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

NAVOIY DAVLAT PEDAGOGIKA INSTITUTI

INFORMATIKA O‘QITISH METODIKASI KAFEDRASI

DASTURLASH TILLARI FANIDAN

O`QUV-USLUBIY MAJMUA

BILIM SOHASI	**100000 – Gumanitar**
TA’LIM SOHASI	**110000 – Pedagogika**
TA’LIM YO`NALISHI	**5110700- Informatika o`qitish metodikasi**

Fan o'qituvchilari: **f.-m.f.n. G‘.R. Yodgorov**
o‘qit. F.J.Toxirov

Navoiy 2019

Ushbu o‘quv-uslubiy majmua 2019 yil 17-avgustda BD-5110700-2.05 bilan ro‘yxatdan o‘tkazilgan va O‘zbekiston Respublikasi Oliy va o‘rta maxsus ta’lim vazirligi tomonidan 2019 yil 29-iyunda tasdiqlangan “Dasturlash tillari” namunaviy fan dasturiga muvofiq ishlab chiqildi.

Tuzuvchilar:

Yodgorov G‘.R. – NavDPI, “Informatika o‘qitish metodikasi” kafedrasi mudiri, f.-m.f.n.,
Toxirov F.J. – NavDPI, “Informatika o‘qitish metodikasi” kafedrasi o‘qituvchisi.

Taqrizchilar:

O‘tapov T.U. - NavDPI, “Informatika o‘qitish metodikasi” kafedrasi dotsenti, p.f.n.

Xudoyorov Sh.J. - NavDPI, “Informatika o‘qitish metodikasi” kafedrasi dotsenti, f.-m.f.n.

Fanning o‘quv-uslubiy majmuasi “Informatika o‘qitish metodikasi” kafedrasining 2019-yil 27 avgustdagi 1 - son yig‘ilishida muhokamadan o‘tgan va fakultet kengashida muhokama qilish uchun tavsiya etilgan.

Kafedra mudiri:_____ f.-m.f.n. Yodgorov G‘.R.

Fanning o‘quv-uslubiy majmuasi “Fizika-matematika” fakultet kengashida muhokama etilgan va foydalanishga tavsiya qilingan (2019 yil 28 avgustdagi 1-sonli bayonnoma).

Fakultet kengashi raisi: _____dots. Kamolov I.R.

Navoiy davlat pedagogika institutining 2019 yil 30 – avgustdagi 1-sonli ilmiy uslubiy kengashida muhokama qilinib tasdiqlandi.

O‘quv uslubiy boshqarma boshlig‘i: _____ Xolmirzayev N.

MUNDARIJA

Ma’ruza matnlari

1-Ma’ruza

Mavzu: C++ tilining yaratilish tarixi va leksik asoslari. C++ tilida o‘zgaruvchilar, ma’lumotlar tiplari va arifmetik operatorlar

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.

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 kelmaydi. 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.

ESC (eskeyp) simvollar jadvali:

Yozilishi	Ichki kodi	Simvoli (nomi)	Ma’nosi
\a	0x07	bel (audible bell)	Tovush signali
\b	0x08	Bs (bascspase)	Bir qadam qaytish
\f	0x0C	Ff (form feed)	Sahifani qaytarish
\n	0x0A	lf (line feed)	Qatorni o’tkazish
\r	0x0D	Cr (carriage return)	Karetkani qaytarish
\t	0x09	Ht (horizontal tab)	Gorizontal tabulyatsi
\v	0x0B	Vt (vertical tab)	Vertikal tabulyatsi
\\	0x5C	\ (bacslash)	Teskari chiziq
\’	0x27	‘ (single out)	Apostrif (oddiy qavs)
\”	0x22	“ (double quote)	Ikkilik qavs
\?	0x3F	? (question mark)	Savol Belgisi
\000	000	Любой (octal number)	Simvol sakkizlik kodi
\xhh	0xhh	Любой (hex number)	Simvol o’n oltilik kodi

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.

CI ++ 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_kiyjmati.

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
- bo'lish	| 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

2-Ma’ruza

Mavzu: **C++ tilida tarmoqlanish operatorlari**

Reja:

1. if shart operatori.
2. Mantiqiy solishtirish operatorlari.

O‘tgan mavzularda misol tariqasida keltirilgan dasturlarda operatorlar yozilish tartibida ketma-ket va faqat bir marta bajarilgan holatlar, ya’ni chiziqli algoritmlar keltirilgan. Amalda esa kamdan–kam masalalar shu tariqa yechilishi mumkin. Aksariyat masalalar esa yuzaga keladigan turli holatlarga bog‘liq ravishda mos qaror qabul qilishni (yechimni) talab etadi. C++ tilida dasturning alohida bo‘laklarini bajarilish tartibini boshqarishga imkon beruvchi qurilmalarning yetarlicha katta majmuasiga ega. Masalan, dastur bajarilishining birorta qadamida qandaydir shartni tekshirish natijasiga ko‘ra dasturning u yoki bo‘lagiga boshqaruvni uzatish mumkin (tarmoqlanuvchi algoritm). Tarmoqlanishni amalga oshirish uchun shartli operatordan foydalaniladi.

IF operatori

if operatori qandaydir shartni rostlikka tekshirish natijasiga ko‘ra dasturda tarmoqlanishni amalga oshiradi:

if (<tekshiriladigan shart>)<operator>1;
<operator>2;

Bu yerda <tekshiriladigan shart> har qanday ifoda bo‘lishi mumkin, odatda u taqqoslash operatori bo‘ladi.

Agar tekshiriladigan shart rost (true) bo‘lsa, <operator>1 bajariladi, aks holda (false) dastur <operator>2 bajarishga o‘tadi.

C++ tilining qurilmalarida operatorlarni blok ko‘rinishida bo‘lishiga imkon beradi. Blok ‘{‘ va ‘}’ belgi oralig‘iga olingan operatorlar ketma-ketligi bo‘lib, u kompilyator tomonidan yaxlit bir operator deb qabul qilinadi.

Quyida keltirilgan dasturda if operatoridan foydalanish ko‘rsatilgan.

```
#include <iostream.h>
int main()
{
  int b;
  cin>>b;
  if (b>0)
    {                        // b>0 shart bajarilgan holat
       …
cout << “b – musbat son”;
...
        }
  if (b<0)
      cout <<“b – manfiy son”; // b < 0 shart bajarilgan holat
  return 0;
}
```

Dastur bajarilishi jarayonida butun turdagi b o‘zgaruvchi e’lon qilinadi va klaviaturadan qiymati kiritiladi. Keyin b qiymatini 0 sonidan kattaligi tekshiriladi, agar shart bajarilsa (true) ‘{‘ va ‘}’ belgilar ichidagi operatorlar bajariladi va ekranga “b – musbat son” xabari chiqadi. Agar shart bajarilmasa, bu operatorlar cheklab o‘tiladi. Navbatdagi shart operatori b o‘zgaruvchi qiymatini manfiylikka tekshiradi, agar shart bajarilsa yagona cout ko‘rsatmasi bajariladi va ekranga “b – manfiy son” xabari chiqadi.

IF … ELSE operatori

```
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 quyiladi.

```
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 quyidagicha

```
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”;
};
}
```

? : shart amali

Agar tekshirilayotgan shart nisbatan sodda bo’lsa, shart amalini «?: » ko’rinishini ishlatish mumkin. Bu operator quyidagi ko’rinishga ega:

<shart ifoda> ? <ifoda1> : <ifoda2>;

if shart operatoriga o’xshash holda bu shart amali quyidagicha ishlaydi: agar <shart ifoda> rost (true) bo’lsa <ifoda1> bajariladi, aks holda <ifoda2>. Odatda ifodalar qiymatlari birorta o’zgaruvchiga o’zlashtiriladi.

Misol: 2 ta sondan kattasini topuvchi dastur tuzilsin.

```
#include <iostream>
using namespace std;
int main()
{
  int a, b, max;
  cout << "a="; cin >> a;
  cout << "b="; cin >> b;
  max = ( a > b ) ? a : b;
  cout << max << endl;
  return 0;
}
```

Agar a>b shart bajarilsa max o'zgaruvchisi a ni, aks xolda b ni o'zlashtiradi.

Mantiqiy solishtirish operatorlari

C++ bir necha solishtirish operatorlariga ega.

Algebraik ifoda	C++ dagi operator	C++ dagi ifoda	Algebraik ma'nosi
tenglik guruhi			
=	==	x==y	x teng y ga
teng emas	!=	x!=y	x teng emas y ga
solishtirish guruhi			
>	>	x>y	x katta y dan
<	<	x<y	x kichik y dan
katta-teng	>=	x>=y	x katta yoki teng y ga
kichik-teng	<=	x<=y	x kichik yoki teng y ga

==, !=, >= va <= operatorlarni yozganda oraga bo'sh joy quyib ketish sintaksis hatodir. Yani kompilyator dasturdagi hatoni ko'rsatib beradi va uni tuzatilishini talab qiladi. Ushbu ikki belgili operatorlarning belgilarining joyini almashtirish, masalan <= ni =< qilib yozish ko'p hollarda sintaksis hatolarga olib keladi. Gohida esa != ni =! deb yozganda sintaksis hato vujudga ham, bu mantiqiy hato bo'ladi. Mantiqiy hatolarni kompilyator topa olmaydi. Lekin ular dastur ishlash mantig'ini o'zgartirib yuboradi. Bu kabi hatolarni topish esa ancha mashaqqatli ishdir (! operatori mantiqiy inkordir). Yana boshqa hatolardan biri tenglik operatori (==) va tenglashtirish, qiymat berish operatorlarini (=) bir-biri bilan almashtirib quyishdir. Bu ham juda ayanchli oqibatlarga olib keladi, chunki ushbu hato aksariyat hollarda mantiq hatolariga olib keladi.

Yuqoridagi solishtirish operatorlarini ishlatadigan bir dasturni ko'raylik.

//Mantiqiy solishtirish operatorlari

```
 # include <iostream.h>
int main()
{
int s1, s2;
cout << "Ikki son kiriting: " << endl;
cin >> s1 >> s2;                    //Ikki son olindi.
if (s1 == s2) cout << s1 << " teng " << s2 << " ga" << endl;
if (s1 < s2) cout << s1 << " kichik " << s2 << " dan" << endl;
if (s1 >= s2) cout << s1 << " katta yoki teng " << s2 << " ga" << endl;
if (s1 != s2) cout << s1 << " teng emas " << s2 << " ga" << endl;
return (0);
}
```

Ekranda:
Ikki sonni kiriting: 74 33
74 katta yoki teng 33 ga
74 teng emas 33 ga

Bu yerda bizga yangi bu C++ ning if (agar) struktura-sidir. if ifodasi ma'lum 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:

```
if (shart) ifoda;
```

Shart qismi qavs ichida bo'lishi majburiydir.Eng ohirida keluvchi nuqta-vergul (;) shart qismidan keyin quyilsa (if (shart); ifoda;) mantiq hatosi vujudga keladi. Chunki bunda if tanasi bo'sh qoladi. Ifoda qismi esa shartning to'g'ri-noto'g'ri bo'lishiga qaramay ijro qilaveradi.

C++ da bitta ifodani quyish mumkin bo'lgan joyga ifodalar guruhini ham quyish mumkin. Bu guruhni {} qavslar ichida yozish kerak. if da bu bunday bo'ladi:

```
if (shart) {
  ifoda1;
  ifoda2;
  ...
  ifodaN;
}
```

Agar shart to'g'ri javobni bersa, ifodalar guruhi bajariladi, aksi taqdirda blokni yopuvchi qavslardan keyingi ifodalardan dastur ijrosi davom ettiriladi.

3-Ma’ruza

Mavzu: **C++ tilida takrorlanish operatorlari**

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.

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’;
  }
  return 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;
 return 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.

4-Ma’ruza

Mavzu: **C++ tilida funksiyalar**.

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.

5-Ma’ruza

Mavzu: **C++ tilida bir o‘lchovli massivlar**.

Reja:

1. Massiv tushunchasi.
2. Massivlarni e’lon qilish.
3. Simvolli 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

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++;
```

6-Ma’ruza

Mavzu: **C++ tilida ikki o‘lchovli massivlar**.

Reja:

1. Ikki o`lchamli statik massivlarni e`lon qilish.
2. Massivlarga qiymat berish.
3. Satrli massivlar.
4. Ko`p o`lchovli massivlar.

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-Ma’ruza

Mavzu: **C++ tilida ko‘rsatkichlar va satrlar**. **C++ tilida strukturalar va birlashmalar.**

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..
6. C++ tilida strukturali tiplar va ularni tasvirlash.
7. C++ tilida strukturalarga murojaat.
8. C++ tilida strukturalar va massivlar
9. C++ tilida strukturalar va ko’rsatkichlar
10. C++ tilida birlashmalar

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.

Misol:

```
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

0123456789012345678 9

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 satr-nusxa 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[]=”0123456789012345678”;
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.

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 denominator;
} 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[20];
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_nomi);

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;
} complex;
void main()
{
complex 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;
```

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;
point++;
}
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.

Quyidagi dastur buferga kelib tushuvchi kodlarni ukib monitorga chiqarishga imkon beradi:

```
# include <iostream.h>
# include <bios.h>
void main()
{
union
{ char rr[2];
int ii;
} cc;
unsigned char scn,asc;
Cout<<(“\n\n Ctrl+Z bilan chikish.”
Cout<<(“\n Klavigani bosib, kodini oling. \n ”);
Cout<<(“\n    SCAN   ||   ASCII”);
Cout<<(“\n  (10) (16)    (10) (16)”);
do
{ Cout<<(“\n”);
cc.ii=bioskey(0);
asc=cc.hh[0];
scn=cc.hh[1];
Cout<<( “  %4d  %3xH || %4d %3xH |”,scn,scn,asc,asc);
}
while(asc!=26 || scn!=44);
}
```

Bu dasturda cc nomli birlashma kiritilgan bo’lib, cc.ii elementiga bioskey(0) funksiyasi natijasi yoziladi. So’ngra natijaning alohida baytlari sken va ASCII kodlar sifatida monitorga chiqariladi.

Tsikl to 26 ASCII kod va 44 sken kod paydo bo’lmaguncha (CTRL+Z klavishlari bosilmaguncha) davom etadi.

8-Ma’ruza

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. U N I X 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, formatlash-qatori, 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[BUFSIZ];
 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:

Long lseek (int fd, long offset, int origin);
Bu funksiya fd deskriptori bilan bog’lik fayldagi joriy pozitsiyani uchinchi parametr (origen) orqali nuqtaga nisbatan ikkinchi parametr (offset) qadamga ko’taradi. Boshlangich nuqta MS DOS da io.h yoki UNIX da unistd.h sarlavhali fayllarda aniqlangan konstantalar orqali aniqlanadi:

SEEK_SET (0 qiymatga ega) fayl boshi

SEEK_CUR (1 qiymatga ega) joriy pozitsiya

SEEK_END (2 qiymatga ega) fayl ohiri

Ko’chish davomida hato kelib chiqsa hato kodi errno global o’zgaruvchisiga yoziladi. Faylda joriy pozitsiyani aniqlash uchun tell ( ) funksiyasidan foydalaniladi:
Bu funksiya prototipi : long tell (int fd) ;
Joriy pozitsiyani fayl boshiga keltirish:
Lseek (fd, oh, seek_set)
Joriy pozitsiyani fayl ohiriga keltirish:

Lseek (fd, oh, seek_end)

9-Ma’ruza

Mavzu: **C++ tilining grafik imkoniyatlari**.

Reja:

1. C++ tilida simvolli kiritish va chiqarish.
2. C++ tilida ekran bilan ishlovchi funksiyalar.
3. C++ tilida grafik rejimda ekran bilan ishlash.
4. C++ tilida grafik sinflar.

Quyidagi funksiyalar dasturda simvollarni kiritish va chiqarish uchun ishlatiladi.

getch(arg) – bitta simvol kiritilishini kutish. Kiritilayotgan simvol monitorda aks etmaydi. Bu funksiyani programma oxirida argumentsiz ishlatilsa, monitorda ma’lumotlarni to klavisha bosilguncha o‘kish mumkin bo‘ladi.

putch(arg)- bitta simvolni standart oqimga chiqarish uchun ishlatiladi.Simvol monitorda aks etmaydi.

getchar(arg) – bitta simvol kiritilishini kutish. Kiritilayotgan simvol monitorda aks etadi. Bu funksiyani programma oxirida argumentsiz ishlatilsa, monitorda ma’lumotlarni to klavisha bosilguncha o’qish mumkin bo‘ladi.

putchar(arg)- bitta simvolni standart oqimga chiqarish uchun ishlatiladi. Simvol monitorda aks etadi.

Bu funksiyalar stdio.h modulida joylashgandir.

Misol:

```
#include <stdio.h>
int main()
{
int c;
c=getchar();
putchar(c);
getch();
return 0;
}
```

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.

Ekran bilan ishlashga misol.

Do‘stona funksiyaga ega bo‘lgan sinfga misol:

```
#include <conio.h>
class charlocus
{
int x,y;
char cc;
```

```
friend void friend_put (charlocus p, char c);
public:
charlocus (int xi, int yi, char ci)
{
x=xi; y=yi; cc=ci;
}
void display (void)
{
gotoxy(x,y); putch(cc);
}
};
void friend_put(charlocus p, char c)
{
p.cc=c;
}
int main ()
{
charlocus D(20,4, 'd');
charlocus S(10,10, 's');
clrscr( );
D.display( ); getch( ); S.display( ); getch( );
friend_put(D, 'x'); D.display( ); getch( );
friend_put(S, '#'); S.display( ); getch( );
return 0;
}
```

Dasturda D va S ob'ektlari yaratilib ular uchun ekranda kordinatalar va (d,s) simvollari aniqlanadi. So‘ng sinf funksiyasi charlocus : :

display ( ) simvollarni ko‘rsatilgan pozitsiyaga chiqaradi. Global friend_put funksiyasi simvollarning o‘rnini almashtirib qo‘yadi.

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;

Grafik sinfga misol. Misol tariqasida nuqta tushunchasini aniqlovchi point sinfini yaratib point.cpp fayliga yozib qo‘yamiz:

```
#ifndef POINTH
#define POINTH 1
class point
{
protected:
int x,y;
public:
point ( int xi=0, int yi=0);
int& givex(void);
int& givey (void);
void show(void);
void move(int xn=0, int yn=0);
private:
void hide();
}
#endif
```

Kelgusida point sinfini boshqa sinflarga qo‘shish mumkin bo‘lgani uchun shartli protsessor direktivasi #ifndef POINTH ishlatilgan. Protsessorli identifikator POINTH #define POINT 1 direktivasi orqali kiritilgan.Shuning uchun #include "point.h" direktivasi bir necha marta ishlatilganda xam POINT sinfi ta’rifi teksti faqat bir marta kompilyatsiya kilinaetgan faylda paydo bo‘ladi.

POINT sinfi komponenta funksiyalarini quyidagicha ta’riflaymiz:

```
ifdef POINTCPP
#define POINTCPP 1
#include <graphics.h>
#include "point.h"
point ::point(int xi=0, int yi=0)
{
x=xi; y=yi;
}
int &point::givex(void)
{
return x;
}
int &point::givey(void)
{
return y;
}
void point::show(void)
{
putpixel(x,y, getcolor());
}
void point::hide (void)
{
```

```
putpixel(x,y,getbcolor());
}
void point::move(int xn=0, int yn=0)
{
hide( );
x=xn; y=yn;
show( );
}
#endif
```

Kiritilgan point sinfi va komponenta funksiyalari bilan ishlovchi dasturni keltiramiz:

```
#include <graphics.h>
#include <conio.h>
#include "point.cpp"
int main()
{
point A(200,50);
point B;
point D(500,200);
int dr=DETECT, mod;
initgraph(&dr, &mod, "c:\\borlandc\\bgi");
A.show();getch();
B.show();getch();
D.show();getch();
A.move();getch();
B.move(50,60);getch();
closegraph();
return 0;
}
```

Qo‘shimcha yuklash. Misol tarikasida '+' amalini point sinfi bilan ta’riflanuvchi displeydagi nuqtalarga qo‘llaymiz. Soddalashtirish uchun point sinfida eng kerakli kompanentalarni qoldiramiz.

