphi dasturlash muxitida fayllar bilan ishlash. Mul'timedia ilovalari
11. Delphida MBni boshqaradigan ilovalar tuzish
12. Delphi ning grafik komponentlari.
13. C++ da ma`lumotlarning asosiy turlari bilan amallar bajarish.
14. C++ tilida chiziqli dasturlash
15. C++ tilida Shartli va shartsiz o'tish operatorlari. Tanlash operatori
16. C++ tilida takrorlanish operatorlari (while, do while, for)
17. C++ tilida massivlar
18. C++ tilida funksiyalar yaratish
19. C++ tilida strukturalar va birlashmalar
20. C++ tilida ko'rsatkichlar
21. C++ tilida sinflar
22. C++ tilida multimedia va animatsiyalar
23. Borland C++ Builderda sikllar bilan ishlash
24. Borland C++ Builderda massivlar bilan ishlash
25. Borland C++ Builderda funksiya va protseduralar bilan ishlash
26. Borland C++ Builderda fayllar bilan ishlash
27. Borland C++ Builderda ko'p formali ilovalar yaratish

Amaliy mashg'ulotlarni tashkil etish bo'yicha kafedra professor-o'qituvchilari tomonidan ko'rsatma va tavsiyalar ishlab chiqiladi. Unda talabalar asosiy ma'ruza mavzulari bo'yicha olgan bilim va ko'nikmalarini amaliy masalalarga dasturlar tuzish orqali bilimlarini yanada boyitadilar. Shuningdek, darslik va o'quv qo'llanmalar asosida talabalar bilimlarini mustahkamlashga erishish, tarqatma materiallardan foydalanish, ilmiy maqolalar va tezislarni chop etish orqali talabalar bilimini oshirish, masalalarning dasturini tuzish, mavzular bo'yicha ko'rgazmali qurollar tayyorlash va boshqalar tavsiya etiladi.

Amaliy mashg'ulotlar multimedia qurulmalari bilam jihozlangan auditoriyada bir akadem guruhga bir o'qituvchi tomonidan o'tkazilishi lozim. Mashg'ulotlar faol va interfaktiv usullar yordamida o'tilishi, mos ravishda munosib pedagogik va axborot texnologiyalar qo'llanilishi maqsadga muvofiq.

V. Laboratoriya mashg'ulotlari bo'yicha ko'rsatma va tavsiyalar

Laboratoriya mashg'ulotlarida talabalar amaliy mashg'ulotlarda tuzilgan dasturlarni kompyuter yordamida natijalalarini ko'rib, ularni taxlil qiladilar va xulosalar chiqaradilar.

Laboratoriya mashg'ulotlari uchun quyidagi mavzular tavsiya etiladi:

1. Delphidagi dasturning interfeys qismini yaratish.
2. Delphida muloqot dasturi yaratish
3. Delphi dasturlash muxitida chiziqli dasturlar tuzish
4. Delphi dasturlash muxitida tarmoqlanuvchi dasturlar tuzish
5. Delphi dasturlash muxitida For siklik operatoridan foydalanish
6. Delphi dasturlash muxitida While siklik operatoridan foydalanish
7. Delphi dasturlash muxitida Repeat siklik operatoridan foydalanish
8. Delphida bir o'lchovli massivlarga doir dastur tuzish
9. Delphida ikki o'lchovli massivlarga doir dastur tuzish
10. Delphi dasturlash tilida to'plamlar bilan ishlash
11. Delphida simvolli kattaliklar bilan ishlash
12. Delphida satriy kattaliklar bilan ishlash
13. Delphi dasturlash muhitida funksiyalardan foydalanish
14. Delphi dasturlash muhitida proseduralardan foydalanish
15. Delphida standart modullar va ulardan foydalanish
16. Delphida modul yaratish va undan foydalanish
17. Delphi dasturlash tilining Word dasturi bilan hamkorligi
18. Delphi dasturlash tilining Excel dasturi bilan hamkorligi
19. Delphi dasturlash tilining Access dasturi bilan hamkorligi
20. Delphi dasturlash tilida grafik elementlar bilan ishlash
21. Delphi dasturlash muxitida fayllar bilan ishlash
22. Delphida multimediyali ilovalar yaratish
23. C++ dasturlash tilida ma'lumotlarning asosiy turlari bilan amallar bajarish
24. C++ dasturlash tilida qiymat berish operatorlari bilan ishlash
25. C++ dasturlash tilida arifmetik va mantiqiy amallar bilan ishlash
26. C++ dasturlash tilida chiziqli dasturlar tuzish
27. C++ dasturlash tilida If/Else strukturasi bilan ishlash
28. C++ dasturlash tilida Switch strukturasi bilan ishlash
29. C++ dasturlash tilida While siklik operatoridan foydalanish
30. C++ dasturlash tilida Do_While siklik operatoridan foydalanish
31. C++ dasturlash muxitida For takrorlash strukturasi bilan ishlash
32. Boshqaruv strukturalarida Continue va Break ifodalarini qo'llash
33. C++ dasturlash tilida funktsiyalar yaratish va ulardan foydalanish
34. C++ da bir o'lchovli massivlar bilan ishlash
35. C++ da ikki o'lchovli massivlar bilan ishlash
36. C++ dasturlash tilida ko'rsatkichlar bilan ishlash
37. C++ da satriy kattaliklar bilan ishlash
38. C++ dasturlash tilida strukturalar bilan ishlash
39. C++ dasturlash tilida strukturada ko'rsatkichlardan foydalanish
40. C++ dasturlash tilida matnli fayllar bilan ishlash

41. C++ dasturlash tilida funktsiyalarni qayta yuklash
42. Borland C++ Builderda Label, Edit va Button kompenentlaridan foydalanib dastur tuzish
43. Borland C++ Builderda CheckBox, RadioGroup, ComboBox va ListBox kompenentlaridan foydalanib dastur tuzish
44. C++ tilida fayllar bilan ishlash. Dialog oynalari
45. C++ tilida panel va menyu yaratuvchi komponentlar
46. Borland C++ Builderda massivlarga doir dastur tuzish
47. Borland C++ Builderda funksiya va protseduralarga doir dastur tuzish
48. Borland C++ Builderda satriy kattaliklar
49. Borland C++ Builderda fayllar bilan ishlash.
50. Borland C++ Builderda grafikaga doir dastur tuzish
51. Borland C++ Builderda multimedia va animatsiyalarga doir dastur tuzish

VI. Mustaqil ta'lim va mustaqil ishlar

Mustaqil ta'lim uchun tavsiya etiladigan mavzular:

1. Yuqori darjali dasturlash tillari.
2. Interpretatorlar va kompilyatorlar.
3. Ob'ektga yo'naltirilgan dasturlash tillari.
4. Ob'ektga yo'naltirilgan loyihalash.
5. Ob'ektlar ierarxiyasi asosida dasturlarni loyihalash.
6. Delphi qo'llanaladigan matematik funksiyalar
7. TForm komponentlari va ularning xossalari
8. Delphi dasturlash tilida sodda dasturlarni tuzish
9. Delphi dasturlash tilida shartli va siklli dasturlar tuzish
10. Delphi dasturlash tilida tasodifiy sonlar bilan ishlash
11. Delphi dasturlash tilida massivlar bilan ishlash
12. Delphi dasturlash tilida sana-vaqt turi bilan ishlash
13. Delphi dasturlash tilida satriy kattaliklar bilan ishlash
14. Delphi dasturlash tilida to'plamlar bilan ishlash
15. Delphi dasturlash tilida fayllar bilan ishlash
16. Delphi dasturlash tilida funksiya va proseduralarni yaratish
17. Delphi dasturlash tilida grafik primitivlar bilan ishlash
18. Delphi dasturlash tilida ListBox da grafiklarni joylashtirish
19. Biror predmet sohasiga oid o'rgatuvchi dasturlar yaratish
20. Delphi dasturlash tilida nazorat qiluvchi dasturlar yaratish
21. Delphi dasturlash tili komponentlar palitrasi bo'limlari va ayrim komponentlar xossalari.
22. Delphida forma xossalari va ularni o'zgartirish.
23. Delphi dasturlash tilida grafika.
24. Delphi dasturlash tilining multimedia imkoniyatlari.
25. C++ tilining boshqarish operatorlari
26. C++ tilida funksiyalar, strukturalar va birlashmalar.
27. C++ tilida ko'rsatkichlar va murojaatlar

28.C++ tilida bir o'lchovli, ikki o'lchovli va dinamik massivlar.
29.C++ tilida satriy kattaliklar
30.C++ tilida sinflar.
31.C++ tilida grafika
32.C++ tilida multimedia
33.C++ tilida animatsiyalar
34.C++ tilida fayllar bilan ishlash.
35.C++ tilida Dialog oynalari.
36.C++ tilida panel yaratuvchi komponentlar.
37.C++ tilida menyu yaratuvchi komponentlar.
38.C++ tilidagi dasturlarning tarkibiy qismlari.
39.C++ da maxsus belgilar.
40.O'zgarmaslar. Literal o'zgarmaslar.
41.Belgili o'zgarmaslar.
42.Ifodalar va operatorlar.
43.Matematik operatorlar. Operatorlar prioriteti.
44.Increment va decrement operatorlari
45.Prefiks va postfiks
46.Bloklar va kompleks ifodalar.
47.Xotirani zahiralash. Butun sonlar o'lchami.
48.Ishorali va ishorasiz tiplar. O'zgaruvchilarning tayanch tiplari
49.Ma'lumotlar tipini keltirish (data casting)
50.Matematik kutubhona funksiyalari
51.Funksiyalarning tuzilishi
52.Funksiyalarning qo'llanilishi.
53.E'lon fayllari
54.Tasodifiy qiymatlarni keltirib chiqarish
55.if operatori orqali murakkab konstruksiyalarni hosil qilish
56.Dastur birliklarining sifatlari
57.O'zgaruvchining qo'llanilish sohasi (scope rules)
58.Argument olmaydigan funksiyalar
59.Ko'rsatkichlar va funksiya chaqiriqlarida ularning qo'llanilishi
60.Funksiya argumentlarning berilgan qiymatlari
61.Funksiya ismi yuklanishi
62.Funksiya shablonlari
63.Bir necha indeksli massivlar
64.Pointer (ko'rsatkich) va satrlar
65.Pointer operatorlari
66.Pointer argumentli funksiyalar
67.Const sifatli pointerlar
68.Pointer va oddiy o'zgaruvchilarning egallagan adres kattaligi
69.Borland C++ Builderda grafik axborotlar bilan ishlovchi komponentlar
70.Borland C++ Builderda Image va PaintBox komponentlarida foydalanish
71.Borland C++ Builderda Chart va VtChart komponentlarida foydalanish

72.Borland C++ Builderda Animation va MediaPlayer komponentlarida foydalanish
73.Borland C++ Builderda Win32 komponentlar palitrasidan foydalanish
74.Borland C++ Builderda System komponentlar palitrasidan foydalanish
75.Borland C++ Builderda ColorDialog va ColorBox komponentlari
76.Borland C++ Builderda PrintDialog va PrintSetupDialog komponentlari
77.Borland C++ Builderda Data Access komponentlar palitrasi
78.Borland C++ Builderda xossa va metodlar
79.Borland C++ Builderda massivlarni saralash
80.Borland C++ Builderning Excel dasturi bilan hamkorligi

Mustaqil o'zlashtiriladigan mavzular bo'yicha talabalar tomonidan referatlar tayyorlash va uni taqdimot qilish tavsiya etiladi.

Fan bo'yicha kurs ishi. Fan bo'yicha kurs ishi rejalashtirilmagan.

VI. Asosiy va qo'shimcha o'quv adabiyotlar hamda axborot manbaalari

Asosiy adabiyotlar

1. Peter Gottschling. Discovering Modern C++. An Intensive Course for Scientists, Engineers, and Programmers. "Addison-Wesley", 2015 y.
2. A. R. Azamatov, B. Boltayev. Algoritmlash va dasturlash asoslari. O'quv qo'llanma. T. : "Cho'lpon", 2010 y.
3. A. R. Azamatov, B. Boltayev. Algoritmlash va dasturlash asoslari. O'quv qo'llanma. T. : "Cho'lpon", 2013 y.
4. Sh. I. Razzoqov, M. J. Yunusova. Dasturlash: Kasb-hunar kollejlari uchun o'quv qo'llanma. T. : "Ilim Ziyo", 2011y.
5. М. Ашуров, М. Мирмахмудов, Ш. Сапаев. Замонавий дастурлаш тиллари фанидан лаборатория ишлари. Т. : ТДПУ, 2008 й.
6. Меняев Михаил Федорович. Информационные технология управления. Москва, «Издательский Омегал», 2003 г.