```
#include <graphics.h>
#include <conio.h>
class point
{
protected:
int x,y;
public:
point (int xi=0, int yi=0)
{
x=xi; y=yi;
}
void show (void)
{
putpixel (x,y,get color ( ) );
};
point operator+ (point p);
};
point point : : operator + (point &p)
```

```
{
point d;
d.x=this.x+p.x;
d.y=this.y+p.y;
return d;
}

int main()
{
int dr=DETECT, mod;
point A(200,50);
point B;
point D(50,120);
INITGRAPH (&dr, &mod, "C\\borlandc \\ bgi");
A.show ( );
getch ( );
B.show ( ); getch ( );
D.show ( ); getch ( );
B=A+D
;
B.show ( ); getch ( );
B=A.operator + (B);
B.show ( ); getch ( );
closegraph ( );
return 0;
}
```

Dastur bajarilishi natijasida displey ekraniga ketma-ket quyidagi nuqtalar qo‘yiladi: A(200,50); B(0,0); D(50,120); B(250,70), B(450,220)

Lokal sinf. Lokal sinflardan foydalanish xususiyatlarini tushuntirish uchun quyidagi masalani ko‘rib chiqamiz. «Kvadrat» sinfini aniqlash kerak bo‘lsin. Kvadrat tomonlarini koordinatalar o‘qiga parallel deb qaraymiz. Xar bir kvadrat berilganligi sifatida markaz koordinatalari va tomon uzunligi olinadi. Kvadrat sinfi ichida «kesma lokal»sinfini aniqlaymiz. Xar bir kesmani berilganlari sifatida uchlarining koordinatalarini olamiz. Uchlari mos ravishda olingan to‘rtta kesma kvadratni xosil qiladi.Shu usulda kvadrat ekranda tasvirlanadi.

```
#include<conio.h>
#include "point.cpp"
class square
{
class segment
{
point pn, pk;
public:
segment(point pin=point(0,0), point  pik=point(0,0))
{
pn.givex()=pin.givex();
pn.givey()=pin.givey();
pk.givex()=pik.givex();   pk.givey()=pik.givey();
}
```

```
point &beg(void)
{
return pn;
}
point &end (void)
{
return pk;
}
void showSeg()
{
line(pn.givex(), pn.givey(), pk.givex(),
pk.givey());
};
};
segment ab,bc,cd,da;
public:
square(point ci=point(0,0), int di=0)
{
point a,b,c,d;
a.givex()=ci.givex()-di/2;
a.givey()=ci.givey()-di/2;
b.givex()=ci.givex()+di/2;
b.givey()=ci.givey()-di/2;
c.givex()=ci.givex()+di/2;
c.givey()=ci.givey()+di/2;
d givex()=ci.givex()-di/2;
d.givey()=ci.givey()+di/2;
ab.beg()=a; ab.end()=b;
bc.beg()=b; bc end()=c;
cd.beg()=c; cd.end()=d;
da.beg()=d; da end()=a;
}
void showsquare(void)
{
ab.showSeg();
bc.showSeg();
cd.showSeg();
da.showSeg();
}
};

int main()
{
int dr=DETECT,mod;
initgraph(&dr,&mod,"c:\\borlondc\\ bgi");
point p1(80,120);
point p2(250,240);
square A(p1,30);
square B(p2,140);
A.showsquare(); getch();
B.showsquare(); getch();
closegraph();
```

```
return 0;
}
```

Vorislikka asoslanib grafik sinflarni yaratish

Grafik sinflarda vorislikkka misol. Keyingi misolda " ekrandagi nuqta" asosida " ekrandagi darcha " sinfi yaratiladi. Nasldan o‘tuvchi komponentalarga qo‘shimcha spot sinfiga quyidagi kompanentalarni kiritamiz: tasvir radiusi (rad); ekranda ko‘rinishi belgisi (vir=0 ekranda tasvir yuk; vi1==1 ekranda tasvir bor); tasvirni bitli matnda saqlash belgisi (tag=0 tasvir saqlanmaydi; tag==1 tasvir saqlanadi); tasvirni bitli matnda saqlash uchun ajratilgan xotira qismiga ko‘rsatgich pspot.

```
//Spot.cpp
#ifndef SPOT
#define SPOT  1
#include "point. cpp"
class spot: public point
{
int rad;
int vil;
int tag;
void * pspot;
public:
spot (int xi, int yi, int ri): point (xi, yi)}
{
int size;
vir =0; tag=0; rad=ri;
size=imagesize (xi-ri, yi-ri, xis- ri, yi- ri);
pspot=new char [size];
}
~ spot ()
{
hide();
tag =0;
delete[] pspot;
}
void show ()
{
if (tag==0)
{
circle (x, y, rad);
floodfill (x, y, getcolor ());
getimage (x-rad, y-rad, y+rad, pspot);
tag=1;
};
else
putimage (x-rad, y-rad, pspot, XOR_PUT);
vis=1;
}
void hide ()
{

if (vis==0) return;
```

```
putimage (x-rad,y-rad, pspot, XOR_PUT);
vis=0;
}
void move (int xn, int yn)
{
hide ();
x= xn; y=yn;
show ();
}
viod vary (float dr)
{
float a;
int size;
hide ();
tag=0;
delete pspot;
a=dr*rad;
if (a<=0) rad=0;
else rad= (int) a;
size=imagesize (x-rad; y-rad, x+rad, y+rad);
pspot=new char [size];
show ();
}
int& giver (void);
{
return rad;
}
};
# endif
```

Spot sinfida konstruktor, destruktor ~ spot () va beshta metod ko‘rsatilgan:

show ()-- ekranga doirani chizib, bitli tasvirni xotiraga olish;

hide ()-- ekrandan doira tasvirini uchirish;

move ()--tasvirni ekranning bitta joyiga kuchirish;

vary ()--ekrandagi tasvirni uzgartirish (kichkinalashtirish yoki kattalashtirish);

giver () --doira radiusiga murojatni ta’minlash;

point sinfidan spot sinfi naslga nuqta markazi (x,u) koordinatalarini va givex, givey metodlarni oladi, point : : show () va point : : move () metodini xuddi shu nomli yangi funksiyalar bilan almashtirilgan point :: hide() funksiyasi nomi o‘tmaydi, chunki point sinfida u xususiy (private) statusiga ega. spot() konstruktori uch parametrga ega - markaz koordinatalari (xi,yi) va doira radiusi (ri).

Avval point sinfi konstruktori chaqiriladi bu konstruktor xi,yi ga mos keluvchi xaqiqiy parametr asosida doira markazini aniklaydi. Asosiy sinf konstruktori xar doim xosilaviy sinf

konstruktoridan oldin chaakiriladi. Sungra spot() sinfi konstruktolari boshlanadi. Bu konstruktor vis, tag parametrlarining boshlangich qiymatini aniklaydi va ri ga mos keluvchi xaqiqiy parametr qiymati asosida doira radiusi red aniklanadi. Standart funksiya imagesize yordamida doira joylashuvchi kvadratik operativ xotirada aniqlash uchun zarur bulgan xotira xajmi xisoblanadi. Kerakli xotira new standart operatsiya yordamida ajratib size elementidan iborat char massivlar yoziladi. Agar aytilgan xotira spot sinfida protected statutisiga ega bulgan pspot ko‘rsatkichiga ulanadi.

Vorislikda destruktorlar xossalari. Sinfning xar bir ob'ekti yaratilganda sinf konstruktori chaqirilib, ob'ekt uchun kerakli xotira yaratish va initsializatsiya qilish vazifalarini bajaradi. Ob'ekt yukotilganda yoki sinf ta'sir doirasidan tashqariga chiqilganda teskari vazifalar bajarish kerak bo'lib, bular ichida eng keraklisi xotirani ozod kilishdir. Bu vazifalarni boshkarish uchun sinfga maxsus funksiya destruktor kiritiladi. Destruktor quyidagi shaklga ega bo'lgan anik nomga ega ~ sinf-nomi.

Destruktor xatto void tipidagi parametrlarga ega bo'lmaydi va xatto void tipidagi qiymat qaytarmaydi. Destruktor statusi aloxida e'lon qilinmagan bo'lsa umumiydir Sodda sinflarda destruktor avtomatik aniklanadi, misol uchun paint sinfida destruktor e'lon kilinmagan va kompilyator quyidagi destruktorni avtomatik chakiradi

~ point () {};

spot sinfida destruktor aniq ko'rinishga ega;

~ spot () {hide (); tag=0;delete [] pspot;}

Bu destruktor vazifalari doira tasvirini spot::hide() funksiyasi orkali o'chirish; tag belgisiga 0 qiymatini berish; ob'ekt bitli tasvirni saklash uchun ajratilgan xotirani tozalash.

Destruktorlar naslga o'tmaydi, shuning uchun xosilaviy sinfda destruktor mavjud bo'lmasa asosiy sinfdagi destruktor chaqirilmaydi. Balki kompilyator tomonidan yaratiladi. Ko'rilayotgan misolda quyidagicha: public: ~spot () {~point ();}

Asosiy sinflar destruktorlar ro'yxatda ko'rsatilganidek teskari tartibda boshqariladi. Shunday qilib ob'ektlarni o'chirish tartibi yaratilish tartibiga teskaridir. Agar ob'ekt yaratilganda dasturda xotira ajratilgan bulsa destruktor dasturda chaqirilishi lozim.

Spot sinfi ob'ektlari bilan ishlovchi dasturni keltiramiz:

```
#include<graphics.h>
#include<conio. h >
#include "spot. cpp"
int main()
{
int dr=DETECT, mod;
initgraph (&dr, &mod);
{
spot A(200,50,20);
spot D(500,200,30);
A.show();
getch ();
D.show ();
getch();
A.move(50,60);
getch ();
}
closegraph();
return 0;
}
```

Grafik abstrakt sinf va uning vorislariga misol. Quyidagi misolda abstrakt sinflar umumiy tushunchalarni tavsiflash uchun ishlatiladi point sinfi vorisi sifatida figure abstrakt sinfi yaratiladi. Bu sinfda konstruktor, sof virtual funksiya show (), xamda hide() va move() metodlari aniqlangan. Dasturda figure sinfi asosida ikkita avlod sinf circle (aylana) va ellips (elips)sinflari aniqlanadi.

Ikkala sinfda point sinfidan shakllar markazlari koordinatalari naslga o'tgan. Ikkala sinfda konkret show() metod aniqlangan va figure () abstrakt sinfidan move() va hide() funksiyalari naslga o'tgan.

```
// figure.cpp abstrakt sinf
# include "point.cpp"
class figure: public point
{
public :
//figure  sinf konstruktori
figure (point p) :point (p.give x(), p.give y()){}
// sof virtual funksiya
virtual void show() = 0;
// figurani o‘chirish funksiyasi
void hide()
{
int bk,cc;
bk = getbkcolor();
cc = getcolor();
setcolor(bk);
show();
setcolor(cc);
}
// figurani xarakatlantirish funksiyasi
void move (point p)
{
hide ();
x=p.givex();y=p.givey();
show();
}
};
```

figure abstrakt sinf asosida konkret sinflar yaratamiz;

```
// ELLIPS CPP
class ellips: public figure
{
int rx, ry;
public:
// konstruktor
ellips(point d, int radx, int rady): figure (d)
{
rx = radx; ry = rady;
}
void show()
{
ellipse (x,y,0, 360, rx, ry);
return;
}
};

// circ.fig  aylana sinfi
class circ: public figure
{
int radius;
public:
// konstruktor
```

```
circ (point e, int rad): figure (e)
{
radius=rad;
}
void show () {
circle(x,y,radius);
}
};
```

quyidagi dasturda uchta sinf xammasi ishlatilgan;

```
# include <graphics.h>
# include "figure.cpp"
# include "circ.fig"
# include "ellips.fig"
# include <conio.h>


int main ()
{
point A(100,80), B(300,200);
circ C(A,60);
ellips E(B,200,100);
{
int dir = DETECT, mod;
initgraph (&dir, &mod, "c\\borlandc\\ bgi");
A.show (); getch();
B.show (); getch();
C.show (); getch();
E.show (); getch();
C.move(B); getch();
E.hide(); getch();
C.hide(); getch();
}
closegraph();
return 0;
}
```

Amaliy mashg‘ulotlar

1-Amaliy mashg‘ulot.
Mavzu: C++ da ma'lumotlarning asosiy turlari. C++tilida 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
4
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;
}
```

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.

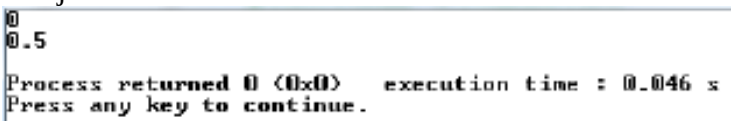
```
#include <iostream>
using namespace std;
int main()
{
    float haqiqiy = 5.57;
int oshkor, oshkormas;
     // oshkormas ravishda butun toifaga o'tish
    oshkormas = haqiqiy;
    oshkor = (int) haqiqiy; // oshkor holda butun toifaga o'tish
     cout << "haqiqiy = " << haqiqiy << endl;
    cout << "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:

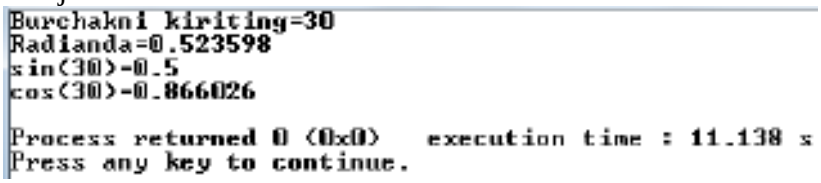


6 - misol. Trigonometrik funksiyalar bilan ishlash

```
#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:



2-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;
 }
```

3-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

4-Amaliy mashg‘ulot.

Mavzu: C++ tilida bir o'lchovli 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);
}
```

5-Amaliy mashg‘ulot.

Mavzu: 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;
}
```

6-Amaliy mashg‘ulot.

Mavzu: C++ tilida funksiyalar yaratish. Strukturalar va birlashmalar. C++ tilida ko'rsatkichlar. C++tilida dinamik massivlar.

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 sinflar.

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. Element-nomi

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

Fizika-matematika fakulteti 5110700-Informatika o`qitish metodikasi ta`lim yo`nalishi talabalari uchun mo`ljallangan.

Tuzuvchilar: f.-m.f.n. Yodgorov G‘.R.
o`q. Toxirov F.J.

Taqrizchi: f.-m.f.n. Ro‘ziyev R.A.
f.-m.f.n. Xudoyorov Sh.J.

KO‘RSATMA:
LABORATORIYA ISHLARINI TANLASH VA TOPSHIRISH TARTIBI

Laboratoriya ishi variantlari talabalarning guruh jurnalidagi tartib raqamiga mos ravishda tanlanadi. Mos variantni tanlagach, savollarga javob berish uchun tayyorlanadi. Savollarga tayyor bo‘lgan talaba navbati bilan o‘qituvchiga topshiradi.

Laboratoriya ishlanmasi Informatika o‘qitish metodikasi kafedrasining yig’ilishida ko’rib chiqilgan va o`quv jarayonida foydalanish uchun tavsiya etilgan (2019-yil 28 -avgust 1-son yig’ilish bayoni).

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:

```
0
0.5

Process returned 0 (0x0)   execution time : 0.046 s
Press any key to continue.
```

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:

```
Burchakni kiriting=30
Radianda=0.523598
sin(30)=0.5
cos(30)=0.866026

Process returned 0 (0x0)   execution time : 11.138 s
Press any key to continue.
```

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 ketma-ketligi 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 ketma-ketligi 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 ketma-ketligi 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 ketma-ketligi 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. Element-nomi

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;

Foydalanilgan adabiyotlar

Asosiy adabiyotlar

1. A. R. Azamatov, B. Boltayev. Algoritmlash va dasturlash asoslari. 0‘quv qo‘llanma. T. : “Cho‘lpon”, 2010 y.
2. A. R. Azamatov, B. Boltayev. Algoritmlash va dasturlash asoslari. 0‘quv qo‘llanma. T. : “Cho‘lpon”, 2013 y.
3. M.A.Aripov, N.A.Otaxonov. Dasturlash asoslari. O`quv qo`llanma. T.: “Tafakkur bo`stoni”, 2015 yil.
4. М. Ашуров, М. Мирмахмудов, Ш. Сапаев. Замонавий дастурлаш тиллари фанидан лаборатория ишлари. Т. : ТДПУ, 2008 й.
5. Меняев Михаил Федорвич. Информационные технология управления. Москва, «Издательский Омегал», 2003 г.
6. S.Tursunov, I.Nazarov. Ta’limda axborot texnologiyalari. Darslik Toshkent. “Adabiyot uchqunlari” 2019-yil 1 tom, 262-b.
7. S.Tursunov, I.Nazarov. Ta’limda axborot texnologiyalari. Darslik Toshkent. “Adabiyot uchqunlari” 2019-yil 1 tom, 300-b.

Qo‘shimcha adabiyotlar

1. Мирзиёев Шавкат Миромонович. Эркин ва фаровон, демократик Узбекистан давлатини биргаликда барпо этамиз. Узбекистан Республикаси Президента лавозимига киришиш тантанали маросимига батишланган Олий Мажлис палаталарининг кушма мажлисидаги нутк / Ш.М. Мирзиёев. - Тошкент : Узбекистан, 2016. - 56 б.
2. Мирзиёев Шавкат Миромонович. Танкидий тахдил, катьий тартиб-интизом ва шахсий жавобгарлик - хар бир рахбар фаолиятининг кундалик коидаси булиши керак. Мамлакатимизни 2016 йилда ижтимоий-иктисодий ривожлантиришнинг асосий якунлари ва 2017 йилга мулжалланган иктисодий дастурнинг энг мухим устувор йунапишларига батишланган Вазирлар Махкамасининг кенгайтирилган мажлисидаги маъруза, 2017 йил 14 январ / Ш.М. Мирзиёев. - Тошкент : Узбекистан, 2017. - 104 б.
3. Мирзиёев Шавкат Миромонович. К,онун устуворлиги ва инсон манфаатларини таъминлаш - юрт тараккиёти ва халк фаровонлигининг гарови. Узбекистан Республикаси Конституцияси кабул килинганининг 24 йиллигига багишланган тантанали маросимдаги маъруза. 2016 йил 7 декабр /Ш.М.Мирзиёев. - Тошкент: “Узбекистан”, 2017. - 48 б.
4. Мирзиёев Шавкат Миромонович. Буюк келажагимизни мард ва олижаноб халкимиз билан бирга курамиз. Мазкур китобдан Узбекистан Республикаси Президенти Шавкат Мирзиёевнинг 2016 йил 1 ноябрдан 24 ноябрга кадар Коракалпогистон Республикаси, вилоятлар ва Тошкент шахри сайловчилари вакиллари билан **утказилган** сайловолди учрашувларида сузлаган нуткдари урин олган. /Ш.М.Мирзиёев. - Тошкент: : “Узбекистан”, 2017. - 488 б
5. Узбекистан Республикаси Президентининг Фармони. Узбекистан республикасини янада ривожлантириш буйича харакатлар стратегияси тугрисида. *(Узбекистон Республикаси цонун уужжатлари туплами, 2017 й., 6- сон, 70-модда)*
6. Узбекистон Республикаси Конституцияси. Т.: Узбекистон. 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.
15. The C++ Programming Language, Third Edition by Bjarne Stroustrup. AT&T Labs. Murray Hill, New Jersey.
16. Культин Н.Б. C++ Builder в задачах и примерах.- СПб.: БХВ- Петербург, 2005. — 336 с: ил.

Elektron ta’lim resurslari

1. www.zivonet.iiz - **Axborot ta’lim portal**
2. www.edu.uz - **Oliy va o‘rta maxsus ta’lim vazirligi portal**
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.timus.ru/ - dasturlarni testlovchi tizim.

Mustaqil ta'lim uchun tavsiya etiladigan mavzular

- Delphi bilan tanishish
- Delphi qoMlanaladigan 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 proseduralami 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.

Kurs ishi uchun taxminiy mavzular

1. Dasturlash tillari va ularning turlari
2. Ob’ektga yo‘naltirilgan dasturlash tillari va ulaming turlari.
3. Delphi ob’ektga yo‘naltirilgan dasturlash tili va uning imkoniyatlari.
4. Delphi dasturlari strukturasi. Loyiha va modul.
5. Delphi dasturlash tilining operatorlari va ularga doir misollar.
6. Delphi dasturlash tilida prosedura va funksiyalar
7. Delphi dasturlash tilining grafik vositalari.
8. Amaliy masalalarga loyihalar tuzish va ulardan foydalanish
9. Delphi dasturlash tilining grafik imkoniyatlaridan foydalanib geometrik figuralarni hosil qilish
10. C++ tilining boshqarish operatorlari (if, for, While, Do-while)
11. C++ tilida funksiyalar, strukturalar va birlashmalar
12. C++ tilida ko‘rsatkichlar va murojaatlar
13. C++ tilida bir o‘lchovli, ikki o‘lchovli va dinamik massivlar.
14. C++ tilida satriy kattaliklar
15. C++ tilida sinflar
16. C++ tilida grafika
17. C++ tilida multimedia va animatsiyalar
18. C++ tilida fayllar bilan ishlash. Dialog oynalari.
19. C++ tilida panel va menyu yaratuvchi komponentlar.
20. C++ tilining multimedia imkoniyatlari.

Ilovalar

Fan dastur

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" 08

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

Ushbu fan dasturi 5110700 – Informatika o‘qitish metodikasi bakalavriat ta’lim yo‘nalishi talabalari uchun mo‘ljallangan bo‘lib, Dasturlash tillari fanining fan dasturida dasturlash tillariga kirish, dasturlash tillari va ularning klassifikasiyasi, dasturlash tilining alifbosi, dasturlash tilining ishchi muhiti, buruqlar tizimi va operatorlari, ob’yektga yo‘naltirilgan dasturlash tillari, ob’yektga yo‘naltirilgan loyihalash, muayyan dasturlash tilida chiziqli, tarmoqlanuvchi va takrorlanuvchi algoritmlarga mos dasturlar tuzishning asosiy nazariya hamda amaliy tomonlari ma’lum izchillikda bayon qilingan.

“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

Fanning maqsadi – bo‘lajak informatika o‘qituvchilariga dasturlash tillarining eng muhim bo‘lgan ilmiy-nazariy asoslari va amaliy jihatlarini chuqur o‘rgatish, Davlat ta`lim standarti va malaka talabalariga javob beradigan bilimlar berish, dasturlash tillari yo‘nalishida innovatsion g‘oyalarni yaratishga bo‘lgan qiziqishlarini oshirish, informatika o‘qituvchisining dasturlashga oid kasbiy kompetensiyalarni shaklantirish hamda rivojlantirishdan iborat.

Fanning vazifalari - talabalardan Kadrlar tayyorlash milliy dasturi asosida shuningdek, mamlakatimizda axborot kommunikatsiya-texnologiyalari sohasini yanada rivojlantirish, talaba-yoshlarni dasturlash tillari, IT sohasida innovatsion g‘oyalarni yaratishga bo‘lgan qiziqishlarini oshirish, ob‘yektga yo‘naltirilgan dasturlash tillarining nazariy asoslarini bilish, ob‘yektga yo‘naltirilgan muhitlarda xabarlarni uzatish, ularga ishlov berish, ob‘yektlar iyerarxiyasi asosida dasturlarni loyihalash, muayyan ob‘yektga yo‘naltirilgan muhitlarda chiziqli, tarmoqlanuvchi va takrorlanuvchi va modulli dasturlar tuza olish, loyihalash va ulardan foydalana olish, masalalarni tahlil qila olish, masalalarga mos tuzilgan dastur va natijalarni taqqoslay olish ko‘nikma va malakalariga ega bo‘lish talab etiladi.

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 ***bilimga;***

– 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 loyihalash ***ko‘nikmasiga;***

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

III. Asosiy nazariy qism (ma‘ruza mashg‘ulotlari)

1-MODUL. OB’YEKTGA 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’yektga yo‘naltirilgan dasturlash tillari.

Ob’yektga yo‘naltirilgan dasturlash tillari. Dasturlashning ob’yektga yo‘naltirilgan paradigmasi.

3-mavzu. Ob’yektga yo‘naltirilgan loyihalash.

Ob‘ektlarni loyihalash: satrlar, steklar, ro‘yxatlar, navbatlar, daraxtlar. Matematik ob’yektlar: rasional va kompleks sonlar, vektorlar, matrisalar. Ob’yektlar kutubxonasi. Interfeys ob’yektlari: boshqarish elementlari, oynalar, dialoglar.

4-mavzu: Voqealar va habarlar.

Voqealar va habarlar. Ob’yektga yo‘naltirilgan muhitlarda habarlarni uzatish va ularga ishlov berish mexanizmlari. Ob’yektlar iyerarxiyasi asosida dasturlarni loyihalash. Muayyan ob’yektga 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’yektlarning daraxtsimon ko‘rinish oynasi, ob’yektlar 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 xossalari.

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. Funksiyalarrning 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.

22-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 nushalash, 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. 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.

6-MODUL. VIZUAL DASTURLAR TUZISH

26-mavzu. Borland C++ Builder dasturlash muhiti.

Borland C++ Builder dasturlash muhitiga kirish, ishchi muhit, oynalar. Borland C++ Builder komponentlari va ularning hossalari. Komponentlar hodiasalari 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 operatorlar.

Matematik funktsiyalar va doimiyliklardan foydalanish. Amallar va ularning bajarilish tartiblari. Mantiqiy amallar. If va Switch operatorlari. For sikli va uning qo'llanilishi. While va do_while sikllari. Ichma-ich joylashgan sikllar.

29-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.

30-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.

31-mavzu. Borland C++ Builder Additional komponentlar palitrasi.

BitBtn, MaskEdit, StringGrid, MaskEdit, CheckListBox, DrawGrid, Image, Shape va boshqa komponentlaridan foydalanish. Komponentlar xossalari.

32-mavzu. Borland C++ Builder Dialogs komponentlar palitrasi.

Borland C++ Builderdagi Dialogs komponentlar palitrasi. OpenDialog, SaveDialog, FindDialog, ColorDialog, FontDialog va hokazo komponentlar va ularning xossalari.

33-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.

34-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.

35-mavzu. Borland C++ Builderning grafik vositalari.

Tayyor grafik fayllardan foydalanish. Grafik fayllar formati. Image Editor grafik muharriri. Tugmalar uchun piktogrammalar yaratish. 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.

36-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 va sinflar
21. C++ tilida multimedia va animatsiyalar
22. Borland C++ Builderda formalar bilan ishlash
23. Borland C++ Builderda Standard bo'limi komponentlari va ularning xossalari
24. Borland C++ Builderda Additional bo'limi komponentlari va ularning xossalari
25. Borland C++ Builderda Edit, LabeledEdit va MaskEdit komponentlaridan foydalanish
26. Borland C++ Builderda Button, BitBtn va SpeedButton komponentlari
27. Borland C++ Builderda sikllar bilan ishlash
28. Borland C++ Builderda massivlar bilan ishlash
29. Borland C++ Builderda funksiya va protseduralar bilan ishlash
30. Borland C++ Builderda fayllar bilan ishlash
31. Borland C++ Builderda ko'p formali ilovalar yaratish
32. Borland C++ Builderda sana va vaqt bilan ishlash

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 chiziqli dasturlar tuzish
24. C++ dasturlash tilida If/Else va Switch strukturasi bilan ishlash
25. C++ dasturlash tilida While, Do While va For takrorlash strukturasi bilan ishlash
26. Boshqaruv strukturalarida Continue va Break ifodalarini qo'llash

27. C++ dasturlash tilida funktsiyalar yaratish va ulardan foydalanish
28. C++ dasturlash tilida massivlar bilan ishlash
29. C++ dasturlash tilida ko'rsatkichlar bilan ishlash
30. C++ dasturlash tilida satriy kattaliklar bilan ishlash
31. C++ dasturlash tilida fayllar bilan ishlash.
32. C++ tilida panel va menyu yaratuvchi komponentlar
33. Borland C++ Builderda Label, Edit va Button kompenentlaridan foydalanib dastur tuzish
34. Borland C++ Builderda tarmoqlanuvchi dasturlar tuzish
35. Borland C++ Builderda While siklik operatoridan foydalanib dastur tuzish
36. Borland C++ Builderda Do..While siklik operatoridan foydalanib dastur tuzish
37. Borland C++ Builderda For siklik operatoridan foydalanib dastur tuzish
38. Borland C++ Builderda sichqoncha va klaviatura xodisalarini qayta ishlash
39. Borland C++ Builderda xabarlar oynasi bilan ishlash protsedura va funktsiyalari
40. Borland C++ Builderda CheckBox, RadioGroup, ComboBox va ListBox kompenentlaridan foydalanib dastur tuzish
41. Borland C++ Builderda bir o'lchovli massivlarga doir dastur tuzish
42. Borland C++ Builderda ikki o'lchovli massivlarga doir dastur tuzish
43. Borland C++ Builderda funksiyalarga doir dastur tuzish
44. Borland C++ Builderda satriy kattaliklar
45. Borland C++ Builderda fayllar bilan ishlash.
46. Borland C++ Builderda grafikaga doir dastur tuzish
47. 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'yektga yo'naltirilgan dasturlash tillari.
4. Ob'yektga yo'naltirilgan loyihalash.
5. Ob'yektlar 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 dasturlar tuzish
10. Delphi dasturlash tilida For sikl operatoridan foydalanish
11. Delphi dasturlash tilida While..Do sikl operatoridan foydalanish
12. Delphi dasturlash tilida Repeat..Until sikl operatoridan foydalanish
13. Delphi dasturlash tilida tasodifiy sonlar bilan ishlash

14. Delphi dasturlash tilida bir o‘lchovli massivlar bilan ishlash
15. Delphi dasturlash tilida ikki o‘lchovli massivlar bilan ishlash
16. Delphi dasturlash tilida sana-vaqt turi bilan ishlash
17. Delphi dasturlash tilida satriy kattaliklar bilan ishlash
18. Delphi dasturlash tilida to‘plamlar bilan ishlash
19. Delphi dasturlash tilida fayllar bilan ishlash
20. Delphi dasturlash tilida funksiyalar yaratish
21. Delphi dasturlash tilida proseduralar yaratish
22. Delphi dasturlash tilida grafik primitivlar bilan ishlash
23. Delphi dasturlash tilida ListBox da grafiklarni joylashtirish
24. Biror predmet sohasiga oid o‘rgatuvchi dasturlar yaratish
25. Delphi dasturlash tilida nazorat qiluvchi dasturlar yaratish
26. Delphi dasturlash tili komponentlar palitrasi bo‘limlari va ayrim komponentlar xossalari.
27. Delphida forma xossalari va ularni o‘zgartirish.
28. Delphi dasturlash tilida grafika.
29. Delphi dasturlash tilining multimedia imkoniyatlari.
30. C++ tilida funksiyalar, strukturalar va birlashmalar.
31. C++ tilida ko‘rsatkichlar va murojaatlar
32. C++ tilida bir o‘lchovli, ikki o‘lchovli va dinamik massivlar.
33. C++ tilida satriy kattaliklar
34. C++ tilida sinflar.
35. C++ tilida grafika
36. C++ tilida multimedia
37. C++ tilida animatsiyalar
38. C++ tilida fayllar bilan ishlash.
39. C++ tilida Dialog oynalari.
40. C++ tilida panel yaratuvchi komponentlar.
41. C++ tilida menyu yaratuvchi komponentlar.
42. C++ tilidagi dasturlarning tarkibiy qismlari.
43. C++ da maxsus belgilar.
44. O‘zgarmaslar. Literal o‘zgarmaslar.
45. Belgili o‘zgarmaslar.
46. Ifodalar va operatorlar.
47. Matematik operatorlar. Operatorlar prioriteti.
48. Increment va decrement operatorlari
49. Ishorali va ishorasiz tiplar. O‘zgaruvchilarning tayanch tiplari
50. Matematik kutubhona funksiyalari
51. Funksiyalarning tuzilishi

52. Funksiyalarning qo‘llanilishi.
53. Borland C++ Builderda Animation va MediaPlayer komponentlarida foydalanish
54. Borland C++ Builderda Chart va VtChart komponentlarida foydalanish
55. Borland C++ Builderda ColorDialog va ColorBox komponentlari