Qo'shimcha adabiyotlar

1. Мирзиёев Шавкат Миромонович. Эркин ва фаровон, демократик Ўзбекистон давлатини биргаликда барпо этамиз. Ўзбекистон Республикаси Президенти лавозимига киришиш тантанали маросимига бағишланган Олий Мажлис палаталарининг қўшма мажлисидаги нутқ / Ш.М. Мирзиёев. – Тошкент : Ўзбекистон, 2016. - 56 б.
2. Мирзиёев Шавкат Миромонович. Танқидий таҳлил, қатъий тартиб-интизом ва шахсий жавобгарлик – ҳар бир раҳбар фаолиятининг кундалик қоидаси бўлиши керак. Мамлакатимизни 2016 йилда ижтимоий-иқтисодий ривожлантиришнинг асосий якунлари ва 2017 йилга мўлжалланган иқтисодий дастурнинг энг муҳим устувор йўналишларига бағишланган Вазирлар Маҳкамасининг кенгайтирилган

мажлисидаги маъруза, 2017 йил 14 январ / Ш.М. Мирзиёев. – Тошкент : Ўзбекистон, 2017. – 104 б.
3. Мирзиёев Шавкат Миромонович. Қонун устуворлиги ва инсон манфаатларини таъминлаш – юрт тараққиёти ва халқ фаровонлигининг гарови. Ўзбекистон Республикаси Конституцияси қабул қилинганининг 24 йиллигига бағишланган тантанали маросимдаги маъруза. 2016 йил 7 декабр /Ш.М.Мирзиёев. – Тошкент: "Ўзбекистон", 2017. – 48 б.
4. Мирзиёев Шавкат Миромонович. Буюк келажагимизни мард ва олижаноб халқимиз билан бирга қурамиз. Мазкур китобдан Ўзбекистон Республикаси Президенти Шавкат Мирзиёевнинг 2016 йил 1 ноябрдан 24 ноябрга қадар Қорақалпоғистон Республикаси, вилоятлар ва Тошкент шаҳри сайловчилари вакиллари билан ўтказилган сайловолди учрашувларида сўзлаган нутқлари ўрин олган. /Ш.М.Мирзиёев. – Тошкент: : "Ўзбекистон", 2017. – 488 б
5. Ўзбекистон Республикаси Президентининг Фармони. Ўзбекистон республикасини янада ривожлантириш бўйича харакатлар стратегияси тўғрисида. (Ўзбекистон Республикаси конун хужжатлари тўплами, 2017 й., 6-сон, 70-модда)
6. Ўзбекистон Республикаси Конституцияси. Т.: Ўзбекистон. 2014. -46 б.
7. Nazirov Sh.A., Qobulov R.V., Bobojanov M.R., Raxmanov Q.S. C va C++ tili. "Voris-nashriyot" MCHJ, Toshkent 2013. 488 b.
8. Безручко В.Т., Практикум по курсу информатики. М. : «Финансы и статистика», 2004 г.
9. Дарахвелидзе П., Марков Э.. Программирование в Delphi7. Учебник. Санкт-Петербург, "БХВ-Петербург" 2003 г.
10. Пестиков В. М., Маслобоев А. Н.. Turbo PASCAL 7. 0. Изучаем на примерах. Санкт-Петербург. : "БХВ-Петербург", 2004 г.
11. Фаронов В. В. Программирование на языке высокого уровня Delphi. Учебник. М. : "Питер", 2003 г.

Internet saytlari

1. www.ziyonet.uz – Axborot ta'lim portali
2. www.edu.uz – Oliy va o'rta maxsus ta'lim vazirligi portali
3. www.tdpu.uz – Nizomiy nomidagi TDPU rasmiy sayti
4. http://acm.tuit.uz/ - dasturiy yechim to'g'riligini avtomatik testlovchi tizim.
5. http://acm.tuit.uz/forum/, http://acm.timus.ru/ – dasturlarni testlovchi tizim.

5. М. Ашуров, М. Мирмахмудов, Ш. Сапаев. Замонавий дастурлаш тиллари фанидан лаборатория ишлари. Т. : ТДПУ, 2008 й.
6. Меняев Михаил Федорвич. Информационные технология управления. Москва, «Издательский Омега», 2003 г.

Qo'shimcha adabiyotlar

1. Мирзиёев Шавкат Миромонович. Эркин ва фаровон, демократик Ўзбекистон давлатини биргаликда барпо этамиз. Ўзбекистон Республикаси Президенти лавозимига киришиш тантанали маросимига бағишланган Олий Мажлис палаталарининг қўшма мажлисидаги нутк / Ш.М. Мирзиёев. – Тошкент : Ўзбекистон, 2016. - 56 б.
2. Мирзиёев Шавкат Миромонович. Танқидий таҳлил, қатъий тартиб-интизом ва шахсий жавобгарлик – ҳар бир раҳбар фаолиятининг кундалик қондаси бўлиши керак. Мамлакатимизни 2016 йилда ижтимоий-иқтисодий ривожлантиришнинг асосий якунлари ва 2017 йилга мўлжалланган иқтисодий дастурнинг энг муҳим устувор йўналишларига бағишланган Вазирлар Маҳкамасининг кенгайтирилган мажлисидаги маъруза, 2017 йил 14 январ / Ш.М. Мирзиёев. – Тошкент : Ўзбекистон, 2017. – 104 б.
3. Мирзиёев Шавкат Миромонович. Қонун устуворлиги ва инсон манфаатларини таъминлаш – юрт тараққиёти ва халқ фаровонлигининг гарови. Ўзбекистон Республикаси Конституцияси қабул қилинганининг 24 йиллигига бағишланган тантанали маросимдаги маъруза. 2016 йил 7 декабр /Ш.М.Мирзиёев. – Тошкент: "Ўзбекистон", 2017. – 48 б.
4. Мирзиёев Шавкат Миромонович. Буюк келажагимизни мард ва олижаноб халқимиз билан бирга қурамиз. Мазкур китобдан Ўзбекистон Республикаси Президенти Шавкат Мирзиёевнинг 2016 йил 1 ноябрдан 24 ноябрга қадар Қорақалпоғистон Республикаси, вилоятлар ва Тошкент шаҳри сайловчилари вакиллари билан ўтказилган сайловолди учрашувларида сўзлаган нутклари ўрин олган. /Ш.М.Мирзиёев. – Тошкент: : "Ўзбекистон", 2017. – 488 б
5. Ўзбекистон Республикаси Президентининг Фармони. Ўзбекистон республикасини янада ривожлантириш бўйича ҳаракатлар стратегияси тўғрисида. (Ўзбекистон Республикаси конун ҳужжатлари тўплами, 2017 й., 6-сон, 70-модда)
6. Ўзбекистон Республикаси Конституцияси. Т.: Ўзбекистон. 2014. -46 б.
7. Nazirov Sh.A., Qobulov R.V., Bobojanov M.R., Raxmanov Q.S. C va C++ tili. "Voris-nashriyot" MCHJ, Toshkent 2013. 488 b.
8. Безручко В.Т.. Практикум по курсу информатики. М. : «Финансы и

статистика», 2004 г.
9. Дарахвелидзе П., Марков Э.. Программирование в Delphi7. Учебник. Санкт-Петербург, "БХВ-Петербург" 2003 г.
10. Пестиков В. М., Маслобоев А. Н.. Turbo PASCAL 7. 0. Изучаем на примерах. Санкт-Петербург, : "БХВ-Петербург", 2004 г.
11. Фаронов В. В. Программирование на языке высокого уровня Delphi. Учебник. М. : "Питер", 2003 г.

Internet saytlari

1. www.ziyonet.uz – Axborot ta'lim portali
2. www.edu.uz – Oliy va o'rta maxsus ta'lim vazirligi portali
3. www.tdpu.uz – Nizomiy nomidagi TDPU rasmiy sayti
4. http://acm.tuit.uz/ - dasturiy yechim to'g'riligini avtomatik testlovchi tizim.
5. http://acm.tuit.uz/forum/, http://acm.timus.ru/ – dasturlarni testlovchi tizim.

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

GULISTON DAVLAT UNIVERSITETI

Amaliy matematika va axborot texnologiyalari kafedrasi

“Tasdiqlayman”
O‘quv ishlari bo‘yicha prorektor
_____ F.G.Sharipov
“ 29 ” 07 2020 y.

DASTURLASH TILLARI
fani bo‘yicha

ISHCHI O‘QUV DASTURI

Bilim sohasi: 100000 - Gumanitar
Ta’lim sohasi: 110000 - Pedagogika
Ta’lim yo’nalishi: 5110700 - Informatika o‘qitish metodikasi

Bosqich 2
Semestr III – IV
Umumiy yuklama hajmi: 260
Umumiy o‘quv soati: 154
Shu jumladan:
Ma’ruza - 48
Amaliy mashg’ulot - 42
Laboratoriya mashg’uloti - 64
Mustaqiil ta’lim - 106

GULISTON – 2020

Fanning ishchi o‘quv dasturi namunaviy o‘quv dasturi va o‘quv rejasiga muvofiq ishlab chiqildi.

Tuzuvchi: Abdurahimov D.B. – GulDU “Amaliy matematika va axborot texnologiyalari” kafedrasi katta o‘qituvchisi, pedagogika fanlari nomzodi _____

Taqrizchi: Allayorov S.P.– GulDU “Amaliy matematika va axborot texnologiyalari” kafedrasi dotsenti, texnika fanlari nomzodi _____

Fanning ishchi o‘quv dasturi “Amaliy matematika va axborot texnologiyalari” kafedrasining 2020 yil “27” 08 dagi 1 - sonli yig‘ilishida muhokama qilindi va Fizika-matematika fakulteti Ilmiy – metodik Kengashida ko‘rib chiqish uchun tavsiya qilindi

Kafedra mudiri: _____ A.A.Qalandarov

Fanning ishchi o‘quv dasturi “Fizika-matematika” fakulteti Ilmiy-uslubiy Kengashining 2020 yil “28” 08 dagi 1 - sonli majlisida tasdiqlandi.

Fakultet Ilmiy-uslubiy
Kengashi raisi: _____ S.P.Allayorov

Fanning ishchi o‘quv dasturi Guliston davlat universiteti O‘quv-metodik Kengashining 2020 yil “29” avgustdagi 1 - sonli majlisida muhokoma etildi va maqullandi.

II. O‘quv fanining dolzarbligi va oliy kasbiy ta’limdagi o‘rni

Mustaqil Resrublikamizda yuz berayotgan siyosiy, iqtisodiy, ilmiy-texnikaviy va madaniy o‘zgarishlar Oliy ta’lim tizimida ham o‘z aksini topmoqda. O‘zbekistonda uzluksiz ta’lim-tarbiya tizimini yaratish, shu asosida ta’lim sifatini jahon andozalari darajasiga etkazish ta’lim sistemasining eng dolzarb vazifasiga aylandi. Bu esa barcha mutaxassisliklar qatori Informatika va dasturlash bo‘yicha kadrlar tayyorlash sifatini oshirishni ham taqozo etadi. Bu maqsad vazifalar ushbu fan dasturi mazmunini ham belgilaydi. Algoritm konsepsiyasining vujudga kelishi bilan algebra, sonlar nazariyasi, geometriya va matematikaning boshqa sohalariga tegishli bir qator muammolarning echimli yoki echimli emasligini aniqlashtirish imkonini berdi. Algoritmlar nazariyasi faoliyat sohasi EHMlar vujudga kelishi bilan yanada kengaydi. Yuqoridagi fikrlar “Dasturlash tillari” fanining asosiy mazmunini belgilashga yordam beradi.

“Dasturlash tillari” fani umumkasbiy fanlar blokiga kiritilgan kurs hisoblanib, 2-va 3-kurslarda o‘qitilishi maqsadga muvofiq. “Dasturlash tillari” fani “Informatika o’qitish metodikasi” ta’lim yo‘nalishida o‘qitiladi. Mazkur fan Algoritmlar fanining nazariy va uslubiy asosini tashkil qilib, o‘z rivojida aniq va tabiiy fanlar uchun zamin bo‘lib xizmat qiladi.

III. O‘quv fanining maqsadi va vazifasi

“Dasturlash tillari” fanini o‘qitishdan maqsad – talabalarga dasturlashning ilmiy-nazariy asoslarini, informatika o‘qituvchisining kasbiy sohasida egallashi lozim bo‘lgan bilimlar, amalda qo‘llash uchun ko‘nikma va makalalarni shaklantirish hamda rivojlantirishdan iborat.

Ushbu maqsadga erishish uchun fan talabalarni ob’ektga yo‘naltirilgan dasturlash tillarida ishlash, amaliy masalalarga dasturlar tuzishga oid nazariy bilimlar, amaliy ko‘nikma va malakalarini shakllantirish vazifalarini bajaradi.

Fan bo‘yicha talabalarning bilim, ko‘nikma va malakalariga quyidagi talablar qo‘yiladi. ***Talaba:***

– ob‘yektga yo‘naltirilgan dasturlash tillarining nazariy asoslari, ob‘yektlarni loyihalash, matematik va interfeys ob‘yektlari, voqealar va xabarlar, ob’yektga yo‘naltirilgan muhitlarda xabarlarni uzatish, ularga ishlov berish mexanizmlari, ob’yektlar iyerarxiyasi asosida dasturlarni loyihalash, muayyan ob’yektga yo‘naltirilgan dasturlash tillari to‘g‘risida ***tasavvurga ega bo‘lishi;***
– ob’yektga yo‘naltirilgan dasturlash tillarida chiziqli, tarmoqlanuvchi va takrorlanuvchi va modulli dasturlar tuza olishni, dasturlashning ob’yektga yo‘naltirilgan paradigmasini, ob‘yektga yo‘naltirilgan muhitlarda dasturlarni loyihalashni ***bilishi va ulardan foydalana olishi;***
– ob’yektga yo‘naltirilgan dasturlash tillari muhitida ishlash, masalalarni tahlil qila olish, muayyan dasturlash tillari yordamida masalalarning dasturini tuzish va natijalarni taqqoslay olish ***ko‘nikmalariga ega bo‘lishi lozim.***

Fandan o‘tiladigan mavzular va ular bo‘yicha mashg’ulot turlariga ajratilgan soatlarning taqsimoti

№	Mavzu	Soatlar				
		Jami	Ma’ruza	Ama-liy	Labor	Mustaqil ta’lim
	3 – semestr					
1	Dasturlash tillari faniga kirish	2	2			
2	Yuqori darajali dasturlash tillari	2				2
3	Interpretatorlar va kompilyatorlar	2				2
4	Ob’ektga yo‘naltirilgan dasturlash tillari	4	2			2
5	Ob’ektga yo‘naltirilgan loyihalash.	4	2			2
6	Voqealar va habarlar.	2	2			
7	Ob’ektlar ierarxiyasi asosida dasturlarni loyihalash.	2				2
8	Delphi dasturlash tili ishchi muhiti.	2	2			
9	Komponent xosslarini dinamik va statik o‘zgartirish.	2		2		
10	Delphi qo‘llaniladigan matematik funksiyalar	2				2
11	Delphidagi dasturning interfeys qismini yaratish..	2			2	
12	TForm komponentlari va ularning xossalari	2				2
13	Delphida muloqot dasturi yaratish	2			2	
14	Standard va Additional bo‘limi komponentlaridan foydalanish yo‘llari	4		4		
15	Delphi dasturlash muxitida chiziqli dasturlar tuzish.	4			2	2

16	Standard komponentlar palitrasi	2	2			
17	Additional komponentlar palitrasi.	2	2			
18	Delphi dasturlari strukturasi. Loyiha va modul.	2	2			
19	Delphida tiplar, o‘zgarmaslar, o‘zgaruvchilar va standart funksiyalar.	2	2			
20	Delphidagi dasturlarda tiplardan foydalanish.	2		2		
21	Delphi dasturlash muxitida tarmoq operatorlari.	2	2			
22	Delphi dasturlash tilida tarkibiy operatorlar va tanlash operatori.	2		2		
23	Delphi dasturlash muxitida tarmoqlanuvchi dasturlar tuzish.	2			2	
24	Delphi dasturlash tilida shartli va sikli dasturlar tuzish.	4				4
25	Delphi dasturlash muxitida siklik operatorlar.	4	2	2		
26	Delphi dasturlash muxitida For siklik operatoridan foydalanich.	2			2	
27	Delphi dasturlash muxitida While siklik operatoridan foydalanich.	2			2	
28	Delphi dasturlash muxitida Repeat siklik operatoridan foydalanich.	2			2	
29	Delphi dasturlash tilida tasodofiy sonlar bilan ishlash	2				2
30	Delphida massivlar.	4	4			
31	Delphi dasturlash tilida massivlar bilan ishlash	2				2
32	Delphi dasturlash tilida massivlar va satriy kattaliklar	2		2		
33	Delphida bir o’lchovli massivlarga doir dastur tuzich.	2			2	
34	Delphida ikki o’lchovli massivlarga doir dastur tuzich.	2			2	
35	Delphi dasturlash tilida sana-vaqt turi bilan ishlash	2				2
36	Prosedura va funksiyalar	2	2			
37	Delphi dasturlash tilida protsedura va funksiyalar	2		2		
38	Delphi dasturlash tilida to‘plamlar bilan ishlash.	4			2	2
39	Delphi dasturlash tilining grafik vositalari.	2	2			
40	Delphi dasturlash tilida grafik primitivlar bilan ishlash	2				2
41	Delphi dasturlash tilida ListBox ga grafiklarni joylashtirish	2				2
42	Delphida simvolli va satriy kattaliklar bilan ishlash.	4			2	2
43	Delphida modullar va ulardan foydalanish	2		2		
44	Delphi dasturlash muhitida funksiyalardan foydalanish	4			2	2
45	Delphi dasturlash muhitida proseduralardan foydalanish	4			2	2
46	Delphi dasturlash tilining Office dasturlari bilan hamkorligi	2		2		
47	Delphi dasturlash muxitida fayllar bilan ishlash. Mul’timedia ilovalari	4		2		2
48	Delphida MBni boshqaradigan ilovalar tuzish	2		2		

49	Delphida standart modular va ulardan foydalanish	2			2	
50	Delphida modul yaratish va undan foydalanish	2			2	
51	Biror predmed sohasiga oid o‘rgatuvchi dasturlar yaratish	4				4
52	Delphi dasturlaqsh tilida nazorat qiluvchi dasturlar yaratish	4				4
53	Delphida forma xossalari va ularni o‘zgartirish	2				2
54	Delphi dasturlash tilining Word, Excel, Access dasturlari bilan hamkorligi	2			2	
55	Delphi ning grafik komponrntlari	2		2		
56	Delphi dasturlash tili komponentlar palitrasi bo‘limlari va ayrim komponentlar xossalari	4				4
57	Delphi dasturlash tilida grafik elementlar bilan ishlsdh	6			2	4
58	Delphi dasturlash muxitida fayllar bilan ishlash	2			2	
59	Delphida multimediyali ilovalar yaratish.	2			2	
60	Delphi dasturlash tilining multimedia imkoniyatlari	4				4
	3 - semestr jami:	**156**	**30**	**26**	**38**	**62**
	4 - semestr					
1	C++ tilining leksik asoslari.	2	2			
	C++ da maxsus belgilar.	**2**				2
2	C++ da ma’lumotlarning asosiy turlari bilan amallar bajarish	2		2		
3	C++ dasturlash tilida ma’lumotlarning asosiy turlari bilan amallar bajarish	2			2	
4	O’zgaruvchi va o’zgarmas tipli kattaliklar.	2	2			
5	C++ dasturlash tilida qiymat berish operatorlari bilan ishlash	4			2	
6	C++ tilida chiziqli dasturlash	2		2		
7	C++ dasturlash tilida arifmetik va mantiqiy amallar bilan ishlash. C++ dasturlash tilida chiziqli dasturlar tuzish	4			2	2
8	Dasturlash operatorlari. Shartli operatorlar	4	2			2
9	C++ tilining boshqarish operatorlari	2				2
10	C++ tilida shartli va shartsiz o‘tish operatorlari. Tanlash operatori	4		2		2
11	C++ dasturlash tilida If/Else hamda Switch strukturalari bilan ishlash	4			2	2
12	C++ dasturlash tilida takrorlanuvchi jarayonlar.	2	2			
13	C++ tilida takrorlanish operatorlari (while, do while, for)	4		2		2
14	C++ dasturlash tilida While, Do_While siklik operatoridan foydalanish	4			2	2
15	C++ dasturlash muxitida *For* takrorlash strukturasi bilan ishlash. Boshqaruv strukturalarida Continue va Break ifodalarini qo’llash	4			2	2
16	C++ dasturlash tilida funksiyalar.	4	2			2
17	C++ dasturlash tilida funktsiyalar yaratish va ulardan foydalanish	2			2	

18	C++ tilida funksiyalar, strukturalar va birlashmalar.	2				2
19	C++ tilida ko‘rsatkichlar va murojaatlar	2				2
20	C++ dasturlash tilida massivlar	**6**	2	2		2
21	C++ da bir o’lchovli massivlar bilan ishlash	**2**			2	
22	C++ da ikki o’lchovli massivlar bilan ishlash	**2**			2	
23	C++ tilida bir o‘lchovli, ikki o‘lchovli va dinamik massivlar.	**2**				2
24	C++ tilida funksiyalar yaratish	**4**		2		2
25	C++ da ko`rsatkichlar. C++ da satrlar va ular ustida amallar	**2**	2			
26	C++ dasturlash tilida ko’rsatkichlar bilan ishlash	**4**			2	2
27	C++ da satriy kattaliklar bilan ishlash	**4**			2	2
28	C++ da strukturalar va birlashmalar. C++ tilida ko`rsatkichlar	**4**	2	2		
29	C++ dasturlash tilida strukturalar bilan ishlash va strukturada ko’rsatkichlardan foydalanish	**4**			2	2
30	C++ da fayllar bilan ishlash	**4**	2			2
31	C++ dasturlash tilida matnli fayllar bilan ishlash va funktsiyalarni qayta yuklash	**4**			2	2
32	C++ tilida grafika	**2**				2
33	C++ tilida sinflar. C++ tilida multimedia va animasiyalar	**4**		2		2
	4 - semestr jami:	**104**	**18**	**16**	**26**	**44**
	Umumiy jami:	2 60	48	**42**	**64**	**106**

IV. Asosiy nazariy qism (ma‘ruza mashg‘ulotlari)
Fanning nazariy mashg’ulotlari mazmuni

1-MODUL. OB’EKTGA YO‘NALTIRILGAN DASTURLASH TILLARI

1-mavzu. Dasturlash tillari faniga kirish (2 soat).

Dasturlash tillari va ularning klassifikasiyasi. Mashinaga mo‘ljallangan va proseduraga mo‘ljallangan dasturlash tillari. Yuqori darjali dasturlash tillari. Interpretatorlar va kompilyatorlar. Dasturlarni translyasiyalash. Muyyan dasturlash tilining alifbosi, buruqlar tizimi va operatorlari.

2-mavzu. Ob’ektga yo‘naltirilgan dasturlash tillari. (2 soat).

Ob’ektga yo‘naltirilgan dasturlash tillari. Dasturlashning ob’ektga yo‘naltirilgan paradigmasi.

3-mavzu. Ob’ektga yo‘naltirilgan loyihalash. (2 soat).

Ob‘ektlarni loyihalash: satrlar, steklar, ro’yxatlar, navbatlar, daraxtlar. Matematik ob’ektlar: rasional va kompleks sonlar, vektorlar, matrisalar. Ob’ektlar kutubxonasi. Interfeys ob’ektlari: boshqarish elementlari, oynalar, dialoglar.

4-mavzu: Voqealar va habarlar. (2 soat).

Voqealar va habarlar. Ob’ektga yo’naltirilgan muhitlarda habarlarni uzatish va ularga ishlov berish mexanizmlari. Ob’ektlar ierarxiyasi asosida dasturlarni loyihalash. Muayyan ob’ektga yo’naltirilgan dasturlash tili va unda dastur tuzish asoslari

2-MODUL. DELPHI DASTURLASH TILI

5-mavzu. Delphi dasturlash tili ishchi muhiti. (2 soat).

Delphi dasturlash tilining ishchi muhiti, undagi oynalar (Ob’ektlarning daraxtsimon ko‘rinish oynasi, ob’ektlar inspektori oynasi, kod brauzeri oynasi, asosiy oyna, forma oynasi, dastur kodi oynasi), u o‘rnatilishi zarur bo‘lgan kompyuterga qo‘yiladigan texnik talablar va instrumental tugmalar. Komponentlar palitrasi.

6-mavzu. Standard komponentlar palitrasi. (2 soat).

Frame komponenti va uning xossalari. MainMenu, PopupMenu komponentlari. Label, Edit, Button, Memo, Panel komponentlari va ularning xossalari. CheckBox, ListBox, ComboBox, ListBox, RadioGroup, RadioButton, ScrollBar komponentlari.

7-mavzu. Additional komponentlar palitrasi. (2 soat).

BitBtn, MaskEdit, StringGrid, MaskEdit, CheckListBox, DrawGrid, Image, Shape va boshqa komponentlaridan foydalanish. Komponentlar xossalalri.

3-MODUL. DELPHIDA DASTURLASH

8-mavzu. Delphi dasturlari strukturasi. Loyiha va modul. (2 soat).

Bo‘sh forma va uning modifikasiyasi. Delphida nomlanishlar, forma xossalarini o‘zgartirish. Formaga yangi komponent joylashtirish va unda komponent xossalaridan foydalanish, Xodisa tushunchasi. Loyiha va modul strukturasi. Dastur elementlari (alfavit, identifikatorlar, doimiyliklar, ifodalar va amallar).

9-mavzu. Delphida tiplar, o‘zgarmaslar, o‘zgaruvchilar va standart funksiyalar. (2 soat).

Delphida tiplar, ularning ahamiyati. Butun tiplar: sodda (tartib va xaqiqiy) tiplar, mantiqiy va simvolli tiplar, tip-diapazon, vaqt-sana tipi. Delphida simvolli va satriy tiplar. Simvolli va satriy tiplarning berilishi, ular bilan bajariladigan amallar. Simvolli va satriy kattaliklar. Delphi dasturlash tilida o‘zgarmaslar, o‘zgaruvchilar va standart funksiyalar.

4-MODUL. DELPHI DASTURLASH TILI OPERATORLARI.

10-mavzu. Delphi dasturlash muxitida tarmoq operatorlari. (2 soat).