56. Borland C++ Builderda Data Access komponentlar palitrasi
57. Borland C++ Builderda grafik axborotlar bilan ishlovchi komponentlar
58. Borland C++ Builderda Image va PaintBox komponentlarida foydalanish
59. Borland C++ Builderda PrintDialog va PrintSetupDialog komponentlari
60. Borland C++ Builderda System komponentlar palitrasidan foydalanish
61. Borland C++ Builderda Win32 komponentlar palitrasidan foydalanish
62. Borland C++ Builderda xossa va metodlar
63. Borland C++ Builderda Additional bo‘limi komponentlari va ularning xossalari
64. Borland C++ Builderda BorderStyle xususiyati
65. Borland C++ Builderda Button, BitBtn va SpeedButton komponentlari
66. Borland C++ Builderda Dialog bo‘limi komponentlari va ularning xossalari
67. Borland C++ Builderda Edit, LabeledEdit va MaskEdit komponentlaridan foydalanish
68. Borland C++ Builderda fayllar bilan ishlash
69. Borland C++ Builderda formalar bilan ishlash
70. Borland C++ Builderda funksiya va protseduralar bilan ishlash
71. Borland C++ Builderda istisnolar. Xatolarni qayta ishlash
72. Borland C++ Builderda ko‘p formali ilovalar yaratish
73. Borland C++ Builderda massivlar bilan ishlash
74. Borland C++ Builderda massivlarni saralash
75. Borland C++ Builderda Memo va RichEdit komponentlari
76. Borland C++ Builderda sikllar bilan ishlash
77. Borland C++ Builderda Standard bo‘limi komponentlari va ularning xossalari
78. Borland C++ Builderda System32 bo‘limi komponentlari va ularning xossalari
79. Borland C++ Builderda xabarlar oynasi bilan ishlovchi funktsiyalar va protseduralar
80. Borland C++ Builderning Excel dasturi bilan hamkorligi

Mustaqil o‘zlashtiriladigan mavzular bo‘yicha talabalar tomonidan referatlar tayyorlash va uni taqdimot qilish tavsiya etiladi.

VII. Asosiy va qo‘shimcha o‘quv adabiyotlar hamda axborot manbalari

Asosiy adabiyotlar:

1. A.R.Azamatov, B.Boltayev. Algoritmlash va dasturlash asoslari. O‘quv qo‘llanma. T.: “Cho‘lpon”, 2010 y.
2. A.R.Azamatov, B.Boltayev. Algoritmlash va dasturlash asoslari. O‘quv qo‘llanma. T.: “Cho‘lpon”, 2013 y.
3. M.M.Aripov, N.A.Otaxanov. Dasturlash asoslari. O‘quv qo‘llanma. T.: “Tafakkur bo‘stoni”, 2015 y.
4. М.Ашуров, М.Мирмахмудов, Ш.Сапаев. Замонавий дастурлаш тиллари фанидан лаборатория ишлари. Т.: ТДПУ, 2008 й.
5. Меняев Михаил Федорвич. Информационные технология управления. Москва, «Издательский Омегал», 2003 г.
6. S.Tursunov, I.Nazarov. Ta’limda axbotor texnologiyalari. Darslik Toshkent: “Adabiyot uchqunlari”, 2019. 1-tom, -262 b.
7. S.Tursunov, I.Nazarov. Ta’limda axbotor texnologiyalari. Darslik Toshkent: “Adabiyot uchqunlari”, 2019. 2-tom, -300 b.

Qo‘shimcha adabiyotlar

8. 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. – Т.: Ўзбекистон, 2016. - 56 б.
9. Мирзиёев Шавкат Миромонович. Танқидий таҳлил, қатъий тартиб-интизом ва шахсий жавобгарлик – ҳар бир раҳбар фаолиятининг кундалик коидаси бўлиши керак. Мамлакатимизни 2016 йилда ижтимоий-иқтисодий ривожлантиришнинг асосий якунлари ва 2017 йилга мўлжалланган иқтисодий дастурнинг энг мухим устувор йўналишларига бағишланган Вазирлар Маҳкамасининг кенгайтирилган мажлисидаги маъруза, 2017 йил 14 январь. /Ш.М. Мирзиёев. – Т.: Ўзбекистон, 2017. – 104 б.
10. 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 декабрь. /Ш.М.Мирзиёев. – Т.: “Ўзбекистон”, 2017. – 48 б.
11. Мирзиёев Шавкат Миромонович. Буюк келажагимизни мард ва олижаноб халкимиз билан бирга қурамиз. Мазкур китобдан Ўзбекистон Республикаси Президенти Шавкат Мирзиёевнинг 2016 йил 1 ноябрдан 24 ноябрга кадар Қоракалпоғистон Республикаси, вилоятлар ва Тошкент шахри сайловчилари вакиллари билан ўтказилган сайловолди учрашувларида сўзлаган

нуткларн ўрин олган. /Ш.М.Мирзиёев. – Т.: "Ўзбекистон", 2017. – 488 б

12. Ўзбекистон Республикаси Президентининг Фармони. Ўзбекистон республикасини янада ривожлантириш бўйича харакатлар стратегияси тўғрисида. (Ўзбекистон Республикаси конун хужжатлари тўплами, 2017 й., 6-сон, 70-модда)

13. Ўзбекистон Республикаси Конституцияси. – Т.: Ўзбекистон. 2014 й. -46 б.

14. Nazirov Sh.A., Kabulov R.V., Babajanov M.R., Raxmanov Q.S. C va C++ tili. "Voris-nashriyot", Toshkent, 2013 y. 488 b.

15. Nazirov Sh.A., Musayev M.M., Ne'matov A.N., Qobulov R.V., Delphi tilida dasturlash asoslari. "G'.G'ulom nomidagi nashriyot-matbaa ijodiy uyi", Toshkent 2008 y. 280 b.

16. Дарахвелидзе П., Марков Э.. Программирование в Delphi7. Учебник. Санкт-Петербург. "БХВ-Петербург" 2003 г.

17. Фаронов В. В. Программирование на языке высокого уровня Delphi. Учебник. – М.: "Питер", 2003 г.

18. Peter Gottschling. Discovering Modern C++. An Intensive Course for Scientists, Engineers, and Programmers. "Addison-Wesley", 2015 y.

Internet saytlari

19. www.ziyonet.uz - Axborot ta'lim portal.
20. www.edu.uz - Oliy va o'rta maxsus ta'lim vazirligi portali.
21. www.tdpu.uz - Nizomiy nomidagi TDPU rasmiy sayti.
22. http://acm.tuit.uz/ - dasturiy yechim to'g'riligini avtomatik testlovchi tizim.
23. http://acm.tuit.uz/forum/, http://acm.timus.ru/ – dasturlarni testlovchi tizim.
24. http://codeforces.com/ - dasturiy yechim to'g'riligini avtomatik testlovchi tizim.
25. www.dasturchi.uz
26. www.ozon.ru/context/detail/id/2705337/
27. http://www.borlpasc.narod.ru/
28. www.nmarket.ru/program/delphi/lessons

O‘ZBEKISTON RESPUBLIKASI
XALQ TA’LIMI VAZIRLIGI

NAVOIY DAVLAT PEDAGOGIKA INSTITUTI

Ro‘yxatga olindi:	«TASDIQLAYMAN»
	O‘quv ishlari bo‘yicha prorektor
№ BDI-5110700-2.05	_____ b.f.n. Kushakov A.J.
«__»_____2019 yil	«___»_____2019 yil

DASTURLASH TILLARI
fanining

ISHCHI O‘QUV DASTURI

Bilim sohasi:	100000 – Gumanitar
Ta’lim sohasi:	110000 – Pedagogika
Ta’lim yo‘nalishi	5110700- Informatika o‘qitish metodikasi

Navoiy 2019

Ushbu ishchi dastur 2019 yil 17-avgustda BD-5110700-2.05 bilan ro‘yxatdan o‘tkazilgan va O‘zbekiston Respublikasi Oliy va o‘rta maxsus ta’lim vazirligi tomonidan 2019 yil 29-iyunda tasdiqlangan “Dasturlash tillari” namunaviy fan dasturi, namunaviy o‘quv rejaga muvofiq ishlab chiqildi.

Tuzuvchi:

Yodgorov G‘.R. – NavDPI, “Informatika o‘qitish metodikasi” kafedrasi mudiri, f.-m.f.n.,

Toxirov F.J. – NavDPI, “Informatika o‘qitish metodikasi” kafedrasi o‘qituvchisi.

Taqrizchilar:

O‘tapov T.U. - NavDPI, “Informatika o‘qitish metodikasi” kafedrasi dotsenti, p.f.n.

Xudoyorov Sh.J. - NavDPI, “Informatika o‘qitish metodikasi” kafedrasi dotsenti, f.-m.f.n.

Fanning ishchi o‘quv dasturi “Informatika o‘qitish metodikasi” kafedrasining 2019-yil 28 avgustdagi 1 - son yig‘ilishida muhokamadan o‘tgan va fakultet kengashida muhokama qilish uchun tavsiya etilgan.

Kafedra mudiri:_____ **f.-m.f.n. Yodgorov G‘.R.**

Fanning ishchi o‘quv dasturi “Fizika-matematika” fakultet kengashida muhokama etilgan va foydalanishga tavsiya qilingan (2019 yil 29 avgustdagi 1-sonli bayonnoma).

Fakultet kengashi raisi: _____dots. Kamolov I.R.

Navoiy davlat pedagogika institutining 2019 yil 30 – avgustdagi 1-sonli ilmiy uslubiy kengashida muhokama qilinib tasdiqlandi.

Kelishildi: O‘quv uslubiy boshqarma boshlig‘i

_____Xolmirzayev N.

KIRISH

Mazkur ishchi o‘quv dastur bakalavriat yo‘nalishi: *5110700 – Informatika o‘qitish metodikasi ta’lim yo‘nalishida taxsil* olayotgan talabalarning o‘zlashtirishi lozim bo‘lgan bilimlari va unga qo‘yiladigan talablar asosida tuzilgan bo‘lib, bo‘lajak fan o‘qituvchisi egallashi kerak bo‘lgan bilimlar va ko‘nikmalar majmuini o‘z ichiga oladi:

- dasturlash tillari va ularning klassifikasiyasi, dasturlash tilining alifbosi, buruqlar tizimi va operatorlarini, chiziqli, tarmoqlanuvchi va takrorlanuvchi dasturlar tuzishni;
- ob’ektga yo‘naltirilgan dasturlash tillari hakidagi, ob’ektga yo‘naltirilgan loyihalash, ob’ektlarni loyihalash, ob’ektga yo‘naltirilgan dasturlash tili va unda dastur tuzish.

Fanning maqsadi va vazifalari

“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 makalalami shaklantirish hamda rivojlantirishdan iborat.

Ushbu maqsadga erishish uchun fan talabalami 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 talabaning malakasiga qo‘yiladigan talablar

«Dasturlash tillari» o‘quv fanini o‘zlashtirish jarayonida amalga oshiriladigan masalalar doirasida bakalavr:

- 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 eg a 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‘naltiri!gan 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 natijalami taqqoslay olish ko‘nikmalariga ega bo‘lishi lozim.

O‘quv rejadagi boshqa fanlar bilan bog‘liqligi

Dasturlash tillari fani asosiy kasbiy fanlaridan biri hisoblananib, 3, 4 va 5-semestrlarda o‘qitiladi. Dasturni amalga oshirish o‘quv rejasidagi rejalashtirilgan oliy matematika, fizika, informatika, ma’lumotlar bazasi va ularni boshqarish tizimlari fanlaridan olingan nazariy va amaliy bilimlarga tayanadi.

Fanni o‘qitishda zamonaviy axborot va pedagogik texnologiyalar

O‘quv jarayoni bilan bog‘liq ta’lim sifatini belgilovchi holatlar quyidagilar: yuqori ilmiy-pedagogik darajada dars berish, muammoli ma’ruzalar o‘qish, darslarni savol-javob tarzida qiziqarli tashkil qilish, ilg‘or pedagogik texnologiyalardan va mul’timedia vositalaridan foydalanish, tinglovchilarni undaydigan, o‘ylantiradigan muammolarni ular oldiga qo‘yish, talabchanlik, tinglovchilar bilan individual ishlash, erkin muloqot yuritishga, ilmiy izlanishga jalb qilish.

“Dasturlash tillari” kursini loyihalashtirishda quyidagi asosiy kontseptual yondoshuvlardan foydalaniladi:

Shaxsga yo‘naltirilgan ta’lim. Bu ta’lim o‘z mohiyatiga ko‘ra ta’lim jarayonining barcha ishtirokchilarini to‘laqonli rivojlanishlarini ko‘zda tutadi. Bu esa ta’limni

loyihalashtirilayotganda, albatta, ma’lum bir ta’lim oluvchining shaxsini emas, avvalo, kelgusidagi mutaxassislik faoliyati bilan bog‘liq o‘qish maqsadlaridan kelib chiqqan holda yondoshilishni nazarda tutadi.

Tizimli yondoshuv. Ta’lim texnologiyasi tizimning barcha belgilarini o‘zida mujassam etmog‘i lozim: jarayonning mantiqiyligi, uning barcha bo‘g‘inlarini o‘zaro bog‘langanligi, yaxlitligi.

Faoliyatga yo‘naltirilgan yondoshuv. Shaxsning jarayonli sifatlarini shakllantirishga, ta’lim oluvchining faoliyatni aktivlashtirish va intensivlashtirish, o‘quv jarayonida uning barcha qobiliyati va imkoniyatlari, tashabbuskorligini ochishga yo‘naltirilgan ta’limni ifodalaydi.

Dialogik yondoshuv. Bu yondoshuv o‘quv munosabatlarini yaratish zaruriyatini bildiradi. Uning natijasida shaxsning o‘z-o‘zini faollashtirishi va o‘z-o‘zini ko‘rsata olishi kabi ijodiy faoliyati kuchayadi.

Hamkorlikdagi ta’limni tashkil etish. Demokratik, tenglik, ta’lim beruvchi va ta’lim oluvchi faoliyat mazmunini shakllantirishda va erishilgan natijalarni baholashda birgalikda ishlashni joriy etishga e’tiborni qaratish zarurligini bildiradi.

Muammoli ta’lim. Ta’lim mazmunini muammoli tarzda taqdim qilish orqali ta’lim oluvchi faoliyatini aktivlashtirish usullaridan biri. Bunda ilmiy bilimni ob’ektiv qarama-qarshiligi va uni hal etish usullarini, ialektik mushohadani shakllantirish va rivojlantirishni, amaliy faoliyatga ularni ijodiy tarzda qo‘llashni mustaqil ijodiy faoliyati ta’minlanadi.

Axborotni taqdim qilishning zamonaviy vositalari va usullarini qo‘llash - yangi kompyuter va axborot texnologiyalarini o‘quv jarayoniga qo‘llash.

O‘qitishning usullari va texnikasi. Ma’ruza (kirish, mavzuga oid, vizuallash), muammoli ta’lim, keys-stadi, pinbord, paradoks va loyihalash usullari, amaliy ishlar.

O‘qitishni tashkil etish shakllari: dialog, polilog, muloqot hamkorlik va o‘zaro o‘rganishga asoslangan frontal, kollektiv va guruh.

O‘qitish vositalari: o‘qitishning an’anaviy shakllari (darslik, ma’ruza matni) bilan bir qatorda – kompyuter va axborot texnologiyalari.

Kommunikatsiya usullari: tinglovchilar bilan operativ teskari aloqaga asoslangan bevosita o‘zaro munosabatlar.

Teskari aloqa usullari va vositalari: kuzatish, blits-so‘rov, oraliq va joriy va yakunlovchi nazorat natijalarini tahlili asosida o‘qitish diagnostikasi.

Boshqarish usullari va vositalari: o‘quv mashg‘uloti bosqichlarini belgilab beruvchi texnologik karta ko‘rinishidagi o‘quv mashg‘ulotlarini rejalashtirish, qo‘yilgan maqsadga erishishda o‘qituvchi va tinglovchining birgalikdagi harakati, nafaqat auditoriya mashg‘ulotlari, balki auditoriyadan tashqari mustaqil ishlarning nazorati.

Monitoring va baholash: o‘quv mashg‘ulotida ham butun kurs davomida ham o‘qitishning natijalarini rejali tarzda kuzatib borish. Kurs oxirida test topshiriqlari yoki yozma ish variantlari yordamida tinglovchilarning bilimlari baholanadi.

“Dasturlash tillari” fanini o‘qitish jarayonida kompyuter texnologiyasidan, Borland Delphi 7, Dev C++ va Borland C++ Builder dasturlaridan foydalaniladi. Ayrim mavzular bo‘yicha talabalar bilimini baholash test asosida va kompyuter yordamida bajariladi. “Internet” tarmog‘idagi rasmiy saytlardan foydalaniladi, tarqatma materiallar tayyorlanadi, test tizimi hamda tayanch so‘z va iboralar asosida oraliq va yakuniy nazoratlar o‘tkaziladi.

“Dasturlash tillari” fanidan mashg‘ulotlarning mavzular va soatlar bo‘yicha taqsimlanishi:

t/r	Mavzular nomi	Jami soat	Ma’ruza	Amaliy mash g‘ulot	Laboratoriya mash g‘uloti	Mustaqil ta’lim
III semester						
1.	1-modul. Ob’ektga yo‘naltirilgan dasturlash tillari.	14	4	2	2	6
2.	2-modul. Delphi dasturlash tili.	28	4	4	6	14
3.	3-modul. Delphida dasturlash.	42	6	4	10	22
4.	4-modul. Delphi dasturlash tili operatorlari.	86	16	16	20	34
Jami		**170**	**30**	**26**	**38**	**76**
IV semester						
5.	5-modul. C++ dasturlash tiliga kirish.	106	18	16	24	48
Jami		**106**	**18**	**16**	**24**	**48**
V semester						
6.	6-modul. Vizual dasturlar tuzish.	136	24	22	30	60
Jami		**136**	**24**	**22**	**30**	**60**
Hammasi		**412**	**72**	**64**	**92**	**184**

Asosiy qism: Fanning uslubiy jihatdan uzviy ketma-ketligi

Asosiy qismda (ma’ruza) fanni mavzulari mantiqiy ketma-ketlikda keltiriladi. Har bir mavzuning mohiyati asosiy tushunchalar va tezislar orqali ochib beriladi. Bunda mavzu bo‘yicha talabalarga DTS asosida yetkazilishi zarur bo‘lgan bilim va ko‘nikmalar to‘la qamrab olinishi kerak.

Asosiy qism sifatiga qo‘yiladigan talab mavzularning dolzarbligi, ularning ish beruvchilar talablari va ishlab chiqarish ehtiyojlariga mosligi, mamlakatimizda bo‘layotgan ijtimoiy-siyosiy va demokratik o‘zgarishlar, iqtisodiyotni erkinlashtirish, iqtisodiy-huquqiy va boshqa sohalardagi islohatlarning ustuvor masalalarini qamrab olishi hamda fan va texnologiyalarning so‘ngi yutuqlari e’tiborga olinishi tavsiya etiladi.

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 darajali dasturlash tillari. Interpretatorlar va kompilyatorlar. Dasturlarni translyatsiyalash. 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.

Qo‘llaniladigan ta’lim texnologiyalari: dialogik yondoshuv, muammoli ta’lim, blits, munozara, o‘z-o‘zini nazorat, SWOT-tahlil, baliq skeleti, pog‘ona, piramida, kichik guruhlarda ishlash, BBB jadvali, Venna diagrammasi, T chizma.

Adabiyotlar: A2; A3; A4; A5. Q10; Q14.

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‘rnatiIishi 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 ulaming 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.