Tarkibiy va bo‘sh operatorlar. Shart va mantiqiy ifodalar. If…Then…else shartli operatori. Tanlash (case) operatori. Goto o‘tish operatori. Label (belgilar) xizmatchi so‘zidan foydalanish qoidalari bilan tanishish.

11-mavzu. Delphi dasturlash muxitida siklik operatorlar. (2 soat).

Delphi dasturlash tilida sikllar. For sikli. While sikli. Repeat sikli. Murakkab sikllar.

12-mavzu. Delphida massivlar. (4 soat).

Delphi dasturlash tilida massivlar. Massivlarni tavsiflash, e'lon qilish. Massivlarni berilish usullari. Massiv elementlarini kiritish va chiqarish. Random (max) funksiyasi bilan tanishtirish. Ko'p o'lchovli massivlar. Massiv elementlarini saralash va tartiblash.

13-mavzu. Prosedura va funksiyalar. (2 soat).

Prosedura va funksiyalar Delphi dasturlash tilining muhim instrumenti sifatida. Prosedura tarifi, uning nomi, undan foydalanish yo'llari. Funksiya tarifi, uning nomlanishi, undan dasturda foydalanish va uning proseduradan farqi.

14-mavzu. Delphi dasturlash tilining grafik vositalari. (2 soat).

Delphi dasturlash tilining grafik imkoniyatlari. Delphidagi maxsus TCanvas, TFont, TPen, Tbrush klasslari. TFont klassi xossalari: Color, Name, Size, Style. TPen klassi xossalari: Color, Mode, Width, Style. TBrush klassi xossalari: Bitmap, Color, Style.

5-MODUL. C++ DASTURLASH TILIGA KIRISH

15-mavzu. C++ tilining leksik asoslari. (2 soat).

C++ dasturlash tiliga kirish. C++ dasturlash tili alifbosi va xizmatchi so'zlari. Amallar. Izohlar satrini tavsiflash. C++ tilida operatorlar. Standart funksiyalar va ularning yozilishi. Konsol orqali muloqot qilish. Chiqarish operatori. Kiritish operatori.

16-mavzu. O'zgaruvchi va o'zgarmas tipli kattaliklar. (2 soat).

O'zgaruvchilar. Identifikatorlar. Ma'lumotlar tipi. O'zgaruvchilarni e'lon qilish. O'zgaruvchilarni initsializatsiya qilish.

17-mavzu. Dasturlash operatorlari. Shartli operatorlar. (2 soat).

C++ dasturlash tilidagi operatsiyalar. Arifmetik operatorlar. Qiymatni bir birlikka o'zgartiruvchi operatorlar. Taqqoslash operatorlari.

C++ dasturlash tilida o'tish operatori. Shartli operatorning qisqa ko'rinishi. Shartli operatorning uzun ko'rinishi. Tanlash operatorlari. Shartsiz o'tish operatori. Ko'p tarmoqlanishlar va variant tanlash operatorlari.

18-mavzu. C++ dasturlash tilida takrorlanuvchi jarayonlar. (2 soat).

Sikl operatorlari. Oldingi shartli While takrorlash operatori. Keyingi shartli do-while takrorlash operatori. For takrorlash operatori. Uzish break operatori. Umumiy takrorlanish algoritmlari va ichma-ich takrorlanishlar. Continue operatori.

19-mavzu. C++ dasturlash tilida funksiyalar. (2 soat).

Funsiyalar haqida tushuncha va ularni yaratish. Funksiyalarrning tuzilishi. Funksiya parametrlari. Lokal va global o'zgaruvchilar. Tipsiz funksiyalar. Void

funksiyasi. Funksiyalardan foydalanish. Qiymat qaytarmaydigan funksiyalar va ular yordamida masala yechish.

20-mavzu. C++ dasturlash tilida massivlar. (2 soat).

Massivlar haqida tushuncha. Massivlarni tavsiflash va ulardan foydalanish. Bir o'lchovli massivlar. Ko'p o'lchovli (indeksli) massivlar. Massivlarni navlarga ajratish usullari.

21-mavzu. C++ da ko`rsatkichlar. C++ da satrlar va ular ustida amallar. (2 soat).

Adres (manzil) operatori. Jo'natish operatori. Ko'rsatkich tipidagi o'zgaruvchilarni e'lon qilish. Ko‘rsatkichga boshlang‘ich qiymat berish. Ko‘rsatkich ustida amallar. Adresni olish amali. Ko‘rsatkichlar va adres oluvchi o‘zgaruvchilar funksiya parametri sifatida. Ko'rsatkichlar va massivlar.

22-mavzu. C++ da strukturalar va birlashmalar. (2 soat).

Strukturalar. Ma'lumot strukturalari. Struktura ko'rsatkichlari. Strukturalar bilan ko'rsatgich a'zolar. Birlashmalar va ular ustida amallar. Foydalanuvchi tomonidan aniqlangan berilganlar turi. Sinflar.

23-mavzu. C++ da fayllar bilan ishlash. (2 soat).

Matn fayllarini o'qish va yozish. Oqimni ochish. Fayldan o'qish. Faylga yozish. Binary fayllar bilan ishlash operatorlari. Matn va binar fayllar. O‘qish-yozish oqimlari. Standart oqimlar. Belgilarni o‘qish-yozish funksiyalari. Satrlarni o‘qish - yozish funksiyalari. Fayldan o‘qish-yozish funksiyalari. Formatli o‘qish va yozish funksiyalari. Fayl ko‘rsatkichini boshqarish funksiyalari.

Amaliy mashg'ulotlar mazmuni.

1-mavzu. Komponent xosslarini dinamik va statik o‘zgartirish. (2 soat).

Talabalarni Delphi dasturlash tilida komponent xosslarini dinamik va statik o‘zgartirishga o'rgatish.

2-mavzu. Standard va Additional bo‘limi komponentlaridan foydalanish yo‘llari. (4 soat).

Talabalarni Delphi dasturlash tilining Standard va Additional bo‘limi komponentlaridan foydalanish yo‘llari va ko‘nikmalariga ega bo‘lish.

3-mavzu. Delphidagi dasturlarda tiplardan foydalanish. (2 soat).

Talabalarni Delphi dasturlash tilidagi dasturlarda tiplardan foydalanish usullarini o'rgatish.

4-mavzu. Delphi dasturlash tilida tarkibiy operatorlar va tanlash operatori. (2 soat).

Talabalarni Delphi dasturlash tilida tarkibiy operatorlar va tanlash operatoridan foydalanib daturlar tuzish ko‘nikmalariga ega bo‘lishni o'rgatish.

5-mavzu. Delphi dasturlash tilida sikl operatorlari. (2 soat).

Talabalarni Delphi dasturlash tilida sikl operatorlaridan foydalanib daturlar tuzish ko‘nikmalariga ega bo‘lishni o’rgatish.

6-mavzu. Delphi dasturlash tilida massivlar va satriy kattaliklar. **(2 soat).**

Talabalarni Delphi dasturlash tilida massivlar va satriy kattaliklardan foydalanib daturlar tuzishga o’rgatish.

7-mavzu. Delphi dasturlash tilida protsedura va funksiyalar. **(2 soat).**

Talabalarni Delphi dasturlash tilida protsedura va funksiyalar foydalanib daturlar tuzish ko‘nikmalariga ega bo‘lishni o’rgatish.

8-mavzu. Delphida modullar va ulardan foydalanish. **(2 soat).**

Talabalarni Delphida modullar va ulardan foydalanib daturlar tuzishga o’rgatish.

9-mavzu. Delphi dasturlash tilining Office dasturlari bilan hamkorligi. **(2 soat).**

Talabalarni Delphi dasturlash tilining Office dasturlari bilan hamkorligi imkoniyatlarini o’rgatish.

10-mavzu. Delphi dasturlash muxitida fayllar bilan ishlash. Mul’timedia ilovalari. (2 soat).

Talabalarni Delphi dasturlash muxitida fayllar bilan ishlash va mul’timedia ilovalaridan foydalanib daturlar tuzishga o’rgatish.

11-mavzu. Delphida MBni boshqaradigan ilovalar tuzish. **(2 soat).**

Talabalarni Delphida MBni boshqaradigan ilovalar tuzishga o’rgatish.

12-mavzu. Delphi ning grafik komponentlari. (2 soat).

Talabalarni Delphining grafik komponentlaridan foydalanib dasturlar tuzishga o’rgatish.

13-mavzu. C++ da ma`lumotlarning asosiy turlari bilan amallar bajarish. **(2 soat).**

Talabalarni C++ da ma`lumotlarning asosiy turlari bilan amallar bajarishga o‘rgatish.

14-mavzu. C++ tilida chiziqli dasturlash. **(2 soat).**

Talabalarni C++ tilida chiziqli dasturlash tuzishga o‘rgatish.

15-mavzu. C++ tilida shartli va shartsiz o‘tish operatorlari. Tanlash operatori. (2 soat).

Talabalarni C++ tilida shartli va shartsiz o‘tish hamda tanlash operatorlaridan foydalanib dasturlar tuzishga o‘rgatish.

16-mavzu. C++ tilida takrorlanish operatorlari (while, do while, for). (2 soat).

Talabalarni C++ tilida takrorlanish operatorlari (while, do while, for). foydalanib dasturlar tuzishga o’rgatish.

17-mavzu. C++ tilida massivlar. **(2 soat).**

Talabalarni C++ tilida bir o‘lchovli massivlarga doir dasturlar tuzishga o’rgatish.

18-mavzu. C++ tilida funksiyalar yaratish (2 soat).

Talabalarni C++ tilida funksiyalar yaratishga doir dasturlar tuzishga o’rgatish.

19-mavzu. C++ tilida strukturalar va birlashmalar. C++ tilida koʻrsatkichlar (2 soat).

Talabalarni C++ tilida strukturalar va birlashmalarga hamda koʻrsatkichlarga doir dasturlar tuzishga o’rgatish.

1. **20-mavzu. C++ tilida sinflar. C++ tilida multimedia va animatsiyalar**. **(2 soat).**

Talabalarni C++ tilida sinflar hamda C++ tilida multimedia va animatsiyalarga doir dasturlar tuzishga o’rgatish.

Amaliy mashg’ulotlar bo‘yicha ko‘rsatma va tavsiyalar

Amaliy mashgʻulotlarda talabalar muayyan masala bo‘yicha mavjud bo‘lgan yoki mustaqil tarzda kichik ishchi guruhlari yordamida hosil qilingan algoritmlarni muhokama qiladilar. Mazkur mavzularga oid test masalalar tuzib, ular asosida tuzilgan dasturlar majmuasini tuzadilar va kompyuterda olingan natijalarni birgalikda tahlil qiladilar.

Amaliy mashgʻulotlarni tashkil etish bo‘yicha kafedra professor-o‘qituvchilari tomonidan ko‘rsatma va tavsiyalar ishlab chiqiladi. Unda talabalar asosiy ma’ruza mavzulari bo‘yicha olgan bilim va ko’nikmalarini amaliy masalalarga dasturlar tuzish orqali bilimlarini yanada boyitadilar. Shuningdek, darslik va o‘quv qo‘llanmalar asosida talabalar bilimlarini mustahkamlashga erishish, tarqatma materiallardan foydalanish, ilmiy maqolalar va tezislarni chop etish orqali talabalar bilimini oshirish, masalalarning dasturini tuzish, mavzular bo‘yicha ko‘rgazmali qurollar tayyorlash va boshqalar tavsiya etiladi.

Amaliy mashg‘ulotlar multimedia qurulmalari bilam jihozlangan auditoriyada bir akadem guruhga bir o‘qituvchi tomonidan o‘tkazilishi lozim. Mashg‘ulotlar faol va interfaktiv usullar yordamida o‘tilishi, mos ravishda munosib pedagogik va axborot texnologiyalar qo‘llanilishi maqsadga muvofiq.

Laboratoriya mashg’ulotlarining mazmuni.

1-mavzu. Sodda dasturlarni tuzish. (Standard va Additional bo‘limi komponentlari yordamida) (4 soat).

Sodda dasturlarni tuzish. (Standard va Additional bo‘limi komponentlari yordamida)

2-mavzu. Shartli o‘tish va sikl operatorlari yordamida dasturlar tuzish. (8 soat).

Shartli o‘tish va sikl operatorlari yordamida dasturlar tuzish.

3-mavzu. Tasodifiy sonlar bilan ishlash. (2 soat).

Tasodifiy sonlar bilan ishlash.

4-mavzu. Sana-vaqt tipi bilan ishlash. (2 soat).

Sana-vaqt tipi bilan ishlash.

5-mavzu. Satriy kattaliklar bilan ishlash. (4 soat).

Satriy kattaliklar bilan ishlash.

6-mavzu. Delphi dasturlash muhitida funksiya va proseduralardan foydalanish (8 soat).

Delphi dasturlash muhitida funksiya va proseduralardan foydalanish

7-mavzu. Delphi dasturlash muxitida modul tuzilishi va undan foydalanish. (4 soat).

Delphi dasturlash muxitida modul tuzilishi va undan foydalanish

8-mavzu. Delphi dasturlash tilining Office dasturlari bilan hamkorligi. (4 soat).