Qo‘llaniladigan ta’lim texnologiyalari: dialogik yondoshuv, muammoli ta’lim, blits, munozara, o‘z-o‘zini nazorat, SWOT-tahlil, baliq skeleti, pog‘ona, piramida, kichik guruhlarda ishlash, BBB jadvali, Venna diagrammasi, T chizma.

Adabiyotlar: A2; A3; A4; A6. Q7; Q8; Q9; Q10; Q12; Q13.

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.

Qo‘llaniladigan ta’lim texnologiyalari: dialogik yondoshuv, muammoli ta’lim, blits, munozara, o‘z-o‘zini nazorat, SWOT-tahlil, baliq skeleti, pog‘ona, piramida, kichik guruhlarda ishlash, BBB jadvali, Venna diagrammasi, T chizma.

Adabiyotlar: A2; A3; A4; A6. Q7; Q8; Q9; Q10; Q12; Q13.

4-MODUL. DELPHI DASTURLASH TILI OPERATORLARI

10-mavzu. Delphi dasturlash muhitida 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 muhitida 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, Thrush klasslari. TFont klassi xossalari: Color, Name, Size, Style. TPen klassi xossalari: Color, Mode, Width, Style. TBrush klassi xossalari: Bitmap, Color, Style.

Qo‘llaniladigan ta'lim texnologiyalari: dialogik yondoshuv, muammoli ta'lim, blits, munozara, o‘z-o‘zini nazorat, SWOT-tahlil, baliq skeleti, pog‘ona, piramida, kichik guruhlarda ishlash, BBB jadvali, Venna diagrammasi, T chizma.

Adabiyotlar: A2; A3; A4; A6. Q7; Q8; Q9; Q10; Q12; Q13.

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. 0‘zgaruvchi va o‘zgarmas tipli kattaliklar.

0‘zgaruvchilar. Identifikatorlar. Ma'lumotlar tipi. 0‘zgaruvchilami e'lon qilish. 0‘zgaruvchilami 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 operatoming 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. Funksiyalarming 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.

22-mavzu. C++ da ko'rsatkichlar va satrlar.

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 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.

24-mavzu. C++ da fayllar bilan ishlash.

Matn fayllarini o'qish va yozish. Oqimni ochish. Fayldan o'qish. Faylga yozish. Binary fayllar bilan ishlash operatorlari. Matn va binar fayllar. 0‘qish- yozish oqimlari. Standart oqimlar. Belgilarni o‘qish-yozish funksiyalari. Satrlami o‘qish - yozish funksiyalari. Fayldan o‘qish-yozish funksiyalari. Formatli o‘qish va yozish funksiyalari. Fayl ko‘rsatkichini boshqarish funksiyalari.

Qo‘llaniladigan ta’lim texnologiyalari: dialogik yondoshuv, muammoli ta’lim, blits, munozara, o‘z-o‘zini nazorat, SWOT-tahlil, baliq skeleti, pog‘ona, piramida, kichik guruhlarda ishlash, BBB jadvali, Venna diagrammasi, T chizma.

Adabiyotlar: A1; A2; A5; A6; A7. Q11; Q14; Q15.

6-MODUL. VIZUAL DASTURLAR TUZISH

25-mavzu. Borland C++ Builder dasturlash muhiti.

Borland C++ Builder dasturlash muhitiga kirish, ishchi muhit, oynalar. C++ Bulder komponentlari va ularning hossalari. Komponentlar hodiasalari va metodlari. Komponentlar tarkibi. Hodisalar. Uslublar. Loyihalar menejeri. C++ Builder da ilova dastur yaratish. Oddiy ilova dasturini yararish.

26-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.

27-mavzu. Borland C++ Builder Standard komponentlar palitrasi.

Frame komponenti va uning xossalari. MainMenu, PopupMenu komponentlari. Label, Edit, Button, Memo, Panel komponentlari va ulaming xossalari. CheckBox, ListBox, ComboBox, ListBox, RadioGroup, RadioButton, ScrollBar komponentlari.

28-mavzu. Borland C++ Builder Additional komponentlar palitrasi.

BitBtn, MaskEdit, StringGrid, MaskEdit, CheckListBox, DrawGrid, Image, Shape va boshqa komponentalardan foydalanish. Komponentlar xossalari.

29-mavzu. Borland C++ Builder Dialogs komponentlar palitrasi.

Borland C++ Builderdagi Dialogs komponentlar palitrasi. OpenDialog, SaveDialog, FindDialog, ColorDialog, FontDialog va hokazo komponentlar va ularning xossalari.

30-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.

31-mavzu. Borland C++ Builderda multimedia va animatsiyalar.

Borland C++ Builderda multimedia va animatsiyalar. C++ tilida ko‘p formali loyihalar yaratish.

Qo‘llaniladigan ta’lim texnologiyalari: dialogik yondoshuv, muammoli ta’lim, blits, munozara, o‘z-o‘zini nazorat, SWOT-tahlil, baliq skeleti, pog‘ona, piramida, kichik guruhlarda ishlash, BBB jadvali, Venna diagrammasi, T chizma.

Adabiyotlar: A1; A2; A5; A6; A7. Q11; Q14; Q15; Q16.

“Dasturlash tillari” fani bo‘yicha ma’ruza mashg‘ulotining kalendar tematik rejasi

t/r	Ma’ruza mavzulari	Soat
III semester		
1-modul. Ob’ektga yo‘naltirilgan dasturlash tillari		
1.	Dasturlash tillari va ularning klassifikatsiyasi. Ob’ektga yo‘naltirilgan dasturlash tillari.	2
	2-modul. Delphi dasturlash tili	
2.	Delphi dasturlash tilining ishchi muhiti.	2
3.	Standard komponentalar palitrasi.	
4.	Additional komponentalar palitrasi.	2
5.	Dialogs komponentalar palitrasi.	2
6.	Samples komponentalar palitrasi.	2
	3-modul. Delphida dasturlash	
7.	Forma va uning xossalari. Xodisa tushunchasi.	2
8.	Delphida tiplar, o‘zgarmaslar, o‘zgaruvchilar va standart funksiyalar.	2
9.	Delphida simvolli va satriy kattaliklar.	2
	4-modul. Delphi dasturlash tili operatorlari	
10.	Delphi dasturlash muhitida tarmoqlanish operatorlari.	2
11.	Delphi dasturlash muhitida takrorlanish operatorlari.	2
12.	Delphi dasturlash muhitida bir o‘lchovli massivlar.	2
13.	Delphi dasturlash muhitida ikki o‘lchovli massivlar.	2
14.	Delphi dasturlash muhitida prosedura va funksiyalar.	2
15.	Delphi dasturlash muhitining grafik imkoniyatlari.	2
Jami		**30**
IV semester		
5-modul. C++ dasturlash tiliga kirish		
1.	C++ tilining yaratilish tarixi va leksik asoslari. C++ tilida o‘zgaruvchilar, ma’lumotlar tiplari va arifmetik operatorlar.	2
2.	C++ tilida tarmoqlanish operatorlari.	2
3.	C++ tilida takrorlanish operatorlari.	2
4.	C++ tilida funksiyalar.	2
5.	C++ tilida bir o‘lchovli massivlar.	2
6.	C++ tilida ikki o‘lchovli massivlar.	2
7.	C++ tilida ko‘rsatkichlar va satrlar. C++ tilida strukturalar va birlashmalar.	2
8.	C++ tilida fayllar bilan ishlash.	2
9.	C++ tilining grafik imkoniyatlari.	2
	Jami	**18**
VI semester		
6-modul. Vizual dasturlar tuzish		
1.	Borland C++ Builder dasturlash muhitiga kirish.	2
2.	Borland C++ Builder komponentlar palitrasi. Hodisa jarayonlarini yaratish.	2
3.	Panel va menyu yaratuvchi komponentlar: Panel, GroupBox, Bevel, ScroolBox, ToolBar, StatusBar.	2
4.	Borland C++ Builder Standard komponentlar palitrasi.	2
5.	Borland C++ Builder Additional komponentlar palitrasi.	2
6.	Borland C++ Builder Additional komponentlar palitrasi.	2
7.	Borland C++ Builder Dialogs komponentlar palitrasi.	2
8.	Borland C++ Builder Samples komponentlar palitrasi.	2
9.	Borland C++ Builderning grafik vositalari.	2

10.	Borland C++ Builderning grafik vositalari.	2
11.	Borland C++ Builderda multimedia va animatsiyalar.	2
12.	Borland C++ Builderda ko‘p formali loyihalar yaratish	2
	Jami	**24**
Jami		**72**

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‘ulotlarning tavsiya etiladigan mavzulari

1-MODUL. OB’EKTGA YO‘NALTIRILGAN DASTURLASH TILLARI

1-mavzu. “Dasturlash tillari” faniga kirish.

Dasturlash tillari va ularning klassifikasiyasi. Interpretatorlar va kompilyatorlar. Dasturlarni translyatsiyalash. 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. Ob’ektlar kutubxonasi. Interfeys ob’ektlari.

4-mavzu: Voqealar va habarlar.

Voqealar va habarlar. Ob’ektga yo‘naltirilgan muhitlarda habarlarni uzatish va ularga ishlov berish mexanizmlari.

Qo‘llaniladigan ta’lim texnologiyalari: dialogik yondoshuv, muammoli ta’lim, blits, munozara, o‘z-o‘zini nazorat, SWOT-tahlil, baliq skeleti, pog‘ona, piramida, kichik guruhlarda ishlash, BBB jadvali, Venna diagrammasi, T chizma.

Adabiyotlar: A2; A3; A4; A5. Q10; Q14.

2-MODUL. DELPHI DASTURLASH TILI

5-mavzu. Delphi dasturlash tili ishchi muhiti.

Delphi dasturlash tilining ishchi muhiti. Komponentlar palitrasi. Palitra bo‘limlari va ayrim komponentlar xossalari bilan tanishish.

6-mavzu. Standard komponentlar palitrasi.

Frame, MainMenu, PopupMenu, Label, Edit, Button, Memo, Panel, 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.

Qo‘llaniladigan ta’lim texnologiyalari: dialogik yondoshuv, muammoli ta’lim, blits, munozara, o‘z-o‘zini nazorat, SWOT-tahlil, baliq skeleti, pog‘ona, piramida, kichik guruhlarda ishlash, BBB jadvali, Venna diagrammasi, T chizma.

Adabiyotlar: A2; A3; A4; A6. Q7; Q8; Q9; Q10; Q12; Q13.

3-MODUL. DELPHIDA DASTURLASH

8-mavzu. Delphi dasturlari strukturasi. Loyiha va modul.

Bo‘sh forma va uning modifikasiyasi. Formaga yangi komponent joylashtirish va unda komponent xossalaridan foydalanish, Xodisa tushunchasi. Loyiha va modul strukturasi.

9-mavzu. Delphida tiplar, o‘zgarmaslar, o‘zgaruvchilar va standart funksiyalar.

Delphida tiplar, ularning ahamiyati. Butun tiplar. Delphida simvolli va satriy tiplar. Delphi dasturlash tilida o‘zgarmaslar, o‘zgaruvchilar va standart funksiyalar.

Qo‘llaniladigan ta’lim texnologiyalari: dialogik yondoshuv, muammoli ta’lim, blits, munozara, o‘z-o‘zini nazorat, SWOT-tahlil, baliq skeleti, pog‘ona, piramida, kichik guruhlarda ishlash, BBB jadvali, Venna diagrammasi, T chizma.

Adabiyotlar: A2; A3; A4; A6. Q7; Q8; Q9; Q10; Q12; Q13.

4-MODUL. DELPHI DASTURLASH TILI OPERATORLARI

10-mavzu. Delphi dasturlash muhitida 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 muhitida siklik operatorlar.

Delphi dasturlash tilida sikllar. For sikli. While sikli. Repeat sikli. Murakkab sikllar.

12-mavzu. Delphida massivlar.

Delphi dasturlash tilida massivlar. Massivlarni berilish usullari. Massiv elementlarini kiritish va chiqarish. Ko‘p o‘lchovli massivlar. Massiv elementlarini saralash va tartiblash.

13-mavzu. Prosedura va funksiyalar.

Prosedura va funksiyalar Delphi dasturlash tilining muhim instrumenti sifatida.

14-mavzu. Delphi dasturlash tilining grafik vositalari.

Delphi dasturlash tilining grafik imkoniyatlari. Delphidagi maxsus TCanvas, TFont, TPen, TBrush klasslari.

Qo‘llaniladigan ta’lim texnologiyalari: dialogik yondoshuv, muammoli ta’lim, blits, munozara, o‘z-o‘zini nazorat, SWOT-tahlil, baliq skeleti, pog‘ona, piramida, kichik guruhlarda ishlash, BBB jadvali, Venna diagrammasi, T chizma.

Adabiyotlar: A2; A3; A4; A6. Q7; Q8; Q9; Q10; Q12; Q13.

5-MODUL. C++ DASTURLASH TILIGA KIRISH

15-mavzu. C++ tilining leksik asoslari.

C++ dasturlash tiliga kirish. C++ dasturlash tili alifbosi va xizmatchi so‘zlari. Chiqarish operatori. Kiritish operatori

16-mavzu. 0‘zgaruvchi va o‘zgarmas tipli kattaliklar.

0‘zgaruvchilar. Identifikatorlar. Ma’lumotlar tipi. 0‘zgaruvchilami e’lon qilish. 0‘zgaruvchilami 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 uzun ko‘rinishi. Tanlash operatorlari. Shartsiz o‘tish operatori.

19-mavzu. C++ dasturlash tilida takrorlanuvchi jarayonlar.

Sikl operatorlari. Oldingi shartli While takrorlash operatori. Keyingi shartli do-while takrorlash operatori. For, break, Continue operatori.

20-mavzu. C++ dasturlash tilida funksiyalar.

Funsiyalar haqida tushuncha va ularni yaratish. Funksiyalarming tuzilishi. Funksiya parametrlari. Funksiyalardan foydalanish.

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.

22-mavzu. C++ da ko'rsatkichlar va satrlar.

Adres (manzil) operatori. Jo‘natish operatori. Ko‘rsatkich tipidagi o‘zgaruvchilarni e’lon qilish. Ko‘rsatkichga boshlang‘ich qiymat berish. Ko‘rsatkich ustida amallar.

23-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.

24-mavzu. C++ da fayllar bilan ishlash.

Matn fayllarini o'qish va yozish. Belgilarni o‘qish-yozish funksiyalari. Satrlami o‘qish - yozish funksiyalari. Fayldan o‘qish-yozish funksiyalari.

Qo‘llaniladigan ta’lim texnologiyalari: dialogik yondoshuv, muammoli ta’lim, blits, munozara, o‘z-o‘zini nazorat, SWOT-tahlil, baliq skeleti, pog‘ona, piramida, kichik guruhlarda ishlash, BBB jadvali, Venna diagrammasi, T chizma.

Adabiyotlar: A1; A2; A5; A6; A7. Q11; Q14; Q15.

6-MODUL. VIZUAL DASTURLAR TUZISH

25-mavzu. Borland C++ Builder dasturlash muhiti.

Borland C++ Builder dasturlash muhitiga kirish, ishchi muhit, oynalar. C++ Bulder komponentlari va ularning hossalari.

26-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. Menyu yaratish.

27-mavzu. Borland C++ Builder Standard komponentlar palitrasi.

Frame, MainMenu, PopupMenu, Label, Edit, Button, Memo, Panel, CheckBox, ListBox, ComboBox, ListBox, RadioGroup, RadioButton, ScrollBar komponentlari.

28-mavzu. Borland C++ Builder Additional komponentlar palitrasi.

BitBtn, MaskEdit, StringGrid, MaskEdit, CheckListBox, DrawGrid, Image, Shape va boshqa komponenti arid an foydalanish.

29-mavzu. Borland C++ Builder Dialogs komponentlar palitrasi.

Borland C++ Builderdagi Dialogs komponentlar palitrasi. **30-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.

31-mavzu. Borland C++ Builderda multimedia va animatsiyalar.

Borland C++ Builderda multimedia va animatsiyalar. C++ tilida ko‘p formali loyihalar yaratish.

Qo‘llaniladigan ta’lim texnologiyalari: dialogik yondoshuv, muammoli ta’lim, blits, munozara, o‘z-o‘zini nazorat, SWOT-tahlil, baliq skeleti, pog‘ona, piramida, kichik guruhlarda ishlash, BBB jadvali, Venna diagrammasi, T chizma.

Adabiyotlar: A1; A2; A5; A6; A7. Q11; Q14; Q15; Q16.

“Dasturlash tillari” fani bo‘yicha amaliy mashg‘ulotining kalendar tematik rejasi

t/r	Amaliy mashg‘ulot mavzulari	Soat
III semester		
2-modul. Delphi dasturlash tili		
1.	Komponent xossalarini dinamik va statik o‘zgartirish. Standart bo`limi komponentlaridan foydalanish.	2
2.	Additional bo`limi komponentlaridan foydalanish.	2
3.	Dialogs va Samples komponentalar palitralari.	2
4.	Forma tuzilishi va unda komponentlarni joylashtirish. Loyiha va modul strukturasi.	2

3-modul. Delphida dasturlash		
5.	Delphidagi dasturlarda tiplardan foydalanish. Tarkibiy va tanlash operatorlari.	2
4-modul. Delphi dasturlash tili operatorlari		
6.	Delphi dasturlash tilida takrorlanish operatorlari.	2
7.	Delphi dasturlash tilida massivlar va satriy kattaliklar.	2
8.	Delphi dasturlash tilida protsedura va funksiyalar.	2
9.	Delphida modullar va ulardan foydalanish	2
10.	Delphi dasturlash tilining Office dasturlari bilan hamkorligi.	2
11.	Delphi dasturlash muhitida fayllar bilan ishlash. Multimedia ilovalari.	2
12.	Shaxsiy loyihalarni yaratish bosqichlari. MBni boshqaradigan ilovalar tuzish.	2
13.	Delphi ning grafik komponentlari.	2
Jami		**26**
IV semester		
5-modul. C++ dasturlash tiliga kirish		
1.	C++ da ma'lumotlarning asosiy turlari. C++tilida chiziqli dasturlash.	2
2.	C++ tilida Shartli va shartsiz o'tish operatorlari. Tanlash operatori.	2
3.	C++ tilida takrorlanish operatorlari (while, do while, for).	2
4.	C++ tilida bir o'lchovli massivlar.	2
5.	C++tilida ko'p o'lchovli massivlar.	2
6.	C++ tilida funksiyalar yaratish. Strukturalar va birlashmalar. C++ tilida ko'rsatkichlar. C++tilida dinamik massivlar.	2
7.	C++ tilida sinflar.	2
8.	C++ tilida multimedia va animatsiyalar.	2
Jami		**16**
V semestr		
6-modul. Vizual dasturlar tuzish		
1.	Borland C++ Builder dasturlash muhitiga kirish. Guruhli operatsiyalar uchun komponentlarni tanlash. Menyu yaratish.	2
2.	Borland C++ Builder Standart komponentalar palitrasi.	2
3.	Borland C++ Builder Additional komponentalar palitrasi.	2
4.	Borland C++ Builder Dialog komponentalar palitrasi	2