Delphi dasturlash tilining Office dasturlari bilan hamkorligi.

9-mavzu. Delphi dastur tuzish muhitining grafik imkoniyatlari. (6 soat).

Delphi dastur tuzish muhitining grafik imkoniyatlari

10-mavzu. Delphi dasturlash muxitida fayllar bilan ishlash. (6 soat).

Delphi dasturlash muxitida fayllar bilan ishlash

11-mavzu. Delphining multimediyali imkoniyatlari. (6 soat).

Delphining multimediyali imkoniyatlari

12-mavzu. Delphida spravka tizimini yaratish. **(2 soat).**

Delphida spravka tizimini yaratish

13-mavzu. Borland C++ Builder dasturlash muhitiga kirish, ishchi muhit, oynalar. (2 soat).

Borland C++ Builder dasturlash muhitiga kirish, ishchi muhit, oynalar.

14-mavzu. Borland C++ Builderda Label, Edit va Button kompenentlaridan foydalanib dastur tuzish (4 soat). Borland C++ Builderda Label, Edit va Button kompenentlaridan foydalanib dastur tuzish

15-mavzu. Borland C++ Builderda CheckBox, RadioGroup, ComboBox va ListBox kompenentlaridan foydalanib dastur tuzish. (4 soat).

Borland C++ Builderda CheckBox, RadioGroup, ComboBox va ListBox kompenentlaridan foydalanib dastur tuzish

16-mavzu. C++ tilida fayllar bilan ishlash. Dialog oynalari. (OpenDialog, SaveDialog, FindDialog va hokazo). (4 soat).

C++ tilida fayllar bilan ishlash. Dialog oynalari. (OpenDialog, SaveDialog, FindDialog va hokazo)

17-mavzu. C++ tilida panel va menyu yaratuvchi komponentlar (Panel, GroupBox, Bevel, ScroolBox, ToolBar, StatusBar). (4 soat).

C++ tilida panel va menyu yaratuvchi komponentlar (Panel, GroupBox, Bevel, ScroolBox, ToolBar, StatusBar)

18-mavzu. Borland C++ Builderda bir o'lchovli massivlarga doir dastur tuzish. (4 soat).

Borland C++ Builderda bir o'lchovli massivlarga doir dastur tuzish

Laboratoriya mashg'ulotlari bo'yicha ko'rsatma va tavsiyalar

Laboratoriya mashgʻulotlarida talabalar amaliy mashgʻulotlarda tuzilgan dasturlarni kompyuter yordamida natijalalarini koʻrib, ularni taxlil qiladilar va xulosalar chiqaradilar.

2.5. Mustaqil ta'lim topshiriqlari bo'yicha tavsiyalar

Darslik va oʻquv qoʻllanmalardan foydalanib, barcha mavzularni oʻrganish. Tarqatma materiallar boʻyicha ma'ruza qismlarini oʻzlashtirish

Talabalarning mustaqil ishlari har bir ma'ruza mavzusi asosida tashkil etiladi. Fanni oʻrganish jarayonida mustaqil ishlarning bir necha turlaridan foydalaniladi:

1) adabiyotlar bilan ishlash;
2) ijodiy ish;
3) ishlarni elektron koʻrinishda bajarish;
4) ba'zi mavzular boʻyicha referatlar tayyorlash.

Mustaqil ta'limlarni tashkil etishda internet va axborot manbalaridan doimiy foydalaniladi.

Mustaqil ta’lim va mustaqil ishlar

- Delphi bilan tanishish
- Delphi qo‘llanaladigan matematik funksiyalar
- TForm komponentlari va ularning xossalari
- Sodda dasturlarni tuzish
- Shartli va siklli dasturlar tuzish
- Tasodifiy sonlar bilan ishlash
- Massivlar bilan ishlash
- Sana-vaqt turi bilan ishlash
- Xarfiy kattaliklar bilan ishlash
- To‘plamlar bilan ishlash
- Fayllar bilan ishlash
- Funksiya va proseduralarni yaratish
- Grafik uskunalar bilan ishlash
- ListBox da grafiklarni joylashtirish
- Biror predmet sohasiga oid o‘rgatuvchi dasturlar yaratish
- Nazorat qiluvchi dasturlar yaratish
- C++ tilining boshqarish operatorlari
- C++ tilida funksiyalar, strukturalar va birlashmalar.
- C++ tilida ko‘rsatkichlar va murojaatlar
- C++ tilida bir o‘lchovli, ikki o‘lchovli va dinamik massivlar.
- C++ tilida satriy kattaliklar
- C++ tilida sinflar. C++ tilida grafika
- C++ tilida multimedia va animatsiyalar
- C++ tilida fayllar bilan ishlash.
- C++ tilida Dialog oynalari.
- C++ tilida panel va menyu yaratuvchi komponentlar.

Mustaqil o‘zlashtiriladigan mavzular bo‘yicha talabalar tomonidan referatlar tayyorlash va uni taqdimot qilish tavsiya etiladi.

2.6. Fanni o‘qitish jarayonini tashkil etish va o‘tkazish bo‘yicha tavsiyalar

“Dasturlash tillari” fanini o‘rganish davomida mashg’ulotlar paytida axborot (taqdimot, multimedia texnologiyalari) va ta’limning zamonaviy texnologiyalari (rivojlantiruvchi ta’lim texnologiyalari, fanni to‘liq o‘zlashtirishga yo‘naltirilgan texnologiyalar, shaxsga yo‘naltirilgan ta’lim texnologiyalari) hamda interfaol metodlar (“Aqliy hujum”,”BBB”, “Venn diagrammasi”, “T-chizma”, “Insert”, “Bir-biridan so‘rash”, “FSMU”, “Bumerang”, “Klaster”) qo‘llaniladi. Bundan tashqari darsliklar, o‘quv qo‘llanmalari, ma’lumotnomalar, pedagogik entsiklopediyalar va lug’atlar, ma’ruza matnlari, tarqatma materiallaridan foydalaniladi.

Ma’ruza darslarida zamonaviy kompyuter texnologiyalari yordamida prezentatsion va elektron-didaktik texnologiyalaridan, amaliy mashg`ulotlarda zamonviy pedagogik va innovatsion texnologiyalaridan, laboratoriya mashg`ulotlarida zamonaviy kompyuter sinflaridan foydalanish ko`zda tutilgan. Shuningdek buguni kun talabiga javob beradigan dasturlash tillaridan Paskal, Delphi, C++ dasturlash tillarini o`rnatuvchi disk ham bo`lishi lozim.

2.7. Fanni baholash tizimi:

2.7.1. Talabalar bilimini baholash mezonlari

Talabalarning bilimi quyidagi mezonlar asosida:

- talaba mustaqil xulosa va qaror qabul qiladi, ijodiy fikrlay oladi, mustaqil mushohada yuritadi, olgan bilimini amalda qollay oladi, fanning (mavzuning) mohiyatini tushunadi, biladi, ifodalay oladi, aytib beradi hamda fan (mavzu) bo‘yicha tasavvurga ega deb topilganda — 5 (alo) baho;
- talaba mustaqil mushohada yuritadi, olgan bilimini amalda qollay oladi, fanning (mavzuning) mohiyatni tushunadi, biladi, ifodalay oladi, aytib beradi hamda fan (mavzu) boyicha tasavvurga ega deb topilganda — 4 (yaxshi) baho;
- talaba olgan bilimini amalda qollay oladi, fanning (mavzuning) mohiyatni tushunadi, biladi, ifodalay oladi, aytib beradi hamda fan (mavzu) bo‘yicha tasavvurga ega deb topilganda — 3 (qoniqarli) baho;
- talaba fan dasturini o‘zlashtirmagan, fanning (mavzuning) mohiyatini tushunmaydi hamda fan (mavzu) boyicha tasavvurga ega emas deb topilganda — 2 (qoniqarsiz) baho bilan baholanadi.

Talabaning “Dasturlash tillari” fani bo‘yicha bilim, ko‘nikma va malakalarini baholashda quyidagi mezonlarga asoslaniladi:

a) **5 (a’lo) baho** uchun talabaning bilim darajasi quyidagilarga javob berishi lozim:

kasbiy sohasida uchraydigan turli hil masalalarga algoritmlar tuza olishi, algoritmning turlarni farqlay olish, tasvirlash usullariga oid misollar keltira olish, rekursiya va iteratsiya, algoritmning murakkabligi tushunchalarni ajrat olishi, samarali algoritmlar ishlab chiqishning asosiy usullari(balansirovka, dinamik dasturlash va boshqalar)ni amaliy qo’llay olishi, biror bir dasturlash tillari va ularning turlarini farqlay olish, dasturlash tillalari yordamida amaliy masallalarga dasturlar tuza olish, massivlar, grafik operatorlar, satriy kattaliklar bilan ishlash, funktsiyalar va protseduralar, yozuvlar, ro`yxatlar, fayllar, modulli dasturlar haqidagi bilimlarni amalda qo’llay olish, ob’ektga yo`naltirilgan dasturlash tillaridan foydalana olish, boshqarish elementlari, oynalar, dialoglar; voqealar va habarlar, ob’ektga yo`naltirilgan muhitlarda habarlarni uzatish va ularga ishlov berish, ob’ektlar ierarxiyasi asosida dasturlarni loyihalash haqidagi nazariy bilimlarga ega bo‘lishi, ushbu nazariy bilimlarni amalda qo‘llay olishi, kasbiy sohalarida fanning amaliy imkoniyatlaridan foydalana olishi, mustaqil ishlash ko‘nikmalariga ega bo‘lishi;

b) **4 (yaxshi) baho** uchun talabaning bilim darajasi quyidagilarga javob berishi lozim:

turli hil masalalarga algoritmlar tuza olishi, algoritmning turlarni farqlay olish, tasvirlash usullariga oid misollar keltira olish, rekursiya va iteratsiya, algoritmning murakkabligi tushunchalarni ajrat olishi, biror bir dasturlash tillari va ularning turlarini farqlay olish, dasturlash tillalari yordamida amaliy masallalarga dasturlar tuza olish, massivlar, grafik operatorlar, satriy kattaliklar bilan ishlash, funktsiyalar va protseduralar, haqidagi bilimlarni amalda qo’llay olish, ob’ektga yo`naltirilgan dasturlash tillaridan foydalana olish, boshqarish elementlari, oynalar, ob’ektlar ierarxiyasi asosida dasturlarni loyihalash haqidagi nazariy bilimlarga ega bo‘lishi; ushbu nazariy bilimlarni amalda qo‘llay olishi; kasbiy soxalarida fanning amaliy imkoniyatlaridan foydalana olishi;

v) **3 (qoniqarli) baho** uchun talabaning bilim darajasi quyidagilarga javob berishi lozim:

turli hil masalalarga algoritmlar tuza olishi, algoritmning turlarni farqlay olish, tasvirlash usullariga oid misollar keltira olish, rekursiya va iteratsiya, algoritmning murakkabligi tushunchalarni ajrat olishi, biror bir dasturlash tillari va ularning turlarini farqlay olish, dasturlash

tillalari yordamida amaliy masallalarga dasturlar tuza olish haqidagi qisman tassavurga ega bo‘lishi; amaliyotda ayrim dasturlarni ko‘llay olishi;

g) fanning nazariy qismini tushunmaydigan, amaliy qo‘llash imkoniyatlari juda past, dasturlarni mutaqil ravishda ishlata olmaydigan talabalarga **2 (qoniqarsiz) baho** qo‘yiladi.

2.7.2. Talabaning amaliy va laboratoriya mashgulotlarni o‘zlashtirish darajasi quyidagi mezon asosida aniqlanadi.

Baholash mezonlari	5 baholik shkala	100 ballik shkala
Yetarli nazariy bilimga ega. Topshiriqlarni mustaqil yechgan. Berilgan savollarga toliq javob beradi. Masalaning mohiyatiga toliq tushunadi. Auditoriyada faol. Oquv tartib intizomiga toliq rioya qiladi. Topshiriqlarni namunali rasmiylashtirgan.	«5»	“Alo” 90 - 100%
Yetarli nazariy bilimga ega. Topshiriqlarni yechgan. Berilgan savollarga yetarli javob beradi. Masalaning mohiyatini tushunadi. Oquv tartib intizomiga toliq rioya qiladi.	«4»	“Yaxshi” 70 - 89,9%
Topshiriqlarni yechishga harakat qiladi. Berilgan savollarga javob berishga harakat qiladi. Masalaning mohiyatini chala tushungan. Oquv tartib intizomiga rioya qiladi.	«3»	“Qoniqarli” 60 - 69,9%
Talaba amaliy mashgulot darsi mavzusiga nazariy tayyorlanib kelmasa, mavzu boyicha masala, misol va savollariga javob bera olmasa, darsga sust qatnashsa bilim darajasi qoniqarsiz baholanadi	«2»	“Qoniqarsiz” 0 - 59,9%

ON ni baholash

Oraliq nazorat “Dasturlash tillari” fanining bir necha mavzularini qamrab olgan bo‘limi bo‘yicha yozma yoki test ravishda amalga oshiriladi. Bundan maqsad talabalarning tegishli savollarni bilishi yoki muammolarni echish konikmalari va bilim malakalari aniqlanadi.

Oraliq nazorat turini otkazish va mazkur nazorat turi boyicha talabaning bilimini baholash tegishli fan boyicha oquv mashgulotlarini olib borgan professor-oqituvchi tomonidan amalga oshiriladi.

Oraliq nazorat turini topshirmagan, shuningdek ushbu nazorat turi boyicha «2» (qoniqarsiz) baho bilan baholangan talaba yakuniy nazorat turiga kiritilmaydi.

Talaba nazorat turi otkazilgan vaqtda uzrli sabablarsiz qatnashmagan hollarda jurnalga «0» belgisi yozib qoyiladi.

Oraliq nazorat turi har bir fan boyicha fanning xususiyatidan kelib chiqqan holda 2 martagacha otkazilishi mumkin. Oraliq nazorat turini otkazish shakli va muddati fanning xususiyati va fanga ajratilgan soatlardan kelib chiqib tegishli kafedra tomonidan belgilanadi.

Talabani oraliq nazorat turi boyicha baholashda, uning o‘quv mashgulotlari davomida olgan baholari inobatga olinadi.

O‘quv yilining kuzgi hamda bahorgi semestrida 2 ta yozma ish va 2 ta mustaqil ish rejalashtirilgan bo‘lib, yozma ish 5 baholik shkalada baholanadi.

Oraliq baholash (OB) “Dasturlash tillari” fanining bir necha mavzularini qamrab olgan bolimi boyicha, tegishli nazariy, amaliy va labaratoriya mashgulotlari o‘tib bo‘lingandan so‘ng yozma ish shaklida amalga oshiriladi. Bundan maqsad talabalarning tegishli savollarni bilishi yoki muammolarni yechish konikmalari va malakalari aniqlanadi. O‘quv yilining **1-semestrida** 2 ta yozma ish va 10 ta mustaqil ish rejalashtirilgan bo‘lib, yozma ishga o‘rtacha 5 ball, mustaqil ishga ham o‘rtacha 5 ball ajratilgan. **2-semestrida** 2 ta yozma ish va 6 ta mustaqil ish rejalashtirilgan bo‘lib, yozma ishga o‘rtacha 5 ball, mustaqil ishga ham o‘rtacha 5 ball ajratilgan. OB nazorat ishlari yozma ish shaklda o‘tkazilishi nazarda tutilgan, yozma ish savollari ishchi o‘quv dastur

asosida tayyorlanadi. OB da “2” baho olgan talaba o‘zlashtirmagan hisoblanadi. OB ni o‘zlashtirmagan talabalarga qayta topshirish imkoniyati beriladi.

YaN ni baholash

Yakuniy nazorat “Dasturlash tillari” fanining barcha mavzularini qamrab olgan bo‘lib, nazariy, amaliy mashgulotlar o‘tib bo‘lingandan so‘ng test yoki yozma ravishda amalga oshiriladi. Bundan maqsad talabalarning fan bo‘yicha o‘zlashtirish ko‘rsatkichlari, yani bilim darajasi yoki muammolarni echish konikmalari va malakalari aniqlanadi.

Yakuniy nazorat ishlari test usulida ham o‘tkazilishi nazarda tutilgan, test savollari ishchi o‘quv dasturi asosida tayyorlanadi.

Yakuniy nazorat turini o‘tkazish va mazkur nazorat turi bo‘yicha talabaning bilimini baholash o‘quv mashgulotlarini olib bormagan professor-o‘qituvchi tomonidan amalga oshiriladi.

Talaba tegishli fan bo‘yicha yakuniy nazorat turi o‘tkaziladigan muddatga qadar oraliq nazorat turini topshirgan bo‘lishlari shart.

Yakuniy nazorat turiga kirmagan yoki kiritilmagan, shuningdek ushbu nazorat turi bo‘yicha «2» (qoniqarsiz) baho bilan baholangan talaba akademik qarzdor hisoblanadi.

Yakuniy nazorat turi bo‘yicha talabaning bilimi «2» (qoniqarsiz) baho bilan baholangan yoki Jurnalga «0» belgisi yozib qo‘yilgan hollarda ushbu baho yoki belgi talabaning Baholash daftariga yozilmaydi.

Yakuniy baholash (YaB) “Dasturlash tillari o‘tib bo‘lingandan so‘ng test yoki yozma ish shaklida amalga oshiriladi. Bundan maqsad talabalarning fan bo‘yicha o‘zlashtirish ko‘rsatkichlari, yani bilim darajasi yoki muammolarni yechish konikmalari va malakalari aniqlanadi. YaB nazorat ishlari test yoki yozma ish usulida ham otkazilishi nazarda tutilgan, test va yozma ish savollari ishchi o‘quv dasturi asosida tayyorlanadi. OB ga ajratilgan balldan “2” va undan past ball to‘plagan talaba o‘zlashtirmagan hisoblanadi va YaB ga kiritilmaydi. YaB ni ozlashtirmagan talabalarga qayta topshirish imkoniyati beriladi. YaB bo‘yicha olinadigan test yoki yozma ish variantlari kafedra mudiri rahbarligida tuziladi va dekanatga topshiriladi.

Test usulida YaN ni baholash mezonlari:

YaB test yoki yozma ish shaklida otkaziladi. YaB test shaklida otkazilsa talabalarga variantlar asosida 30 ta test savoli beriladi. Har bir togri javob quyidagicha balldan baholanadi. To‘gri javoblar soniga qarab talabaning YaB da to‘plagan ballari aniqlanadi.

“5” – 27-30 ta

“4” – 21-26 ta

“3” – 18-20 ta

“2” – 0-16 ta

Fan dasturining information – uslubiy ta’minoti

Didaktik vositalar

1.Jihozlar va uskunalar, moslamalar: LCD-monitor, electron ko’rsatgich.

2.Video-audio uskunalar: video va audiomagnitofon, mikrofon, kolonkalar.

3.Kompyuter va mul’timediali vositalar: komp’yuter, proektor, DVD-diskovod, Web-kamera, video-ko‘z(glazok).

Asosiy va qo‘shimcha o‘quv adabiyotlar hamda axborot manbaalari

Asosiy adabiyotlar

1. Peter Gottschling. Discovering Modern C++. An Intensive Course for Scientists, Engineers, and Programmers. “Addison-Wesley”, 2015 y.
2. M.Ashurov, N.Mirzahmedova, N.Xaytullayeva. Algoritmlash va dasturlash asoslari. Uslubiy qo‘llanma. T. : “Bayoz”, 2016 y.
3. R. Azamatov, B. Boltayev. Algoritmlash va dasturlash asoslari. O‘quv qo‘llanma. T. : “Cho‘lpon”, 2010 y.

A. R. Azamatov, B. Boltayev. Algoritmlash va dasturlash asoslari. O‘quv qo‘llanma. T. : “Cho‘lpon”, 2013 y.

4. Sh. I. Razzoqov, M. J. Yunusova. Dasturlash: Kasb-hunar kollejlari uchun o‘quv qo‘llanma. T. : “Ilim Ziyo”, 2011y.
5. М. Ашуров, М. Мирмахмудов, Ш. Сапаев. Замонавий дастурлаш тиллари фанидан лаборатория ишлари. Т. : ТДПУ, 2008 й.
6. Меняев Михаил Федорвич. Информационные технология управления. Москва, «Издательский Омегал», 2003 г.

Qo‘shimcha adabiyotlar

1. Мирзиёев Шавкат Миромонович. Эркин ва фаровон, демократик Ўзбекистон давлатини биргаликда барпо этамиз. Ўзбекистон Республикаси Президенти лавозимига киришиш тантанали маросимига бағишланган Олий Мажлис палаталарининг қўшма мажлисидаги нутқ / Ш.М. Мирзиёев. – Тошкент : Ўзбекистон, 2016. - 56 б.
2. Мирзиёев Шавкат Миромонович. Танқидий таҳлил, қатъий тартиб-интизом ва шахсий жавобгарлик – ҳар бир раҳбар фаолиятининг кундалик қоидаси бўлиши керак. Мамлакатимизни 2016 йилда ижтимоий-иқтисодий ривожлантиришнинг асосий якунлари ва 2017 йилга мўлжалланган иқтисодий дастурнинг энг муҳим устувор йўналишларига бағишланган Вазирлар Маҳкамасининг кенгайтирилган мажлисидаги маъруза, 2017 йил 14 январ / Ш.М. Мирзиёев. – Тошкент : Ўзбекистон, 2017. – 104 б.
3. Мирзиёев Шавкат Миромонович**.** Қонун устуворлиги ва инсон манфаатларини таъминлаш – юрт тараққиёти ва халқ фаровонлигининг гарови. Ўзбекистон Республикаси Конституцияси қабул қилинганининг 24 йиллигига бағишланган тантанали маросимдаги маъруза. 2016 йил 7 декабр /Ш.М.Мирзиёев. – Тошкент: “Ўзбекистон”, 2017. – 48 б.
4. Мирзиёев Шавкат Миромонович. Буюк келажагимизни мард ва олижаноб халқимиз билан бирга қурамиз. Мазкур китобдан Ўзбекистон Республикаси Президенти Шавкат Мирзиёевнинг 2016 йил 1 ноябрдан 24 ноябрга қадар Қорақалпоғистон Республикаси, вилоятлар ва Тошкент шаҳри сайловчилари вакиллари билан ўтказилган сайловолди учрашувларида сўзлаган нутқлари ўрин олган. /Ш.М.Мирзиёев. – Тошкент: : “Ўзбекистон”, 2017. – 488 б
5. Ўзбекистон Республикаси Президентининг Фармони. Ўзбекистон республикасини янада ривожлантириш бўйича ҳаракатлар стратегияси тўғрисида. *(Ўзбекистон Республикаси қонун ҳужжатлари тўплами, 2017 й., 6-сон, 70-модда)*
6. Ўзбекистон Республикаси Конституцияси. Т.: Ўзбекистон. 2014. -46 б.
7. П. Дарахвелидзе, Э. Марков. Программирование в Delphi7. Учебник. Санкт-Петербург, “БХВ-Петербург” 2003 г.
8. В. М. Пестиков, А. Н. Маслобоев. Turbo PASCAL 7. 0. Изучаем на примерах. Санкт-Петербург. : “БХВ-Петербург”, 2004 г.

9. Фаронов В. В. Программирование на языке высокого уровня Delphi. Учебник. М. : “Питер”, 2003 г.

10. В.Т.Безручко. Практикум по курсу информатики. М. : «Финансы и статистика», 2004 г.

11. Nazirov Sh.A., Qobulov R.V., Bobojanov M.R., Raxmanov Q.S. C va C++ tili. “Voris-nashriyot” MCHJ, Toshkent 2013. 488 b.

12. П. Дарахвелидзе, Э. Марков. Программирование в Delphi7. Учебник. Санкт-Петербург, “БХВ-Петербург” 2003 г.

13. Фаронов В. В. Программирование на языке высокого уровня Delphi. Учебник. М. : “Питер”, 2003 г.

14. В. Т. Безручко. Практикум по курсу информатики. М. : «Финансы и статистика», 2004 г.

Internet saytlari

6. **www.ziyonet.uz** – Axborot ta’lim portali
7. **www.edu.uz** – Oliy va o‘rta maxsus ta’lim vazirligi portali
8. www.tdpu.uz **– Nizomiy nomidagi TDPU rasmiy sayti**
9. http://acm.tuit.uz/ **- dasturiy yechim to‘g‘riligini avtomatik testlovchi tizim.**
10. http://acm.tuit.uz/forum/**, http://acm.timus.ru/ – dasturlarni testlovchi tizim.**

Dasturlash tillari fanidan nazorat variantlari

Variant № 1

1. C++ dasturlash muhitiga kirish, ishchi muhit, oynalar.
2. C++ tili operatorlari
3. Quyidagi ifodani hisoblash algortmi va dasturini C++ tilida tuzing.

$$y=\begin{cases}2{,}7x+3\sqrt{x}-1{,}2x^2 & x<8\\ \sqrt[3]{x^2}+tg^3\left(x^2+1{,}2x\right) & x>=8\end{cases}$$

Variant № 2

1. C++ tili va uning leksik asosi(o'zgaruvchilar, ularning tiplari, operatorlar, standart funksiyalar)
2. C++ tilida satriy kattaliklar
3. Quyidagi ifodani hisoblash algortmi va dasturini C++ tilida tuzing.

$$y=\begin{cases}\dfrac{5}{x}+3x^2+\cos^3 x & x<9\\ \sqrt[3]{x^2-3x}-tg^2 x & x>=9\end{cases}$$

Variant № 3

1. C++ tili va uning leksik asosi(o'zgaruvchilar, ularning tiplari, operatorlar, standart funksiyalar)
2. C++ tilida satriy kattaliklar.
3. Quyidagi ifodani hisoblash algortmi va dasturini C++ tilida tuzing.

$$S=\sum_{k=1}^{n}\frac{k^4+3}{k^2\left(k+1\right)}$$

Variant № 4

1. C++ tilining boshqarish operatorlari (if, for, While, Do-while)
2. C++ tilida funktsiyalar
3. Quyidagi ifodani hisoblash algortmi va dasturini C++ tilida tuzing.

$$y=\begin{cases}\sqrt{x+\sin^2 x}-\cos^2 x & x<7\\ \cos^3 x-\sqrt[3]{x-1} & x>=7\end{cases}$$

Variant № 5

1. C++ tili va uning leksik asosi
2. C++ tilining boshqarish operatorlari (if, for, While, Do-while)
3. $P=\sum_{j=2}^{8}\prod_{i=1}^{n}\left(2ij+2^{ij}\right)^{2ji}$ ni xisoblash algoritmi va dasturini C++ tilida tuzing.