5.	Borland C++ Builder Win32 komponentalar palitrasi	2
6.	Borland C++ Builder Samples komponentalar palitrasi	2
7.	Borland C++ Builderda massivlarga doir dastur tuzish.	2
8.	Borland C++ Builderda funksiyalarga doir dastur tuzish.	2
9.	Borland C++ Builderda grafika, multimedia va animatsiyalar.	2
10.	Borland C++ Builderda foydalanuvchi kutubxonasini yaratish	2
11.	Borland C++ Builderda ko‘p formali loyihalar yaratish	2
	Jami	**22**
Hammasi		**64**

Laboratoriya mashg‘ulotlarning tavsiya etiladigan mavzulari

1-MODUL. OB’EKTGA YO‘NALTIRILGAN DASTURLASH TILLARI

1-mavzu. “Dasturlash tillari” faniga kirish.

Dasturlash tillari va ularning klassifikasiyasi. Interpretatorlar va kompilyatorlar. Dasturlarni translyatsiyalash. 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. Ob’ektlar kutubxonasi. Interfeys ob’ektlari.

4-mavzu: Voqealar va habarlar.

Voqealar va habarlar. Ob’ektga yo‘naltirilgan muhitlarda habarlarni uzatish va ularga ishlov berish mexanizmlari.

Qo‘llaniladigan ta’lim texnologiyalari: dialogik yondoshuv, muammoli ta’lim, blits, munozara, o‘z-o‘zini nazorat, SWOT-tahlil, baliq skeleti, pog‘ona, piramida, kichik guruhlarda ishlash, BBB jadvali, Venna diagrammasi, T chizma.

Adabiyotlar: A2; A3; A4; A5. Q10; Q14.

2-MODUL. DELPHI DASTURLASH TILI

5-mavzu. Delphi dasturlash tili ishchi muhiti.

Delphi dasturlash tilining ishchi muhiti. Komponentlar palitrasi. Palitra bo‘limlari va ayrim komponentlar xossalari bilan tanishish.

6-mavzu. Standard komponentlar palitrasi.

Frame, MainMenu, PopupMenu, Label, Edit, Button, Memo, Panel, 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.

Qo‘llaniladigan ta’lim texnologiyalari: dialogik yondoshuv, muammoli ta’lim, blits, munozara, o‘z-o‘zini nazorat, SWOT-tahlil, baliq skeleti, pog‘ona, piramida, kichik guruhlarda ishlash, BBB jadvali, Venna diagrammasi, T chizma.

Adabiyotlar: A2; A3; A4; A6. Q7; Q8; Q9; Q10; Q12; Q13.

3-MODUL. DELPHIDA DASTURLASH

8-mavzu. Delphi dasturlari strukturasi. Loyiha va modul.

Bo‘sh forma va uning modifikasiyasi. Formaga yangi komponent joylashtirish va unda komponent xossalaridan foydalanish, Xodisa tushunchasi. Loyiha va modul strukturasi.

9-mavzu. Delphida tiplar, o‘zgarmaslar, o‘zgaruvchilar va standart funksiyalar.

Delphida tiplar, ularning ahamiyati. Butun tiplar. Delphida simvolli va satriy tiplar. Delphi dasturlash tilida o‘zgarmaslar, o‘zgaruvchilar va standart funksiyalar.

Qo‘llaniladigan ta’lim texnologiyalari: dialogik yondoshuv, muammoli ta’lim, blits, munozara, o‘z-o‘zini nazorat, SWOT-tahlil, baliq skeleti, pog‘ona, piramida, kichik guruhlarda ishlash, BBB jadvali, Venna diagrammasi, T chizma.

Adabiyotlar: A2; A3; A4; A6. Q7; Q8; Q9; Q10; Q12; Q13.

4-MODUL. DELPHI DASTURLASH TILI OPERATORLARI

10-mavzu. Delphi dasturlash muhitida 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 muhitida siklik operatorlar.

Delphi dasturlash tilida sikllar. For sikli. While sikli. Repeat sikli. Murakkab sikllar.

12-mavzu. Delphida massivlar.

Delphi dasturlash tilida massivlar. Massivlarni berilish usullari. Massiv elementlarini kiritish va chiqarish. Ko‘p o‘lchovli massivlar. Massiv elementlarini saralash va tartiblash.

13-mavzu. Prosedura va funksiyalar.

Prosedura va funksiyalar Delphi dasturlash tilining muhim instrumenti sifatida.

14-mavzu. Delphi dasturlash tilining grafik vositalari.

Delphi dasturlash tilining grafik imkoniyatlari. Delphidagi maxsus TCanvas, TFont, TPen, TBrush klasslari.

Qo‘llaniladigan ta’lim texnologiyalari: dialogik yondoshuv, muammoli ta’lim, blits, munozara, o‘z-o‘zini nazorat, SWOT-tahlil, baliq skeleti, pog‘ona, piramida, kichik guruhlarda ishlash, BBB jadvali, Venna diagrammasi, T chizma.

Adabiyotlar: A2; A3; A4; A6. Q7; Q8; Q9; Q10; Q12; Q13.

5-MODUL. C++ DASTURLASH TILIGA KIRISH

15-mavzu. C++ tilining leksik asoslari.

C++ dasturlash tiliga kirish. C++ dasturlash tili alifbosi va xizmatchi so‘zlari. Chiqarish operatori. Kiritish operatori

16-mavzu. 0‘zgaruvchi va o‘zgarmas tipli kattaliklar.

0‘zgaruvchilar. Identifikatorlar. Ma’lumotlar tipi. 0‘zgaruvchilami e’lon qilish. 0‘zgaruvchilami 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 uzun ko‘rinishi. Tanlash operatorlari. Shartsiz o‘tish operatori.

19-mavzu. C++ dasturlash tilida takrorlanuvchi jarayonlar.

Sikl operatorlari. Oldingi shartli While takrorlash operatori. Keyingi shartli do-while takrorlash operatori. For, break, Continue operatori.

20-mavzu. C++ dasturlash tilida funksiyalar.

Funsiyalar haqida tushuncha va ularni yaratish. Funksiyalarming tuzilishi. Funksiya parametrlari. Funksiyalardan foydalanish.

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.

22-mavzu. C++ da ko'rsatkichlar va satrlar.

Adres (manzil) operatori. Jo‘natish operatori. Ko‘rsatkich tipidagi o‘zgaruvchilarni e’lon qilish. Ko‘rsatkichga boshlang‘ich qiymat berish. Ko‘rsatkich ustida amallar.

23-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.

24-mavzu. C++ da fayllar bilan ishlash.

Matn fayllarini o'qish va yozish. Belgilarni o‘qish-yozish funksiyalari. Satrlami o‘qish - yozish funksiyalari. Fayldan o‘qish-yozish funksiyalari.

Qo‘llaniladigan ta’lim texnologiyalari: dialogik yondoshuv, muammoli ta’lim, blits, munozara, o‘z-o‘zini nazorat, SWOT-tahlil, baliq skeleti, pog‘ona, piramida, kichik guruhlarda ishlash, BBB jadvali, Venna diagrammasi, T chizma.

Adabiyotlar: A1; A2; A5; A6; A7. Q11; Q14; Q15.

6-MODUL. VIZUAL DASTURLAR TUZISH

25-mavzu. Borland C++ Builder dasturlash muhiti.

Borland C++ Builder dasturlash muhitiga kirish, ishchi muhit, oynalar. C++ Bulder komponentlari va ularning hossalari.

26-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. Menyu yaratish.

27-mavzu. Borland C++ Builder Standard komponentlar palitrasi.

Frame, MainMenu, PopupMenu, Label, Edit, Button, Memo, Panel, CheckBox, ListBox, ComboBox, ListBox, RadioGroup, RadioButton, ScrollBar komponentlari.

28-mavzu. Borland C++ Builder Additional komponentlar palitrasi.

BitBtn, MaskEdit, StringGrid, MaskEdit, CheckListBox, DrawGrid, Image, Shape va boshqa komponenti arid an foydalanish.

29-mavzu. Borland C++ Builder Dialogs komponentlar palitrasi.

Borland C++ Builderdagi Dialogs komponentlar palitrasi.

30-mavzu. Borland C++ Builderning grafik vositalari.

Borland C++ Builderning maxsus TCanvas, TFont, TPen, Tbrush klasslari. TFont klassi xossalari. TPen klassi xossalari. TBrush klassi xossalari.

31-mavzu. Borland C++ Builderda multimedia va animatsiyalar.

Borland C++ Builderda multimedia va animatsiyalar. C++ tilida ko‘p formali loyihalar yaratish.

Qo‘llaniladigan ta’lim texnologiyalari: dialogik yondoshuv, muammoli ta’lim, blits, munozara, o‘z-o‘zini nazorat, SWOT-tahlil, baliq skeleti, pog‘ona, piramida, kichik guruhlarda ishlash, BBB jadvali, Venna diagrammasi, T chizma.

Adabiyotlar: A1; A2; A5; A6; A7. Q11; Q14; Q15; Q16.

“Dasturlash tillari” fani bo‘yicha laboratoriya mashg‘ulotining kalendar tematik rejasi

t/r	Laboratoriya mashg‘uloti mavzulari	Soat
III semester		
2-modul. Delphi dasturlash tili		
1.	Standard bo‘limi komponentlari yordamida sodda dasturlar tuzish.	2
2.	Additional bo‘limi komponentlari yordamida sodda dasturlar tuzish.	2
3.	Dialogs bo‘limi komponentlari yordamida sodda dasturlar tuzish.	2
4.	Samples bo‘limi komponentlari yordamida sodda dasturlar tuzish.	2
3-modul. Delphida dasturlash		
5.	Forma va uning xossalari. Xodisa tushunchasi.	2
4-modul. Delphi dasturlash tili operatorlari		
6.	Tarmoqlanish operatorlari yordamida dasturlar tuzish.	2
7.	Tarmoqlanish operatorlari yordamida dasturlar tuzish.	2
8.	Takrorlanish operatorlari yordamida dasturlar tuzish.	2
9.	Takrorlanish operatorlari yordamida dasturlar tuzish.	2
10.	Tasodifiy sonlar bilan ishlash.	2
11.	Sana-vaqt tipi bilan ishlash. Satriy kattaliklar bilan ishlash.	2
12.	Delphi dasturlash muhitida funksiya va proseduralardan foydalanish.	2
13.	Delphi dasturlash muxitida modul tuzilishi va undan foydalanish.	2
14.	Delphi dasturlash tilining Office dasturlari bilan hamkorligi.	2
15.	Delphi dasturlash muxitining grafik imkoniyatlari.	2
16.	Delphi dasturlash muxitida fayllar bilan ishlash.	2
17.	Delphining multimediyali imkoniyatlari.	2
18.	Delphida spravka tizimini yaratish.	2
19.	Delphida ma’lumotlar bazasini yaratish va boshqarish	2
Jami		**38**
IV semester		

5-modul. C++ dasturlash tiliga kirish		
1.	C++ tilida chiziqli dasturlar tuzish	2
2.	C++ tilida If operatori bilan ishlash	2
3.	C++ tilida For operatori bilan ishlash	2
4.	C++ tilida While operatori bilan ishlash	2
5.	C++ tilida Do-while operatori bilan ishlash	2
6.	C++ tilida bir o‘lchovli, ikki o‘lchovli massivlar.	4
7.	C++ tilida satriy kattaliklar.	2
8.	C++ tilida sinflar.	4
9.	C++ tilida grafika, multimedia va animatsiyalar.	2
10.	C++ tilida fayllar bilan ishlash	2
Jami		**24**
VI semestr		
6-modul. Vizual dasturlar tuzish		
1.	Borland C++ Builder dasturlash muhitiga kirish, ishchi muhit, oynalar.	2
2.	Borland C++ Builderda Label, Edit, Button va Memo komponentlaridan foydalanib dastur tuzish.	2
3.	Borland C++ Builderda CheckBox, RadioGroup, ComboBox va ListBox kompenentlaridan foydalanib dastur tuzish.	2
4.	Borland C++ Builderda Additional komponentlaridan foydalanib dastur tuzish.	2
5.	Panel va menyu yaratuvchi komponentlardan foydalanib dastur tuzish.	2
6.	Borland C++ Builder Dialog komponentalar palitrasidan foydalanib dastur tuzish.	2
7.	Borland C++ Builder Win32 komponentalar palitrasidan foydalanib dastur tuzish.	2
8.	Borland C++ Builder Samples komponentalar palitrasidan foydalanib dastur tuzish.	2
9.	Borland C++ Builderda bir o‘lchovli massivlarga doir dastur tuzish.	2
10.	Borland C++ Builderda ikki o‘lchovli massivlarga doir dastur tuzish.	2
11.	Borland C++ Builderda funksiyalarga doir dastur tuzish.	2
12.	Borland C++ Builderda multimedia va animatsiyalarga doir dastur tuzish	2
13.	Borland C++ Builderda grafikaga doir dastur tuzish	2
14.	Borland C++ Builderda foydalanuvchi kutubxonasini yaratish	2
15.	Borland C++ Builderda ko‘p formali loyihalar yaratish.	2
	Jami	**30**
Jami		**92**

Mustaqil ta’limni tashkil etishning shakli va mazmuni

“Dasturlash tillari” bo‘yicha talabaning mustaqil ta’limi shu fanni o‘rganish jarayonining tarkibiy qismi bo‘lib, uslubiy va axborot resurslari bilan to‘la ta’minlangan.

Talabalar auditoriya mashg‘ulotlarida professor-o‘qituvchilarning ma’ruzasini tinglaydilar, misol va masalalar yechadilar. Auditoriyadan tashqarida talaba darslarga tayyorlanadi, adabiyotlarni konspekt qiladi, uy vazifa sifatida berilgan misol va masalalarni yechadi. Bundan tashqari ayrim mavzularni kengroq o‘rganish maqsadida qo‘shimcha adabiyotlarni o‘qib referatlar tayyorlaydi hamda mavzu bo‘yicha testlar yechadi. Mustaqil ta’lim natijalari reyting tizimi asosida baholanadi.

Uyga vazifalarni bajarish, qo‘shimcha darslik va adabiyotlardan yangi bilimlarni mustaqil o‘rganish, kerakli ma’lumotlarni izlash va ularni topish yo‘llarini aniqlash, internet tarmoqlaridan foydalanib ma’lumotlar to‘plash va ilmiy izlanishlar olib borish, ilmiy to‘garak doirasida yoki mustaqil ravishda ilmiy manbalardan foydalanib ilmiy maqola va ma’ruzalar

tayyorlash kabilar talabalarning darsda olgan bilimlarini chuqurlashtiradi, ularning mustaqil fikrlash va ijodiy qobiliyatini rivojlantiradi. Shuning uchun ham mustaqil ta’limsiz o‘quv faoliyati samarali bo‘lishi mumkin emas.

Talaba mustaqil ishni tayyorlashda fanning xususiyatlarini hisobga olgan holda quyidagi shakllardan foydalanish tavsiya etiladi:

- darslik va o‘quv qo‘llanmalar bo‘yicha fan boblari va mavzularini o‘rganish;
- tarqatma materiallar bo‘yicha ma’ruzalar qismini o‘zlashtirish;
- avtomatlashtirilgan o‘rgatuvchi va nazorat qiluvchi dasturlar bilan ishlash;
- maxsus adabiyotlar bo‘yicha fanlar bo‘limlari yoki mavzulari ustida ishlash;
- talabaning o‘quv-ilmiy-tadqiqot ishlarini bajarish bilan bog‘liq bo‘lgan fanlar bo‘limlari va mavzularni chuqur o‘rganish;
- zamonaviy dasturlash muhitidan foylanishga mustaqil o‘rganishlari;

№	Mustaqil ta’lim mavzulari	Berilgan topshiriqlar	Bajar. muddat	Jami (soatda)
III semester				
1.	Delphi bilan tanishish	Adabiyotlar to‘plash. Yozma hisobot tayyorlash.	1-2 haftalar	4
2.	Delphida qo`llanaladigan matematik funksiyalar	Yozma hisobot tayyorlash. Dasturlar tuzish.	3-4 haftalar	4
3.	TForm komponentlari va ularning xossalari	Mavzuga doir dasturlar tuzish.	5-6 haftalar	4
4.	Standart bo`limi komponentalari	Mavzuga doir dasturlar tuzish.	7-8 haftalar	6
5.	Additional bo`limi komponentalari	Mavzuga doir dasturlar tuzish.	8-9 haftalar	6
6.	Sodda dasturlarni tuzish	Mavzuga doir dasturlar tuzish.	9-10 haftalar	4
7.	Shartli va siklli dasturlar tuzish	Yozma hisobot tayyorlash. Dasturlar tuzish.	11-12 haftalar	6
8.	Funksiya va proseduralami yaratish	Mavzuga doir dasturlar tuzish.	12-13 haftalar	6
9.	Grafik uskunalar bilan ishlash	Mavzuga doir dasturlar tuzish.	13-14 haftalar	6
10.	ListBox da grafiklarni joylashtirish	Yozma hisobot tayyorlash. Dasturlar tuzish.	14-15 haftalar	6
11.	Biror predmet sohasiga oid o‘rgatuvchi dasturlar yaratish	Mavzuga doir dasturlar tuzish.	16-17 haftalar	6
12.	Nazorat qiluvchi dasturlar yaratish	Mavzuga doir dasturlar tuzish.	17-18 haftalar	6
Jami				**76**
IV semester				
13.	C++ tili va uning leksik asosi	Adabiyotlar to‘plash. Yozma hisobot tayyorlash.	1-2 haftalar	2
14.	C++ tilida funksiyalar, strukturalar va birlashmalar.	Yozma hisobot tayyorlash.Amaliy mashqlar bajarish	3-4 haftalar	4
15.	C++ tilida If operatori bilan ishlash	Yozma hisobot tayyorlash.Amaliy mashqlar bajarish	5-6 haftalar	4
16.	C++ tilida For operatori bilan ishlash	Yozma hisobot tayyorlash.Amaliy mashqlar bajarish	7-8 haftalar	4

17.	C++ tilida While operatori bilan ishlash	Yozma hisobot tayyorlash.Amaliy mashqlar bajarish	8-9 haftalar	4
18.	C++ tilida Do-while operatori bilan ishlash	Mavzuga doir dasturlar tuzish	9-10 haftalar	4
19.	C++ tilida ko'rsatkichlar va murojaatlar.	Yozma hisobot tayyorlash.Amaliy mashqlar bajarish	11-12 haftalar	4
20.	C++ tilida massivlar.	Yozma hisobot tayyorlash.Amaliy mashqlar bajarish	12-13 haftalar	4
21.	C++ tilida satriy kattaliklar.	Yozma hisobot tayyorlash. Mavzuga doir dasturlar tuzish	13-14 haftalar	4
22.	C++ tilida sinflar.	Yozma hisobot tayyorlash. Mavzuga doir dasturlar tuzish	14-15 haftalar	4
23.	C++ tilida grafika	Yozma hisobot tayyorlash. Mavzuga doir dasturlar tuzish	15-16 haftalar	4
24.	C++ tilida fayllar bilan ishlash.	Kichik guruhlarga bo'linib mavzuga doir amaliy mashqlar bajarish.	17-18 haftalar	6
Jami				**48**
V semester				
25.	Borland C++ Builder dasturi bilan tanishish.	Adabiyotlar to'plash. Yozma hisobot tayyorlash.	1-2 haftalar	4
26.	Borland C++ Builderda chiziqli dasturlar tuzish.	Yozma hisobot tayyorlash. Amaliy mashqlar bajarish	3-4 haftalar	4
27.	Borland C++ Builderda tarmoqlanuvchi dasturlar tuzish.	Yozma hisobot tayyorlash. Amaliy mashqlar bajarish	5-6 haftalar	6
28.	Borland C++ Builderda takrorlanuvchi dasturlar tuzish.	Yozma hisobot tayyorlash. Amaliy mashqlar bajarish	7-8 haftalar	6
29.	Borland C++ Builderda massivlar ustida amallar bajarish.	Yozma hisobot tayyorlash. Amaliy mashqlar bajarish	8-9 haftalar	6
30.	Borland C++ Builder Dialog oynalari.	Mavzuga doir dasturlar tuzish	10-11 haftalar	6
31.	Borland C++ Builder Samples komponentalar palitrasi.	Yozma hisobot tayyorlash. Amaliy mashqlar bajarish	12-13 haftalar	6
32.	Borland C++ Builderning grafik imkoniyatlari.	Yozma hisobot tayyorlash. Amaliy mashqlar bajarish	14-15 haftalar	6
33.	Borland C++ Builderda multimedia ilovalarini yaratish.	Yozma hisobot tayyorlash. Mavzuga doir dasturlar tuzish	16-17 haftalar	8
34.	Borland C++ Builderda ko`p formali loyihalar yaratish.	Yozma hisobot tayyorlash. Mavzuga doir dasturlar tuzish	18-19 haftalar	8
	Jami			**60**
Hammasi				**184**

Dasturning informatsion – uslubiy ta'minoti

Mazkur fanni o'qitish jarayonida ta'limning zamonaviy metodlari, pedagogik va axborot-kommunikasiya texnologiyalarni, zamonaviy kompyuter texnologiyalarini qo'llanilishi nazarda tutilgan.

- ma’ruza darslarida zamonaviy kompyuter texnologiyalari yordamida prezentasion va elektron-didaktik texnologiyalaridan;
- amaliy mashg‘ulotlarda zamonviy pedagogik va innovasion texnologiyalaridan;
- laboratoriya mashg‘ulotlarida zamonaviy kompyuter sinflaridan foydalanish ko‘zda tutilgan. Shuningdek buguni kun talabiga javob beradigan dasturlash tillaridan Paskal, Delphi dasturlash tillarini o‘rnatuvchi disk ham bo‘lishi lozim.

BAHOLASH MEZONI

I. Umumiy talablar

1. Namunaviy o‘quv reja va ishchi o‘quv rejada mavjud fanlardan talabalar bilimini O‘zbekiston Respublikasi Oliy va o‘rta maxsus ta’lim vazirligining 2018 yil 9 avgustdagi 19-2018-son buyrug‘iga asosan baholash ularning o‘qitilayotgan fanni chuqur egallashi, topshiriqlarga ijodiy yondoshuvi, mustaqil fikrlash, o‘z bilimini muntazam ravishda oshirishga intilishi hamda adabiyotlardan keng foydalanish kabi xususiyatlarni rivojlantirishi va shu tariqa raqobatbardosh mutaxassislarni tayyorlashga erishishdan iboratdir.

2. O‘quv rejasidagi har bir fanidan baholash mezon quyidagi vazifalarni bajarishga qaratilgan:

- talabalar fanni o‘zlashtirishini muntazam ravishda nazorat qilib borish, ularni semestr (o‘quv yili) davomida o‘z ustlarida uzluksiz faol ishlashlarini ta’minlash;
- talabalar bilimini haqqoniy, aniq, adolatli va shaffof baholash hamda natijalarini ularga muntazam ravishda ma’lum qilish;
- talabalarda mustaqil ishlash ko‘nikmalarini keng rivojlantirish;
- professor-o‘qituvchilarda ma’ruza, amaliy, laboratoriya va seminar mashg‘ulotlarga puxta tayyorgarlik ko‘rish, baholash savollarini tuzishda mas’uliyatini oshirish.

3. Fan bo‘yicha maksimal “5” (a’lo) baho quyiladi. O‘quv rejasida aynan shu fanga ajratilgan soatlar miqdori bilan belgilanadi.

Oliy talim muassasalarida talabalar bilimini nazorat qilish va baholash tizimi to‘g‘risidagi Nizom va uning har bir bandi namunaviy o‘quv rejadagi va ishchi o‘quv rejadagi har bir fanning birinchi mashg‘ulotida talabalarga e’lon qilinadi. Fan bo‘yicha talabalarning bilim saviyasi va o‘zlashtirish darajasining **Davlat ta’lim standartlari va malakaviy talablar**ga muvofiqligini ta’minlash uchun quyidagi oraliq va yakuniy nazorat turlari o‘tkaziladi.

II. Baholash turlari va shakllari

II.1. Oraliq nazorat turi

Oliy ta’lim muassasalarida talabalar bilimini nazorat qilish oraliq va yakuniy nazorat turlarini o‘tkazish orqali amalga oshiriladi.

Oraliq nazorat semestr davomida ishchi fan dasturining tegishli bo‘limi tugagandan keyin talabaning bilim va amaliy ko‘nikmalarini baholash maqsadida o‘quv mashg‘ulotlari davomida o‘tkaziladi.

Oraliq nazorat turi har bir fan bo‘yicha fanning hususiyatidan kelib chiqqan holda 2 martagacha o‘tkazilishi mumkin.

Oraliq nazorat turini o‘tkazish shakli (*yozma, og‘zaki, test va hokazo)* va muddati fanning xususiyati va fanga ajratilgan soatlardan kelib chiqib tegishli kafedra tomonidan belgilanadi.

Oraliq nazorat turining topshiriqlari tegishli kafedra professor-o‘qituvchilari tomonidan ishlab chiqiladi va mazkur kafedra mudiri tomonidan tasdiqlanadi.

Semestr davomida haftasiga 2 akademik soatdan kam bo‘lgan fanlar bo‘yicha oraliq nazorat turi o‘tkazilmaydi.

Talabaning amaliy, seminar, laboratoriya mashg‘ulotlari va mustaqil ta’lim topshiriqlarini bajarishi, shuningdek uning ushbu mashg‘ulotlardagi faolligi fan o‘kituvchisi tomonidan baholab boriladi. Baholash O‘zbekiston Respublikasi Oliy va o‘rta maxsus ta’lim vazirligining 2018 yil 9 avgustdagi 19-2018-son buyrug‘ining ya’ni “Oliy talim muassasalarida talabalar bilimini nazorat

qilish va baholash tizimi to‘g‘risida”gi Nizomning 15-bandida nazarda tutilgan mezonlar asosida amalga oshiriladi.

Amaliy, laboratoriya, seminar mashg‘ulotlar va mustaqil ta’lim topshiriqlari bajarishi hamda mashg‘ulotlarda faolligi uchun talabani baholash mezonida quyiladigan talablar qo‘yiladi.

Baho		**Talabaning bilim darajasi va malakasiga talablar**
5	A’lo	Amaliy, laboratoriya, seminar mashg‘ulotlar va mustaqil ta’lim topshiriqlari bajarishi hamda mashg‘ulotlarda faol ishtirok etadi, berilgan topshiriqlarni mustaqil fikr asosida to‘g‘ri bajaradi, javoblarni izohlab ularning amaliy ahamiyatini anglay oladi, topshiriqlarni bajarishda ijodiy yondoshadi va ijodiy fikrlay oladi, o‘z fikrini to‘la ifodalay oladi, laboratoriya mashg‘ulotlarini o‘z vaqtida bajarib topshiradi, mustaqil ta’lim mavzularini to‘liq o‘zlashtiradi, talaba mustaqil xulosa va qaror qabul qiladi, mustaqil mushohada yuritadi, olgan bilimini amalda qo‘llay oladi, fanning (mavzuning) mohiyatini tushunda, biladi, ifodalay oladi, aytib beradi hamda fan (mavzu) bo‘yicha tasavvurga ega deb topilganda.
4	Yaxshi	Amaliy, laboratoriya, seminar mashg‘ulotlar va mustaqil ta’lim topshiriqlari bajarishi hamda mashg‘ulotlarda faol ishtirok etadi, berilgan topshiriq va mashqlarni to‘g‘ri bajaradi, javoblarni izohlay oladi, fikrini mustaqil ifodalay oladi, topshiriq mohiyatini to‘la tushunadi, laboratoriya mashg‘ulotlarini o‘z vaqtida bajarib topshiradi, mustaqil ta’lim mavzularini 70%dan 90%gacha o‘zlashtiradi, talaba mustaqil mushohada yuritadi, olgan bilimini amalda qullay oladi, fanning (mavzuning) mohiyatini tushunadi, biladi, ifodalay oladi, aytib beradi hamda fan (mavzu) bo‘yicha tasavvurga ega deb topilganda.
3	Qoniqarli	Amaliy, laboratoriya, seminar mashg‘ulotlar va mustaqil ta’lim topshiriqlari bajarishi hamda mashg‘ulotlarda ishtirok etib topshiriqlarni o‘qituvchi yordamida to‘g‘ri bajaradi, yechimlardan olingan javoblarni mohiyatini tushunadi, masalani yechish jarayonini tushuntira oladi, mustaqil berilgan laboratoriya mashg‘ulotlarini 60% yoki 70% o‘z vaqtida bajaradi, mustaqil ta’limni qisman o‘zlashtiradi, talaba olgan bilimini amalda qo‘llay oladi, fanning (mavzuning) mohiyatini tushunadi, biladi, ifodalay oladi, aytib beradi hamda fan (mavzu) bo‘yicha tasavvurga ega deb topilganda.
2	Qoniqarsiz	Amaliy, laboratoriya, seminar mashg‘ulotlar va mustaqil ta’lim topshiriqlari bajarishi hamda mashg‘ulotlardagi topshiriqlarning shartini to‘g‘ri tushunib ularni to‘liq bajara olmaydi, topshiriqlarni ba’zilari to‘g‘risida aniq tasavvurga ega bo‘lmaydi, o‘qituvchi ko‘magida ham topshiriqlarni bajarishga qiynaladi, nazariy va amaliy bilimlarini bog‘lay olmaydi, laboratoriya mashg‘ulotlarini o‘z vaqtida bajarmaydi va topshira olmaydi, mustaqil ta’limni o‘zlashtirmaydi, talaba fan dasturini o‘zlashtirmagan, fanning (mavzuning) mohiyatini tushunmaydi hamda fan (mavzu) bo‘yicha tasavvurga ega emas deb topilganda.

Talabani oraliq nazorat turi bo‘yicha baholashda, uning o‘quv mashg‘ulotlari davomida olgan baholari inobatga olinadi.

Talabalar bilimni baholash 5 baholik tizimda amalga oshiriladi. Oraliq nazorat turini o‘tkazish va mazkur nazorat turi bo‘yicha talabaning bilimini baholash tegishli fan bo‘yicha o‘quv mashg‘ulotlarini olib borgan professor-o‘qituvchi tomonidan amalga oshiriladi.
Talaba tegishli fan bo‘yicha yakuniy nazorat turi o‘tkaziladigan muddatga qadar oraliq nazorat turini topshirgan bo‘lishi shart.
Oraliq nazorat turini topshirmagan, shuningdek ushbu nazorat turi bo‘yicha “2” (qoniqarsiz) baho bilan baholangan talaba yakuniy nazorat turiga kiritilmaydi.
Oraliq (yozma, og‘zaki) nazoratda talaba bilimini baholashda quyidagi talablar qo‘yiladi:

Baho		**Talaba bilimi va malakasiga talablar**
5	A’lo	Savollardagi mavzularning barchasiga asoslangan, ilmiy xatoliklarga yo‘l qo‘yilmagan holda javoblar beradi, mavzu materiali mohiyatini to‘la tushunib yetgan bo‘ladi, ijodiy fikr yuritadi, mustaqil mushohada qiladi, nazariy bilimlarni amalda qo‘llashga misollar keltira oladi, xulosa va qarorlar qabul qilishda faol bo‘ladi, material bo‘yicha to‘la tasavvurga ega bo‘ladi va talaba ilmiy-uslubiy maqolalar yozishga loyiq bo‘ladi hamda amaliy, laboratoriya, seminar mashg‘ulotlar va mustaqil ta’lim topshiriqlari bajarishi shuningdek mashg‘ulotlarda faol ishtirok etagan deb topilganda.
4	Yaxshi	Savollarning barchasiga to‘liq javob beradi, juz’iy xatoliklarga yo‘l qo‘ymaydi. Material mohiyatni tushunib yetgan bo‘ladi, ijodiy fikr yurita oladi, nazariy bilimlarni amaliy ahamiyatini anglab yetgan bo‘ladi, material bo‘yicha tasavvurga ega bo‘lsa qo‘shimcha adabiyotlardan mustaqil foydalana olish qobiliyatiga hamda amaliy, laboratoriya, seminar mashg‘ulotlar va mustaqil ta’lim topshiriqlari bajarishi hamda mashg‘ulotlardagi yaxshi ishtirokni inobatga olgan deb topilganda
3	Qoniqarli	Savollarga javoblar yozadi, yo‘l qo‘ygan xatolari juz’iy xatolar bo‘lmaydi, material mohiyatini tushungan bo‘ladi, nazariy bilimlarni amaliy ahamiyatini anglagan bo‘ladi, mavzular bo‘yicha tasavvurga ega bo‘ladi va auditoriya mashg‘ulotlariga to‘liq qatnashgan bo‘ladi hamda fan (mavzu) bo‘yicha tasavvurga ega hamda amaliy, laboratoriya, seminar mashg‘ulotlar va mustaqil ta’lim topshiriqlari bajarishi hamda mashg‘ulotlarda ishtirok etib topshiriqlarni o‘qituvchi yordamida to‘g‘ri bajaradi deb topilganda
2	Qoniqarsiz	Savollarga javob berishga qiynaladi, material mohiyatini tushunmaydi, tasavvuri sayoz bo‘ladi, nazariy bilimlarni amaldagi ahamiyatni anglab yetmaydi, savollarni ko‘pchiligiga javob bera olmaydi va darslarga muntazam qatnashmagan bo‘laydi hamda fan (mavzu) bo‘yicha tasavvurga ega emas hamda amaliy, laboratoriya, seminar mashg‘ulotlar va mustaqil ta’lim topshiriqlari bajarishi hamda mashg‘ulotlardagi topshiriq-larning shartini to‘g‘ri tushunib ularni bajara olmaydi deb topilganda

Oraliq (test) nazoratda talaba bilimini baholashda quyidagi talablar qo‘yiladi:

Baho		**Talaba bilimi va malakasiga talablar**
5	A’lo	Talaba test savollarining 90%dan yuqorisiga to‘g‘ri javob beradi hamda amaliy, laboratoriya, seminar mashg‘ulotlar va mustaqil ta’lim topshiriqlari bajarishi shuningdek mashg‘ulotlarda faol

		ishtirok etagan deb topilganda.
4	Yaxshi	Talaba test savollarining 89,9%dan 70% gacha to‘g‘ri javob beradi hamda amaliy, laboratoriya, seminar mashg‘ulotlar va mustaqil ta’lim topshiriqlari bajarishi hamda mashg‘ulotlardagi yaxshi ishtirokni inobatga olgan deb topilganda
3	Qoniqarli	Talaba test savollarining 69,9%dan 60% gacha to‘g‘ri javob beradi hamda amaliy, laboratoriya, seminar mashg‘ulotlar va mustaqil ta’lim topshiriqlari bajarishi hamda mashg‘ulotlarda ishtirok etib topshiriqlarni o‘qituvchi yordamida to‘g‘ri bajaradi deb topilganda
2	Qoniqarsiz	Talaba test savollarining 59,9% dan kamiga to‘g‘ri javob beradi hamda amaliy, laboratoriya, seminar mashg‘ulotlar va mustaqil ta’lim topshiriqlari bajarishi hamda mashg‘ulotlardagi topshiriq-larning shartini to‘g‘ri tushunib ularni bajara olmaydi deb topilganda

Fan bo‘yicha oraliq nazorat turlarida talaba olgan baho yakuniy nazorat turiga kirish uchun.
Nizomda ko‘rsatib o‘tilgandek, talaba uzrli sabablarga ko‘ra oraliq nazorat turiga kirmagan taqdirda ushbu talabaga oraliq nazorat turini qayta topshirishga fakultet dekanining farmoyishi asosida ruxsat beriladi.
Talabaga oraliq nazorat turini qayta topshirish uchun berilgan muddat davomida talaba tomonidan qayta topshirishlar soni 2 martadan ko‘p bo‘lmasligi kerak.
Talaba oraliq nazorat turini birinchi marta qayta topshirishdan o‘ta olmagan taqdirda, fakultet dekani tomonidan komissiya tuziladi. Komissiya tarkibi tegishli fan bo‘yicha professor-o‘qituvchi va soha mutaxassislari orasidan shakllantiriladi.
Ikkinchi marta oraliq nazorat turini o‘tkazish va talabani baholash mazkur komissiya tomonidan amalga oshiriladi.
Berilgan muddat davomida mavjud bo‘lgan qarzdorlikni topshira olmagan talaba bo‘yicha fakultet dekani bildirgi bilan oliy ta’lim muassasasi rektorini xabardor qiladi va ushbu talaba rektor buyrug‘i asosida kursdan qoldiriladi.

II.2. Yakuniy nazorat turi

Yakuniy nazorat turi semestr yakunida tegishli fan bo‘yicha talabaning nazariy bilim va amaliy ko‘nikmalarini o‘zlashtirish darajasini aniqlash maqsadida o‘tkaziladi.
Yakuniy nazorat turini o‘tkazish shakli tegishli fan bo‘yicha kafedra tomonidan belgilanadi.
Yakuniy nazorat turi oliy ta’lim muassasasining tegishli fakultet dekani yoki o‘quv-uslubiy bo‘lim tomonidan ishlab chiqiladigan hamda o‘quv ishlari bo‘yicha prorektor tomonidan tasdiqlanadigan Yakuniy nazorat turlarini o‘tkazish jadvalga muvofiq o‘tkaziladi.
Yakuniy(yozma, og‘zaki) nazoratda talaba bilimini baholashda quyidagi talablar qo‘yiladi:

Baho		**Talaba bilimi va malakasiga talablar**
5	A’lo	Savollardagi mavzularning barchasiga asoslangan, ilmiy xatoliklarga yo‘l qo‘yilmagan holda javoblar beradi, mavzu materiali mohiyatini to‘la tushunib yetgan bo‘ladi, ijodiy fikr yuritadi, mustaqil mushohada qiladi, nazariy bilimlarni amalda qo‘llashga misollar keltira oladi, xulosa va qarorlar qabul qilishda faol bo‘ladi, material bo‘yicha to‘la tasavvurga ega deb topilganda.
4	Yaxshi	Savollarning barchasiga to‘liq javob beradi, juz’iy xatoliklarga yo‘l qo‘ymaydi. Material mohiyatni tushunib yetgan bo‘ladi, ijodiy fikr yurita oladi, nazariy bilimlarni amaliy ahamiyatini anglab yetgan bo‘ladi, material bo‘yicha tasavvurga ega bo‘lsa

		qo‘shimcha adabiyotlardan mustaqil foydalana olish qobiliyatiga ega deb topilganda
3	Qoniqarli	Savollarga javoblar yozadi, yo‘l qo‘ygan xatolari juz’iy xatolar bo‘lmaydi, material mohiyatini tushungan bo‘ladi, nazariy bilimlarni amaliy ahamiyatini anglagan bo‘ladi, mavzular bo‘yicha tasavvurga ega bo‘ladi va auditoriya mashg‘ulotlariga to‘liq qatnashgan bo‘ladi hamda fan (mavzu) bo‘yicha tasavvurga ega deb topilganda
2	Qoniqarsiz	Savollarga javob berishga qiynaladi, material mohiyatini tushunmaydi, tasavvuri sayoz bo‘ladi, nazariy bilimlarni amaldagi ahamiyatni anglab yetmaydi, savollarni ko‘pchiligiga javob bera olmaydi va darslarga muntazam qatnashmagan bo‘laydi hamda fan (mavzu) bo‘yicha tasavvurga ega emas deb topilganda

Yakuniy(test) nazoratda talaba bilimini baholashda quyidagi talablar qo‘yiladi:

Baho		**Talaba bilimi va malakasiga talablar**
5	A’lo	Talaba test savollarining 90%dan yuqorisiga to‘g‘ri javob berganda. Test savollari joriy semestrdagi mavzularning barchasini qambrab olgan bo‘lishi lozim va talaba test materiali mohiyatini to‘la tushunib yetgan bo‘ladi, ijodiy fikr yuritadi, mustaqil mushohada qiladi, nazariy bilimlarni amalda qo‘llaydi, xulosa va qarorlar qabul qilishda faol bo‘ladi, test material bo‘yicha to‘la tasavvurga ega deb topilganda.