Variant № 6

1. C++ tilining boshqarish operatorlari (if, for, While, Do-while)
2. C++ dasturlash muhitiga kirish, ishchi muhit, oynalar.

3. $P=\sum_{j=2}^{8}\prod_{i=1}^{n}\left(2ij+2^{ij}\right)^{2ji}$ ni xisoblash algoritmi va dasturini C++ tilida tuzing.

Variant № 7

1. C++ tili va uning leksik asosi.
2. C++ tilida funktsiyalar
3. $P=\sum_{j=2}^{8}\prod_{i=1}^{n}\left(2ij+2^{ij}\right)^{2ji}$ ni xisoblash algoritmi va dasturini C++ tilida tuzing.

Variant № 8

1.C++ tili va uning leksik asosi(o'zgaruvchilar, ularning tiplari, operatorlar, standart funksiyalar)
2. C++ tili takrorlash operatorlari
3. $P=\sum_{j=2}^{8}\prod_{i=1}^{n}\left(2ij+2^{ij}\right)^{2ji}$ ni xisoblash algoritmi va dasturini C++ tilida tuzing.

Variant № 9

1. C++ tilida takrorlash operatorlari
2.C++ dasturlash muhitiga kirish, ishchi muhit, oynalar.
3. $P=\sum_{j=2}^{8}\prod_{i=1}^{n}\left(2ij+2^{ij}\right)^{2ji}$ ni xisoblash algoritmi va dasturini C++ tilida tuzing.

Variant № 10

1. C++ tili va uning leksik asosi
2. C++ tilida funktsiyalar
3. $P=\prod_{k=1}^{5}\sum_{n=2}^{4}\frac{\sin(k!)}{n!+k!}$ ni aniqlab beruvchi algoritm va dasturini C++ tilida tuzing.

Variant № 11

1. C++ tilida fayllar bilan ishlash.
2. C++ tilida funktsiyalar
3. $S=\sum_{k=1}^{n}\frac{k^4+3}{k^2(k+1)}$ ni xisoblash algoritmi va dasturini C++ tilida tuzing.

Variant № 12

1. C++ tili va uning leksik asosi
2. C++ tilida dastur bajarilishini boshqarish.
3. Tomonlari bilan berilgan uchburchakning teng tomonli bo'lishini xisoblash algoritmi va dasturini C++ tilida tuzing.

Variant № 13

1. C++ tili va uning leksik asosi(o'zgaruvchilar, ularning tiplari, operatorlar, standart funksiyalar)
2. C++ dasturlash muhitiga kirish, ishchi muhit, oynalar.

1. 3. $P=\prod_{k=1}^{5}\sum_{n=2}^{4}\frac{\sin(k!)}{n!+k!}$ ni xisoblash algoritmi va dasturini C++ tilida tuzing.

Variant № 14

1. C++ tili va uning leksik asosi(o'zgaruvchilar, ularning tiplari, operatorlar, standart funksiyalar)
2. C++ tilida dastur bajarilishini boshqarish.
3.Uchta har xil butun son berilgan. Shu sonlarning arifmetik proggressiya tashkil qilish yoki qilmasligini aniqlang.

Variant № 15

1. C++ tilida shart operatorlari
2.C++ dasturlash muhitiga kirish, ishchi muhit, oynalar.
3. $S=\sum_{k=1}^{n}\frac{k^4+3}{k^2(k+1)}$ ni xisoblash algoritmi va dasturini C++ tilida tuzing.

Variant № 16

1. C++ tilida o'zgarmaslar.
2. C++ tilida fayllar bilan ishlash.
3. Agar 3 ta har xil *x, y, z* butun sonlar yig'indisi 1 dan kichik bo'lsa u holda bu uchta sondan eng kichigini qolgan ikkitasining yarim yig'indilari bilan almashtiring aks holda *x va y* lardan kichigini qolgan ikkitasi yarim yig'indilari kvadrati bilan almashtiring.

ariant № 17

1. C++ tilida fayllar bilan ishlash.
2. C++ tilida dastur bajarilishini boshqarish.
3. $S=\sum_{k=1}^{n}\frac{k^4+3}{k^2(k+1)}$ funksiyani qiymatini xisoblash algoritmi va dasturini C++ tilida tuzing.

Variant № 18

1.C++ tilida o'zgarmaslar
2. C++ tilida dastur bajarilishini boshqarish.
3. Uchta har xil butun son berilgan. Shu sonlarning arifmetik proggressiya tashkil qilish yoki qilmasligini aniqlang.

Variant № 19

1. C++ tilida funksiyalar, strukturalar va birlashmalar
2.C++ tili takrorlash operatorlari
3. $S=\sum_{k=1}^{n}\frac{k^4+3}{k^2(k+1)}$ ni xisoblash algoritmi va dasturini C++ tilida tuzing.

Variant № 20

1. Identifikatorlar va kalit so'zlar

2. C++ tilida bir o'lchovli, ikki o'lchovli va dinamik massivlar.

3. $P=\sum_{j=2}^{8}\prod_{i=1}^{n}\left(2ij+2^{ij}\right)^{2ji}$ ni hisoblash algoritmi va dasturini C++ tilida tuzing

Variant № 21

1.C++ tilida fayllar bilan ishlash.

2. C++ tili operatorlari

3. $K=\sum_{i=1}^{20}\prod_{j=1}^{20}\left(\frac{2i+5j}{2ai}\right)$ ni aniqlab beruvchi algoritm va dasturini C++ tilida tuzing.

Variant № 22

1. for, do, while takrorlash operatorlarii

2.C++ dasturlash muhitiga kirish, ishchi muhit, oynalar.

3. $S=\sum_{i=1}^{5}\sum_{k=1}^{4}\frac{2k^{i-1}+5^{i}}{1n\left|k+i^{2}\right|}$ ni xisoblash algoritmi va dasturini C++ tilida tuzing.

Variant № 23

1. C++ dasturlash muhitiga kirish, ishchi muhit, oynalar.

2. C++ tilida fayllar bilan ishlash.

3. a, b, c, d, h, l, m, r, w, s sonlarining eng kattasini toppish dasturini xisoblash algoritmi va dasturini C++ tilida tuzing.

Variant № 24

1.C++ tili va uning leksik asosi(o'zgaruvchilar, ularning tiplari, operatorlar, standart funksiyalar)

2. C++ tilida bir o'lchovli, ikki o'lchovli va dinamik massivlar.

3. $S=\sum_{i=1}^{5}\sum_{k=1}^{4}\frac{2k^{i-1}+5^{i}}{1n\left|k+i^{2}\right|}$ ni xisoblash algoritmi va dasturini C++ tilida tuzing.

Variant № 25

1.C++ tilida o'zgarmaslar.

2. C++ tilida bir o'lchovli, ikki o'lchovli va dinamik massivlar.

3. $P=\prod_{k=1}^{5}\sum_{n=2}^{4}\frac{\sin(2k+1)}{3n^{3}+k^{2}+4}$ ni qiymatini xisoblash algoritmi va dasturini C++ tilida tuzing.

Variant № 26

1.C++ tilida satriy kattaliklar

2. C++ tilining boshqarish operatorlari (if, for, While, Do-while)

3. $P=\prod_{k=1}^{5}\sum_{n=2}^{4}\frac{\sin(2k+1)}{3n^{3}+k^{2}+4}$ ni hisoblash algoritm va dasturini C++ tilida tuzing.

Variant № 27

1. C++ tilining boshqarish operatorlari (if, for, While, Do-while)

2. C++ tilida fayllar bilan ishlash.

3. a, b, c, d,h,l,m,r,w,s sonlarining eng kattasini toppish dasturini xisoblash algoritmi va dasturini C++ tilida tuzing

Variant № 28

1. C++ tilining boshqarish operatorlari (if, for, While, Do-while)

2. C++ dasturlash muhitiga kirish, ishchi muhit, oynalar.

3. $S=\sum_{k=1}^{n}\frac{k^4+3}{k^2(k+1)}$ ni xisoblash algoritmi va dasturini C++ tilida tuzing.

Variant № 29

1. C++ dasturlash muhitiga kirish, ishchi muhit, oynalar.

2. C++ tilida fayllar bilan ishlash.

3. Agar 3 ta har xil a, *b, c* butun sonlar yig'indisi 1 dan katta bo'lsa u holda bu uchta sondan eng kichigini qolgan ikkitasining yarim yig'indilari bilan almashtiring aks holda a *ва* b lardan kichigini qolgan ikkitasi yig'indilari bilan almashtirishni hisoblash algoritmi va dasturini tuzing

Variant № 30

1. C++ tilida bir o'lchovli, ikki o'lchovli va dinamik massivlar.
2. C++ tilining boshqarish operatorlari (if, for, While, Do-while)
3. $S=\sum_{i=1}^{5}\sum_{k=1}^{4}\frac{2k^{i-1}+5^i}{1n\left|k+i^2\right|}$ ni qiymatini xisoblash algoritmi va dasturini C++ tilida tuzing.

Variant № 31

1. C++ tilida shart operatorlari

2. C++ dasturlash muhitiga kirish, ishchi muhit, oynalar.

3. $y=\frac{2x^2+3ab}{5xc}-\sqrt{x+\ln xa}$ funktsiyani x ning [a,b] oraliqda h=0.1 qadam bilan o'zgarish qiymatini xisoblash algoritmi va dasturini C++ da tuzing.

Variant № 32

1. C++ tilida o'zgarmaslar.
2. C++ tilida fayllar bilan ishlash.
3. $K=\sum_{i=1}^{20}\prod_{j=1}^{20}\left(\frac{2i+5j}{2ai}\right)$ funksiyani qiymatini hisoblash algoritmi va dasturini C++ da tuzing.

Variant № 33

1. C++ tilida fayllar bilan ishlash.
2. C++ tilida dastur bajarilishini boshqarish.
3. $S=\sum_{k=1}^{n}\frac{k^4+3}{k^2(k+1)}$ funksiyani qiymatini xisoblash algoritmi va dasturini C++ tilida tuzing.

Variant № 34

1.C++ tilida o'zgarmaslar

2. C++ tilida dastur bajarilishini boshqarish.

3. $y=\sqrt{x^3+x^2+x}$ funksiyani x ning [a,b] oraliqda hq0.1 qadam bilan o'zgarish qiymatini hisoblash algoritmini va dasturini C++ da tuzing.

Dasturlash tillari fanidan test variantlari

№	Test savollari	To`g`ri javob	Muqobil javob	Muqobil javob	Muqobil javob
1	C++ tili qachon tuzildi?	* 1980 y	1999 y	1986 y	1991 y
2	C++ tili kim tomonidan tuzilgan?	Jon Fon Neyman	Bil Gets	Bjarne Stroustrup	To‘g‘ri javob yo‘q
3	C++ tili qaysi tilga asoslangan?	Paskal	*C	Java	Delphi
4	C++ alfavitiga qanday simvollar kiradi?	Katta va kichik lotin alfaviti xarflari (A,B,..,Z,a,b, …,z)	Raqamlar: 0,1,2,3,4,5,6, 7,8,9	Maxsus simvollar: — , {} | [] () + - / % \ ; _ . : ? < = > _ ! & * # ~ ^	Hamma javoblar to‘g‘ri
5	….. lotin xarflari,ostki chiziq belgisi va sonlar ketma ketligidan iborat bo‘ladi?	*Identifikator	Simvol	Raqamlar	Harflar
6	Tilda ishlatiluvchi ya‘ni dasturchi tomonidan uzgaruvchilar nomlari sifatida ishlatish mumkin bulmagan identifikatorlar ….. deyiladi.	* Xizmatchi so‘zlar	Raqamlar	Harflar	Simvollar
7	C++ dagi xizmatchi so‘zlarni toping?	* Hammasi to`g`ri	Goto	While	Else
8	VARIABLES – bu ….	*o‘zgaruvchilar	O‘zgarmaslar	Simvollar	Raqamlar
9	O‘zgaruvchilarning qanday tiplari mavjud?	*hammasi	Char	Float	int
10	Butun sonlar qaysi toifaga kiradi?	*Int	Char	Float	Hammasi
11	Long char – bu …	*Uzun simvol	Butun son	Bitta simvol	Haqiqiy son
12	CONSTANTS - bu …	*O‘zgarmaslar	O‘zgaruvchil ar	Simvollar	Harflar
13	C++ tilida necha turdagi konstantalar ishlatilishi mumkin?	*5 ta	8 ta	3 ta	7 ta
14	Qanday konstantalar C++ tili konstantalariga kirmaydi?	*Satrli	Char	Float	Int
15	Qanday konstanta bu ikkilik qavslarga olingan ihtiyoriy simvollar ketma ketligidir	*Satrli	Int	Float	Char
16	C++da arifmetik operator qaysilar	+,-	/,*,%	To‘g‘ri javob yo`q	*a va b to‘g‘ri
17	C++ da solishtirish amallari qatorini toping?	*hammasi	<, >	<=, >=	==, !=
18	C++da kiritish operatori qaysi?	*cin	cout	in	Case

19	C++da chiqarish operatori qaysi?	*cout	case	then	cin
20	Char toifaning hajmi qancha	*8	13	5	9
21	Int toifaning hajmi qancha	*16	9	8	24
22	Int toifaning qiymatlar chegarasi qancha?	*-32768…32767	0..255	-128..127	0..65535
23	Unsignet Int toifaningqiymatlar chegarasi qancha?	*0..65535	0..255	-128..127	-32768…32767
24	…- ko‘rsatkich yagona arifmetik bulmagan konstantadir.	* NULL	ted	prt	Key
25	int a, b, c; cout << "a="; cin >> a; cout << "b="; cin >> b; c = a + b; cout << c << endl; return 0; bu dastur nimani hisoblaydi?	*Ikki sonnig yig‘indisi	Ikki sonnig bolinmasi	Ikki sonning ayirmasi	Ikki sonnig kopaytmasi
26	c++da matematik funksiyalardan foydalanganda qaysi fayllardan foydalaniladi?	*math.h	Iostream.h	include	string.h
27	C++da satrli ifodalardan foydalanganda qaysi fayllardan foydalaniladi?	*string.h	iostream.h	include	math.h
28	C++da sqrt qaysi ifoda turiga kiradi?	*matematik	mantiqiy	ilmiy	tog‘ri javob yo`q
29	sqrt funksiyasi nimani ifodalaydi?	*ildiz	Kvadrat	ko‘paytma	bo‘linma
30	power funksiyasi nimani ifodalaydi?	*daraja	ildiz	ko‘paytma	Kvadrat
31	l=sqrt(k) bunda l qaysi toifaga mansub bo‘ladi?	*float	char	string	Int
32	float a; cout << "a="; cin >> a; a = sqrt(a); cout << a << endl; return 0; a ning natijasini top?	*a sonning kvadrat ildizi	a sonning kopaytmasi	a sonning ildizi	tog‘ri javob yoq
33	Butun sonlar toifasini toping?	*int	float	string	Char
34	Haqiqiy sonlar toifasini toping?	*float	int	string	Char
35	Belgilar toifasini toping?	* char	float	string	int
36	Satrli toifani toping?	* string	float	int	Char
37	abs() funksiya nimani ifodalaydi?	*modul	funksiya	kvadrat	Ildiz
38	cos funksiya nimani ifodalaydi?	*sonning cosinusi	modul	kvadrat	Ildiz
39	exp funksiya nimani ifodalaydi?	*e^x	e_x	log	Lg

40	ceil(x) funksiya nimani ifodalaydi?	* x ni x dan katta yoki unga teng b-n eng kichik butun songacha yahlitlaydi	x ni x dan kichik bo'lgan eng katta butun songacha yahlitlaydi	x/y ning qoldig'ini kasr son tipida beradi	x ning absolut qiymati
41	floor(x) funksiya nimani ifodalaydi?	* x ni x dan kichik bo'lgan eng kata butun songacha yahlitlaydi	x ni x dan katta yoki unga teng b-n eng kichik butun songacha yahlitlaydi	x/y ning qoldig'ini kasr son tipida beradi	x ning absolut qiymati
42	fmod(x,y) funksiya nimani ifodalaydi?	*x/y ning qoldig'ini kasr son tipida beradi	x ni x dan kichik bo'lgan eng kata butun songacha yahlitlaydi	x ni x dan katta yoki unga teng bn eng kichik butun songacha yahlitlaydi	x ning absolut qiymati
43	If (ifoda) 1- operator Else 2-operator bu qanday operator?	*shartli	Shartsiz	to‘g‘ri javob yoq	Takrorlanuvchi
44	int a; cin >> a; if (a % 2 == 0) cout << "juft"; else cout << "toq"; return 0; dastur nimani aniqlaydi	*sonning juft yoki toqligi	Sonning juftligi	Sonning toqligi	To‘g‘ri javob yo`q
45	int a, b, max; cout << "a="; cin >> a; cout << "b="; cin >> b; max = (a > b) ? a : b; cout << max << endl; return 0; nimani aniqlaydi?	*sonning maksimali	sonning minimali	sonning toqligi	sonning juftligi
46	int main() {for (int i = 0; i < 5; i++){cout << i << endl;} return (0); yachimni toping?	*1..5 gacha sonlar	sonning minimali	sonning maksimali	sonning juftligi
47	for (int i = 0; i < 10 ; i++) cout << "Hello!"<< endl; nima chop etiladi?	* hello so‘zi 10 marta	hello so‘zi 11 marta	hello so‘zi 12 marta	hello so‘zi 9 marta
48	…. – funksiyasini har qanday sikl operatoriga qo`llash mumkin. Bu funksiya sikl tugatilishini ta‘minlaydi. Ya`ni boshqarilishni sikl operatoridan keyingi operatorga uzatadi.	* break	while	for	Repeat

49	….- funksiyasini har qanday sikl operatoriga qo`llash mumkin. Bu funksiya sikl parametrining keyingi qiymatni qabul qilishini taminlaydi. Boshqacha so`z bilan aytganda sikl tanasi tugatiladi. Bunda siklning o`zi tugatilmaydi.	* continue	for	repeat	While
50	Switch Case qanday operator?	* Kalit bo‘yicha tanlash	Shartsiz	Modul	Takrorlanis h
51	…… sikllarni tashkil qilishning eng umumiy (ommaviy) usulidir.	*For	Case	Then	While
52	Belgilar satrini teskari tartibda yozish uchun qanday operator ishlatiladi?	*Top va bot	For va while	ceil	Break va case
53	…operatori birlashgan switch, do, for, while sikllardan eng ichkisining bajarilishi tugallanilishini ta‘minlaydi.	*break	while	continue	For
54	C++ Builderda Ctrl+R tugmalari birgalikda bosilsa qanday oyna hosil bo‘ladi?	*Izlash va almashtirish	Izlash darchasini ochish	Joriy faylni saqlash	Shablonlar ro‘yxatini ochish
55	Komponentalarning qaysi xususiyati orqali matnlarni o‘ngga, chapga, o‘rtaga tekislash mumkin?	* Alignment	Align	Actions	Caption
56	Satrli sonni butun songa o‘tkazish qanday amalga oshiriladi?	* StrToInt	IntToStr	StrToFloat	FloatToStr
57	Satrli sonni haqiqiy songa o‘tkazish qanday amalga oshiriladi?	*StrToFloat	FloatToStr	StrToInt	IntToStr
58	Taymerdan foydalanishni o‘zgartirish …	*T = !T;	Button2->Caption = "Pusk"	if(!T)	Button2->Caption = "Pauza"
59	Borland C++ Builder dasturining asosiy oynalari nechta?	*5	4	3	6
60	C++ Builder dasturining Align To Grid buyrug‘i menyular satrining qaysi bo‘limida joylashgan?	*Edit	Run	Search	File
61	Memo komponentasi qaysi komponentalar palitrasiga kiradi?	* Standart	Win32	Additional	System

62	Timer komponentasi qaysi komponentalar palitrasiga kiradi?	*Win32	Standart	Additional	System
63	Object Inspector oynasining bo‘limlari nechta?	*2	1	3	4
64	C++ Builderda nechta forma qo‘llanilganini ko‘rish uchun klaviaturadan qaysi tugma orqali bajarish mumkin?	*F12	F8	F10	F5
65	LabeledEdit komponentasi qaysi komponentalar palitrasiga kiradi?	*Additional	Win32	Standart	System
66	C++ Builderda Object Inspector qaysi tugma orqali chiqadi?	*F11	F5	F9	F3
67	Label kompanentasi vazifasi nima?	*satr joylashtirish	kiritish qatori	tugma	ro`yxatdan tanlash
68	Edit kompanentasi vazifasi nima?	*kiritish qatori	tugma	ro`yxatdan tanlash	satr joylashtirish
69	Button kompanentasi vazifasi nima?	*tugma hosil qilish	ro`yxatdan tanlash	satr joylashtirish	ranglar jadvali
70	CheckBox kompanentasi vazifasi nima?	bayroqchali belgilash	ro`yxatdan tanlash	satr joylashtirish	ranglar jadvali
71	Bibtn bilan buttoning farqi nimada?	*dizaynda	amal bajarishda	rangida	tilida
72	Timer kompanentasi vazifasi nima?	*vaqt bilan ishlash	bayroqchali belgilash	ro`yxatdan tanlash	satr joylashtirish
73	MediaPlayer komponentasi qaysi komponentalar palitrasiga kiradi?	*System	Standart	Win32	Additional
74	ColorGrid kompanentasi vazifasi nima?	ranglar jadvali	bayroqchali belgilash	ro`yxatdan tanlash	satr joylashtirish
75	CCalendar komponentasi qaysi komponentalar palitrasiga kiradi?	*Samples	Additional	Win32	System
76	Kontekstli menyu yaratishda asosan qaysi komponentadad foydalaniladi?	*popupmenu	ColorGrid	Timer	Speedbutton
77	Formaning ixtiyoriy joyiga grafik tasvirni joylashtirish imkonini beradigan obyekt	*Image	Brush	Shape	Assigin
78	Tols-?	*servis xizmatidan foydalanish	yordam chaqirish.	formani ishga tushirish	kiritish qatori
79	Sarlavha matnini … aniqlaydi.	*Caption	Uses	Top va Lift	Width va Height
80	Forma balandligini o`zgartiring.	*Width	Caption	Name	Brush
81	Delphi loyiha fayl kengaytmasi	*dpr	pas	res	jpg

82	Delete funksiyaning qiymati qaysi tipga tegishli.	*String	Intejer	Real	Char
83	Fiksirlangan ustun rangi qanday nomlanadi?	* Fixed cols	Fixed Color	Key option	Strings
84	Fayl o‘zgaruvchisi bilan asosiy fayl orasida aloqa o‘rnatadigan funksiya.	*Assign	Label	Close	Type
85	Dinamik bog`lanuvchi bibliotekalar so`zining qisqartmasi.	*DLL	GLL	LLL	BLL
86	Ma`lum shartlarga muvofiq bajariladigan algoritm qanday.	*tarmoqlanuvchi	takrorlanuvch i	chiziqli	barchasi to`g`ri
87	Siljitish yo‘lchasiga ega panelni ko‘rsating.	*SpeedButton	BitBtn	ScrollBar	Memo
88	ListBox kompanentasi qaysi palitrasida joylashgan	*Standart	Additional	Win32	DataAccess
89	DrawGrid kompanentasi qaysi palitrasida joylashgan	*Additional	Standart	Win32	DataAccess
90	SpinEdit kompanentasi qaysi palitrasida joylashgan	*Samples	Additional	Win32	Standart
91	Belgi sifatida qanday sonlardan foydalaniladi?	* to`rtta raqamdan oshmaydigan ishorasiz butun son va lotin harflar	haqiqiy sonlar	to`rt xonali butun sonlar	hamma javob to`g`ri
92	O`zgaruvchilarni tasvirlash bo`limi qanday kalit so`z bilan boshlanadi	*VAR	CONST	TYPE	CASE
93	Ma'lumotlarning barcha turini nechtaga ajratish mukin?	*5 ta	3 ta	4 ta	2 ta
94	Ko`rsatilganlarning qaysi biri standart turga tegishli	* INTEGER, REAL, BOOLEAN, CHAR, STRING	PROGRAM, BEGIN, END	LABEL, CONST, VAR, TYPLE	INPUT, OUTPUT, WRITE
95	Massivlar, yozuvlar, to`plamlar va fayllar ma'lumotlarining qanday turi hisoblanadi?	*Murakkab turlar	standart turlar	oddiy turlar	hamma javob to`g`ri
96	Butun turdagi o`zgarmas deb nimaga aytiladi?	* Nuqtasiz yozilgan ixtiyoriy o`nli son	Ixtiyoriy manfiy son	Nuqtasiz yozilgan ixtiyoriy musbat son	Ixtiyoriy musbat son
97	Butun turdagi ma'lumotlar ustida qanday amallar bajarilganda butun natijalar olish mumkin?	*+,-,*, DIV,MOD	+,-,*,/	*,/, darajaga ko`tarish	+,-,kvadrat ildiz olish

98	Paskal tilida haqiqiy o`zgarmaslar necha xil ko`rinishda tasvirlanadi	*2 xil	3 xil	1 xil	5 xil
99	Quyidagi javoblarning qaysi birida qo`zg`almas konstanta ko`rsatilgan?	*45.6254	-0.2865E-5	5.28E6	-2.65E5
100	Mantiqiy ma'lumotlar turi tog`ri yozilgan javobni ko`rsating	* BOOLEAN	REAL	INTEGER	CHAR