4	Yaxshi	Talaba test savollarining 89,9%dan 70%gachasiga to‘g‘ri javob berishi lozim. Test savollari joriy semestrdagi mavzularning barchasini qambrab olgan bo‘lishi lozim va talaba test materiali mohiyatini tushunib yetgan bo‘ladi, test savollariga javob berishda o‘zining fikr va mulohazalariga tayanadi, nazariy bilimlarni amaliy ahamiyatini anglab yetgan bo‘ladi, test material bo‘yicha yaxshi tasavvurga ega deb topilganda
3	Qoniqarli	Talaba test savollarining 69,9%dan 60%gachasiga to‘g‘ri javob berishi lozim. Test savollari joriy semestrdagi mavzularning barchasini qambrab olgan bo‘lishi lozim va talaba test material mohiyatini tushungan bo‘ladi, nazariy bilimlarni amaliy ahamiyatini anglagan bo‘ladi, test material bo‘yicha o‘rtacha tasavvurga ega deb topilganda
2	Qoniqarsiz	Talaba test savollarining 59,9%dan kamiga to‘g‘ri javob berganda. Test savollari joriy semestrdagi mavzularning barchasini qambrab olgan bo‘lishi lozim va talaba test savollarga javob berishga qiynaladi, material mohiyatini tushunmaydi, tasavvuri sayoz bo‘ladi, nazariy bilimlarni amaldagi ahamiyatni anglab yetmaydi, test savollarni kattf qismiga javob bera olmaydi deb topilganda

Yakuniy nazorat turini o‘tkazish va mazkur nazorat turi bo‘yicha talabaning bilimini baholash o‘quv mashg‘ulotlarini olib bormagan professor-o‘kituvchi tomonidan amalga oshiriladi.
Tegishli fan bo‘yicha o‘quv mashg‘ulotlarini olib borgan professor-o‘qituvchi yakuniy nazorat turini o‘tkazishda ishtirok etishi taqiqlanadi.
Yakuniy nazorat turini o‘tkazishda kelishuv asosida boshqa oliy ta’lim muassasalarining tegishli fan bo‘yicha professor-o‘qituvchilari jalb qilinishi mumkin.

Oliy ta’lim muassasasida nazorat turlarini o‘tkazilishi tegishli oliy ta’lim muassasasining ta’lim sifatini nazorat qilish bo‘limi tomonidan doimiy ravishda o‘rganib boriladi. Bunda nazorat turlarini o‘tkazilish tartibi buzilganligi aniqlangan hollarda, o‘tkazilgan nazorat turlarining natijalari bekor qilinishi hamda tegishli nazorat turi qaytadan o‘tkazilishi mumkin.
Yakuniy nazorat turiga kirmagan yoki kiritilmagan, shuningdek ushbu nazorat turi bo‘yicha “2” (qoniqarsiz) baho bilan baholangan talaba akademik qarzdor hisoblanadi.
Nizomning 23 – bandiga asosan, talaba uzrli sabablarga ko‘ra yakuniy nazorat turiga kirmagan taqdirda ushbu talabaga yakuniy nazorat turini qayta topshirishga fakultet dekanining farmoyishi asosida ruxsat beriladi.
Bir kunda 1 tadan ortiq fan bo‘yicha yakuniy nazorat turi o‘tkazilishiga yo‘l qo‘yilmaydi. Yakuniy nazorat turlarini o‘tkazish kamida 2 kun oralig‘ida belgilanishi lozim.
Yuqorida kelitirib o‘tganimizdek, bitiruvchi kurs bo‘lmagan talabalar kuzgi semestr natijalari bo‘yicha 3 tagacha fandan (fanlardan) akademik qarzdorligi bo‘lgan hollarda talabaga bir oygacha, bahorgi semestr natijalari bo‘yicha 3 tacha fandan (fanlardan) akademik qarzdorligi bo‘lgan talabaga tegishli fan (fanlar) bo‘yicha oralik va (yoki) yakuniy nazorat turlarini yangi o‘quv yili boshidan qayta topshirish uchun 1 oy muddat beriladi.
Bitiruvchi kurs talabalariga bahorgi semestr natijalari bo‘yicha o‘zlashtirmagan fandan (fanlardan) qayta topshirish uchun yakuniy davlat attestatsiyasi boshlangunga qadar ruxsat beriladi.
Fanlardan akademik qarzdorligi 4 ta va undan ko‘p bo‘lgan talabalarga qayta topshirishga ruxsat berilmaydi va ular oliy ta’lim muassasasi rektorning buyrug‘i bilan kursdan qoldiriladi.
Talabaga yakuniy nazorat turini qayta topshirish uchun berilgan muddat davomida talaba tomonidan qayta topshirishlar soni 2 martadan ko‘p bo‘lmasligi kerak.
Talaba yakuniy nazorat turini birinchi marta qayta topshirishdan o‘ta olmagan taqdirda, fakultet dekani tomonidan komissiya tuziladi. Komissiya tarkibi tegishli fan bo‘yicha professor-o‘qituvchi va soha mutaxassislari orasidan shakllantiriladi.
Ikkinchi marta yakuniy nazorat turini o‘tkazish va talabani baholash mazkur komissiya tomonidan amalga oshiriladi.
Berilgan muddat davomida mavjud bo‘lgan qarzdorlikni topshira olmagan talaba bo‘yicha fakultet dekani bildirgi bilan oliy ta’lim muassasasi rektorini xabardor qiladi va ushbu talaba rektor buyrug‘i asosida kursdan qoldiriladi.
Baholash natijasidan norozi bo‘lan talabalar fakultet dekani tomonidan tashkil etiladigan Apellyatsiya komissiyasiga apellyatsiya berishhuquqiga ega.
Apellyatsiya komissiyasi tarkibga talabani baholashda ishtirok etmagan tegishli fan professor-o‘kituvchilari orasidan komissiya raisi va kamida to‘rt nafar a’zo kiritiladi.
Talaba baxolash natijasidan norozi bo‘lgan taqdirda, baholash natijasi e’lon kilingan vaqtdan boshlab 24 soat davomida apellyatsiya berishi mumkin. Talaba tomonidan berilgan apellyatsiya Apellyatsiya komissiyasi tomonidan 2 kun ichida kurib chiqilishi lozim.
Talabaning apellyatsiyasini ko‘rib chiqishda talaba ishtirok etish huquqiga ega.
Apellyatsiya komissiyasi talabaning apellyatsiyasini ko‘rib chiqib, uning natijasi bo‘yicha tegishli qaror qabul qiladi. Qarorda talabaning tegishli fanni o‘zlashtirgani yoki o‘zlashtira olmagani ko‘rsatiladi.
Apellyatsiya komissiyasi tegishli qarorni fakuљtet dekani va talabaga yetkazilishini ta’minlaydi.

III. Baxolash natijalarini qayd qilish

Nizomning 35-bandida keltirilganidek, talabalar bilimini baholash tegishli fan bo‘yicha professor-o‘kituvchi tomonidan Talabalarning fanlarni o‘zlashtirishini hisobga olish jurnalida (bundan buyon matnda Jurnal deb yuritiladi) qayd etib boriladi. Professor-o‘qituvchi qo‘shimcha ravishda talabalar bilimini baholashni elektron tizimda ham yuritishi mumkin.
Professor-o‘qituvchi Jurnalda talabaga qo‘yilgan baholarni shu kunning o‘zida qayd etib boradi. Agar talabaning bilimini baholash yozma ish shaklida o‘tkazilgan bo‘lsa, bunda professor-

o‘qituvchi talabalarning natijalarini 3 kundan ko‘p bo‘lmagan muddatda Jurnalga qayd etishi lozim.
Nazorat turi bo‘yicha talabaning bilimi “3” (qoniqarli) yoki “4” (yaxshi) yoxud ”5” (a’lo) baho bilan baholanganda, nazorat turini qayta topshirishga yo‘l qo‘yilmaydi.
Talaba nazorat turi o‘tkazilgan vaqtda uzrli sabablarsiz qatnashmagan hollarda Jurnalga ”0” belgisi yozib qo‘yiladi.
Jurnal tegishli fan bo‘yicha o‘quv mashg‘ulotlarini olib borgan professor-o‘qituvchi, kafedra mudiri va fakultet tomonidan imzolanadi hamda fakultet dekanatida saqlanadi. Jurnalning saqlanishi uchun fakultet dekani mas’ul hisoblanadi.
Talabalarning yakuniy nazorat turi bo‘yicha baholari Jurnalga kayd etilganda, shu kunning o‘zida talabaning Baholash daftariga ham yozib qo‘yilishi kerak.
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.
Jurnalning o‘z vaqtida, to‘g‘ri va to‘liq yuritilishi, shuningdek undagi baho va boshqa ma’lumotlarga asossiz o‘zgartirishlar kiritilmasligi uchun fakultet dekani va tegishli fan bo‘yicha professor-o‘qituvchi mas’ul hisoblanadi.
Tegishli o‘quv yili yakuni bo‘yicha ishchi o‘quv rejagi fanlar bo‘yicha “3” (qoniqarli) yoki “4” (yaxshi) yoxud “5” (a’lo) baho bilan baholangan talaba oliy ta’lim muassasasi rektorining buyrug‘iga asosan keyingi kursga o‘tkazadi.
Baholash natijalari kafedra yig‘ilishlari, fakultet va oliy ta’lim muassasasi Kengashlarida muntazam ravishda muhokama etib boriladi va tegishli qarorlar qabul qilinadi.

Tavsiya etilgan adabiyotlar ro‘yxati

Asosiy adabiyotlar

8. A. R. Azamatov, B. Boltayev. Algoritmlash va dasturlash asoslari. 0‘quv qo‘llanma. T. : “Cho‘lpon”, 2010 y.
9. A. R. Azamatov, B. Boltayev. Algoritmlash va dasturlash asoslari. 0‘quv qo‘llanma. T. : “Cho‘lpon”, 2013 y.
10. M.A.Aripov, N.A.Otaxonov. Dasturlash asoslari. O`quv qo`llanma. T.: “Tafakkur bo`stoni”, 2015 yil.
11. М. Ашуров, М. Мирмахмудов, Ш. Сапаев. Замонавий дастурлаш тиллари фанидан лаборатория ишлари. Т. : ТДПУ, 2008 й.
12. Меняев Михаил Федорвич. Информационные технология управления. Москва, «Издательский Омегал», 2003 г.
13. S.Tursunov, I.Nazarov. Ta’limda axborot texnologiyalari. Darslik Toshkent. “Adabiyot uchqunlari” 2019-yil 1 tom, 262-b.
14. S.Tursunov, I.Nazarov. Ta’limda axborot texnologiyalari. Darslik Toshkent. “Adabiyot uchqunlari” 2019-yil 1 tom, 300-b.

Qo‘shimcha adabiyotlar

17. Мирзиёев Шавкат Миромонович. Эркин ва фаровон, демократик Узбекистан давлатини биргаликда барпо этамиз. Узбекистан Республикаси Президента лавозимига киришиш тантанали маросимига батишланган Олий Мажлис палаталарининг кушма мажлисидаги нутк / Ш.М. Мирзиёев. - Тошкент : Узбекистан, 2016. - 56 б.
18. Мирзиёев Шавкат Миромонович. Танкидий тахдил, катьий тартиб-интизом ва шахсий жавобгарлик - хар бир рахбар фаолиятининг кундалик коидаси булиши керак. Мамлакатимизни 2016 йилда ижтимоий-иктисодий ривожлантиришнинг асосий якунлари ва 2017 йилга мулжалланган иктисодий дастурнинг энг мухим устувор йунапишларига

батишланган Вазирлар Махкамасининг кенгайтирилган мажлисидаги маъруза, 2017 йил 14 январ / Ш.М. Мирзиёев. - Тошкент : Узбекистан, 2017. - 104 б.
19. Мирзиёев Шавкат Миромонович. К,онун устуворлиги ва инсон манфаатларини таъминлаш - юрт тараккиёти ва халк фаровонлигининг гарови. Узбекистан Республикаси Конституцияси кабул килинганининг 24 йиллигига багишланган тантанали маросимдаги маъруза. 2016 йил 7 декабр /Ш.М.Мирзиёев. - Тошкент: “Узбекистан”, 2017. - 48 б.
20. Мирзиёев Шавкат Миромонович. Буюк келажагимизни мард ва олижаноб халкимиз билан бирга курамиз. Мазкур китобдан Узбекистан Республикаси Президенти Шавкат Мирзиёевнинг 2016 йил 1 ноябрдан 24 ноябрга кадар Коракалпогистон Республикаси, вилоятлар ва Тошкент шахри сайловчилари вакиллари билан **утказилган** сайловолди учрашувларида сузлаган нуткдари урин олган. /Ш.М.Мирзиёев. - Тошкент: : “Узбекистан”, 2017. - 488 б
21. Узбекистан Республикаси Президентининг Фармони. Узбекистан республикасини янада ривожлантириш буйича харакатлар стратегияси тугрисида. *(Узбекистон Республикаси цонун уужжатлари туплами, 2017 й., 6- сон, 70-модда)*
22. Узбекистон Республикаси Конституцияси. Т.: Узбекистон. 2014. -46 б.
23. П. Дарахвелидзе, Э. Марков. Программирование в Delphi7. Учебник. Санкт-Петербург, “БХВ-Петербург” 2003 г.
24. В. М. Пестиков, А. Н. Маслобоев. Turbo PASCAL 7. 0. Изучаем на примерах. Санкт-Петербург. : “БХВ-Петербург”, 2004 г.
25. Фаронов В. В. Программирование на языке высокого уровня Delphi. Учебник. М. : “Питер”, 2003 г.
26. В.Т.Безручко. Практикум по курсу информатики. М. : «Финансы и статистика», 2004 г.
27. Nazirov Sh.A., Qobulov R.V., Bobojanov M.R., Raxmanov Q.S. C va C++ tili. “Voris-nashriyot” MCHJ, Toshkent 2013. 488 b.
28. П. Дарахвелидзе, Э. Марков. Программирование в Delphi7. Учебник. Санкт-Петербург, “БХВ-Петербург” 2003 г.
29. Фаронов В. В. Программирование на языке высокого уровня Delphi. Учебник. М. : “Питер”, 2003 г.
30. В. Т. Безручко. Практикум по курсу информатики. М. : «Финансы и статистика», 2004 г.
31. The C++ Programming Language, Third Edition by Bjarne Stroustrup. AT&T Labs. Murray Hill, New Jersey.
32. Культин Н.Б. C++ Builder в задачах и примерах.- СПб.: БХВ- Петербург, 2005. — 336 с: ил.
33. Peter Gottschling. Discovering Modern C++. An Intensive Course for Scientists, Engineers, and Programmers. “Addison-Wesley”, 2015 y.

Elektron ta’lim resurslari

6. www.zivonet.iiz - **Axborot ta’lim portal**
7. www.edu.uz - **Oliy va o‘rta maxsus ta’lim vazirligi portal**
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.timus.ru/ - dasturlarni testlovchi tizim.

Testlar

№	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'zgaruvchilar	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 b-n 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

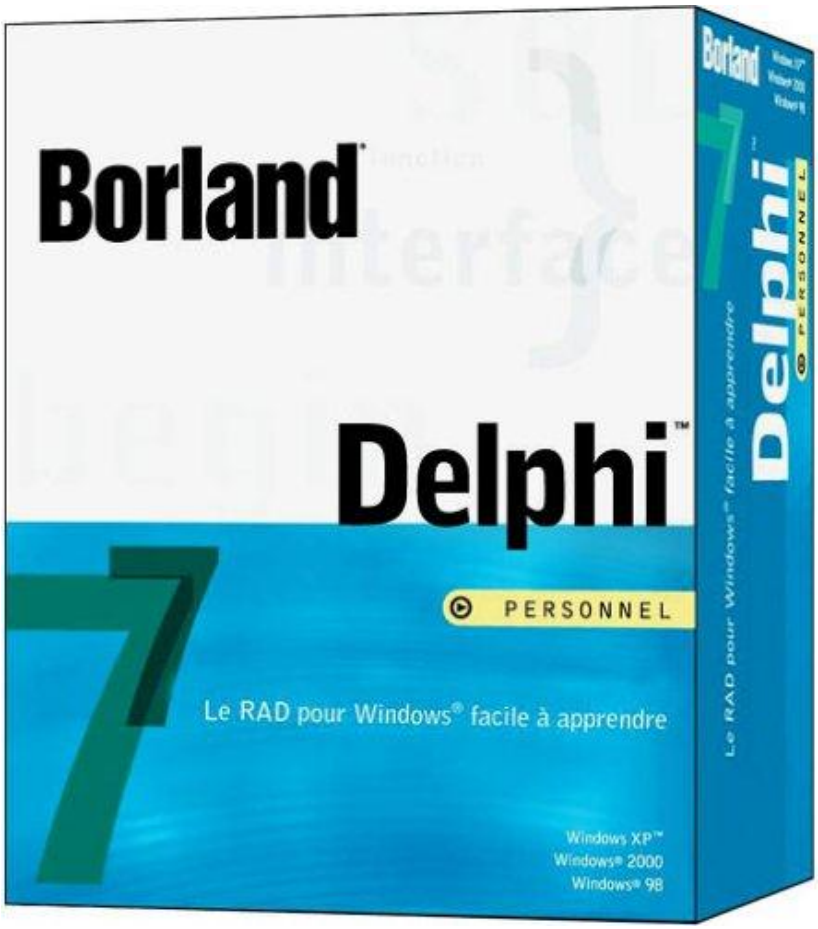
49	….- funksiyasini har qanday sikl operatoriga qo`llash mumkin. Bu funksiya sikl parametrining keyingi qiymatni qabul qilishini taminlaydi. Boshqacha so`z bilan aytganda sikl tanasi tugatiladi. Bunda siklning o`zi tugatilmaydi.	* continue	for	repeat	While
50	Switch Case qanday operator?	* Kalit bo’yicha tanlash	Shartsiz	Modul	Takrorlanish
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

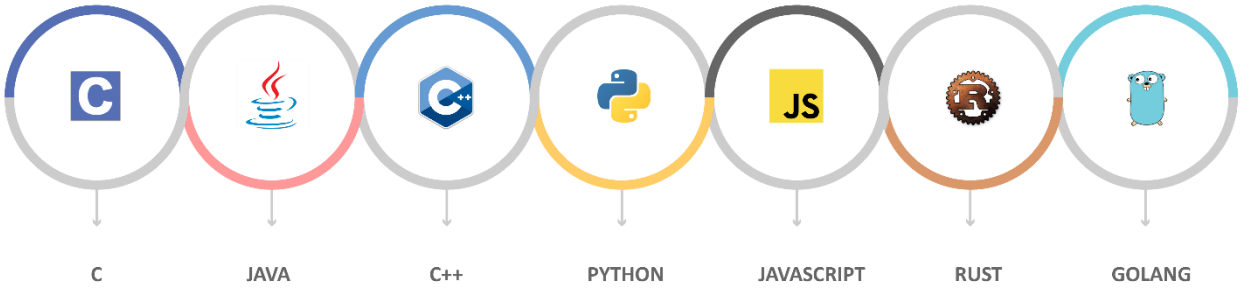
82	Delete funksiyaning qiymati qaysi tipga tegishli.	*String	Intejer	Real	Char
83	Fiksirlangan ustun rangi qanday nomlanadi?	* Fixed cols	Fixed Color	Key option	Strings
84	Fayl o’zgaruvchisi bilan asosiy fayl orasida aloqa o’rnatadigan funksiya.	*Assign	Label	Close	Type
85	Dinamik bog`lanuvchi bibliotekalar so`zining qisqartmasi.	*DLL	GLL	LLL	BLL
86	Ma`lum shartlarga muvofiq bajariladigan algoritm qanday.	*tarmoqlanuvchi	takrorlanuvchi	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

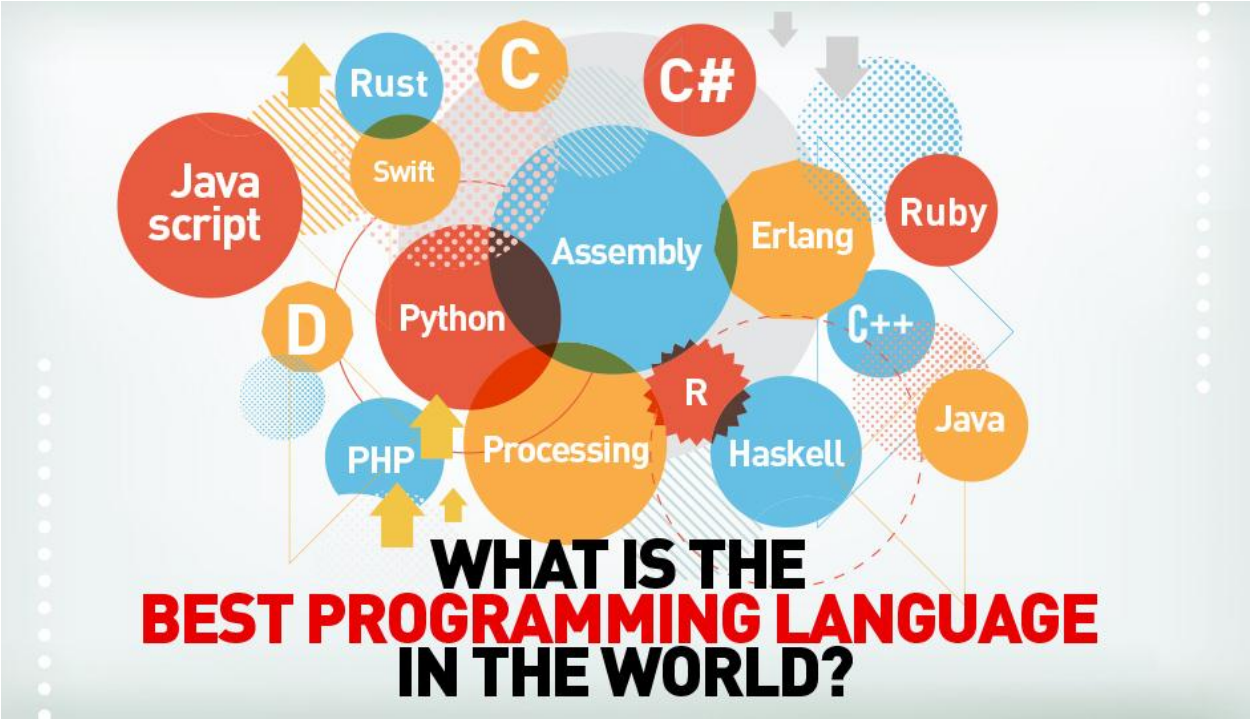
98	Paskal tilida haqiqiy o`zgarmaslar necha xil ko`rinishda tasvirlanadi	*2 xil	3 xil	1 xil	5 xil
99	Quyidagi javoblarning qaysi birida qo`zg`almas konstanta ko`rsatilgan?	*45.6254	-0.2865E-5	5.28E6	-2.65E5
100	Mantiqiy ma'lumotlar turi tog`ri yozilgan javobni ko`rsating	* BOOLEAN	REAL	INTEGER	CHAR

Tarqatma materiallar

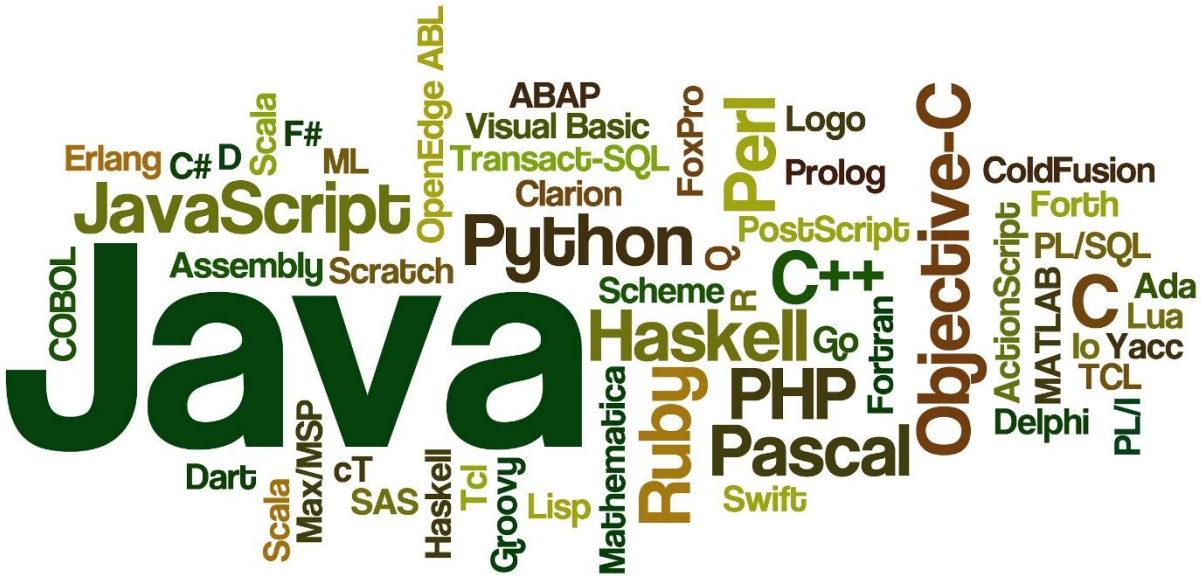


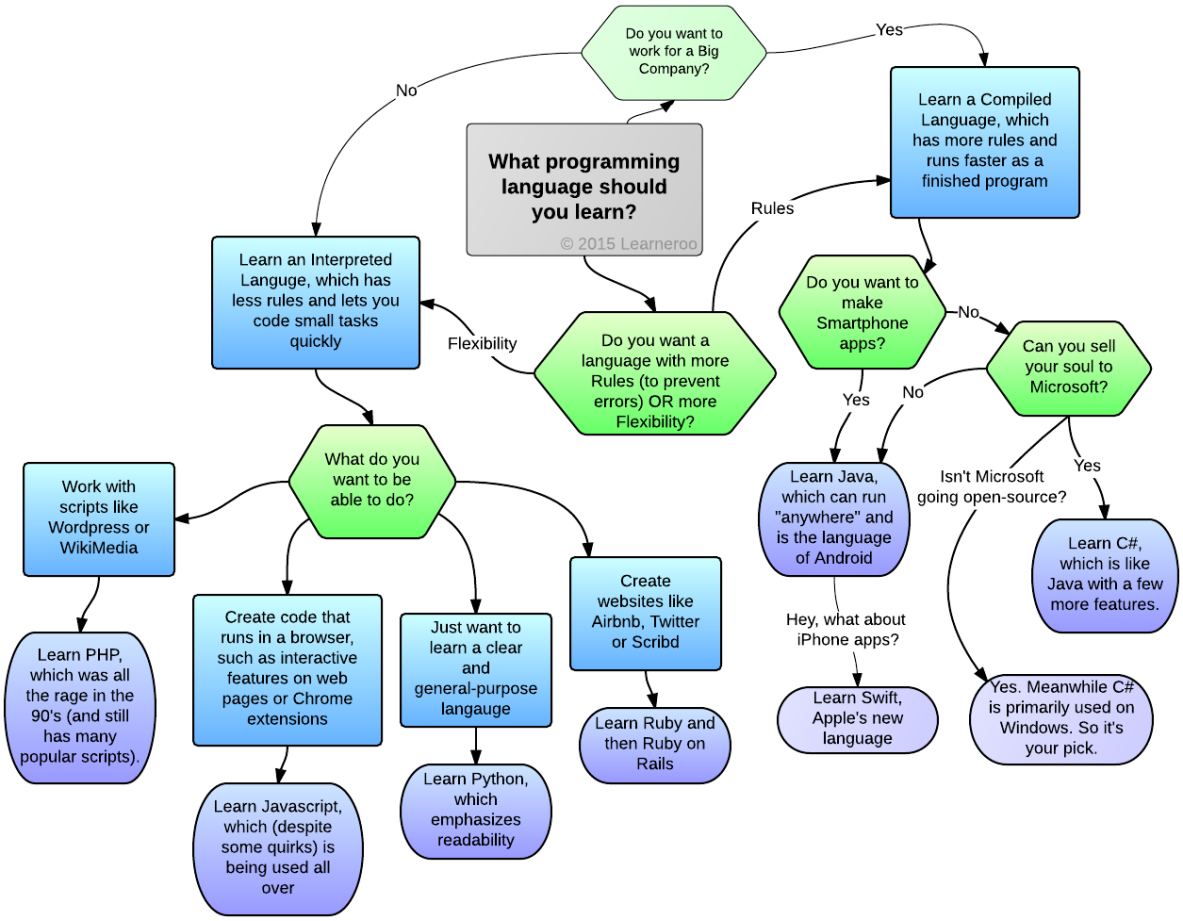
Ob'ektga yo'naltirilgan dasturlash tillari	Strukturali dasturlash tillari	Mashina tillari
Delphi	Basic	Assembler
C++	Fortran	Mashina kodi (0 ва 1)
Java	COBOL	
C# (C sharp)	Pascal	
PHP (Personal home page)	C	
....	....	....





Rank	TIOBE	IEEE	Stack Overflow	PYPL Index
1	Java	C	JavaScript	Java
2	C	Java	SQL	Python
3	C++	Python	Java	PHP
4	C#	C++	C#	C#
5	Python	R	PHP	JavaScript
6	PHP	C#	Python	C
7	JavaScript	PHP	C++	C++
8	Visual Basic NET	JavaScript	Angular JS	Objective–C
9	Delphi/Object Pascal	Ruby	Node.js	R
10	Perl	Go	C	Swift


Java
JavaScript
Python
Perl
C++
C
Objective-C
PHP
Pascal
Haskell
Ruby
Assembly
Scratch
Logo
Prolog
ColdFusion
Forth
PL/SQL
Ada
Lua
Yacc
TCL
Delphi
PL/I
ActionScript
MATLAB
Fortran
Go
Scheme
R
Q
PostScript
ABAP
Visual Basic
Transact-SQL
Clarion
FoxPro
OpenEdge ABL
Erlang
C#
D
Scala
F#
ML
COBOL
Dart
cT
SAS
Max/MSP
Scala
Haskell
Tcl
Groovy
Lisp
Mathematica
Swift


What programming language should you learn?
© 2015 Learneroo
Do you want to work for a Big Company?
Yes
No
Learn a Compiled Language, which has more rules and runs faster as a finished program
Learn an Interpreted Languge, which has less rules and lets you code small tasks quickly
Rules
Flexibility
Do you want a language with more Rules (to prevent errors) OR more Flexibility?
Do you want to make Smartphone apps?
No
Yes
Can you sell your soul to Microsoft?
No
Yes
What do you want to be able to do?
Work with scripts like Wordpress or WikiMedia
Learn PHP, which was all the rage in the 90's (and still has many popular scripts).
Create code that runs in a browser, such as interactive features on web pages or Chrome extensions
Learn Javascript, which (despite some quirks) is being used all over
Just want to learn a clear and general-purpose langauge
Learn Python, which emphasizes readability
Create websites like Airbnb, Twitter or Scribd
Learn Ruby and then Ruby on Rails
Learn Java, which can run "anywhere" and is the language of Android
Hey, what about iPhone apps?
Learn Swift, Apple's new language
Isn't Microsoft going open-source?
Yes
Learn C#, which is like Java with a few more features.
Yes. Meanwhile C# is primarily used on Windows. So it's your pick.