O‘ZBEKISTON RESPUBLIKASI
OLIY TA’LIM, FAN VA INNOVATSIYALAR VAZIRLIGI
GULISTON DAVLAT UNIVERSITETI

Z.T. Negmatulloyev

ALGORITMIK TILLAR VA DASTURLASH

O‘QUV QO‘LLANMA

(60540200 –Amaliy matematika ta‘lim yo‘nalishi talabalari uchun)

Guliston-2023

Z.T. Negmatulloyev, Algoritmik tillar va dasturlash, //o‘quv qo‘llanma, Guliston, 2023 y.-177 b.

Muallif: **Negmatulloyev Zafardjon Turdibekovich** - *Guliston davlat universiteti “Amaliy matematika va axborot texnologiyalari” kafedrasi dotsenti, tex.f.f.d. (PhD).*

O‘quv qo‘llanma amaldagi “Algoritmik tillar va dasturlash” fani bo‘yicha namunaviy dasturlar asosida tayyorlangan bo‘lib, 60540200-Amaliy matematika ta‘lim yo‘nalishda tahsil olayotgan talabalar uchun mo‘ljallangan. Unda “Algoritmik tillar va dasturlash” fani bo‘yicha nazariy va amaliy materiallar, topshiriqlar, bilimlarni nazorat qilish savollar, mustaqil ishlash uchun topshiriqlar majmuasi keltirilgan.

Учебное пособие подготовлено на основе действующих образцов программ по направлению «Алгоритмические языки и программирование» и предназначено для студентов, обучающихся по направлению образования 60540200-Прикладная математика. В нем представлен теоретические и практические материалы, задания, вопросы для контроля знаний, комплекс заданий для самостоятельной работы по дисциплине ”Алгоритмические языки и программирование".

The textbook has been prepared on the basis of existing samples of programs in the direction "Algorithmic languages and programming" and is intended for students studying in the direction of education 60540200-Applied Mathematics. It presents theoretical and practical materials, assignments, questions for knowledge control, a set of tasks for independent work in the discipline "Algorithmic languages and programming".

Taqrizchilar:

Qalandarov A.A.	Guliston davlat pedagogika inistituti o‘quv ishlari bo‘yicha prorektor, f. m.f.b.f.d. (PhD), dotsent.
Abdurahimov D.B.	Guliston davlat universiteti dotsenti, pedagogika fanlari nomzodi.

KIRISH

Bugungi kunda jamiyatda tobora osib borayotgan axborot oqimi, axborot texnologiyalarining turli tumanligi, kompyuterda yechiladigan masalalarning murakkablashuvi ushbu texnologiyalardan foydalanuvchilarning oldiga bir qator vazifalarni qoyadi. Kundalik hayotimizda ma‘lumotlar oqimining kopligi tufayli ularni qisqa vaqt ichida qayta ishlash muommosi ham ortib bormoqda. Hozirgi vaqtda axborot-kommunikasiya vositalari barcha turdagi tashkilot va muassasalarga shiddat bilan kirib kelmoqda. Axborotlarning haddan tashqari kopligi bu axborotlarni saqlashda, qayta ishlashda, hamda har xil turdagi tizimlarni yaratish, ulardan keng foydalanishni va axborot tizimlari yaratishni talab qiladi.

O‘zbekiston Respublikasi Prezidentining 2020 yil 6 oktyabrdagi PQ-4851-son «Axborot texnologiyalari sohasida ta’lim tizimini yanada takomillashtirish, ilmiy tadqiqotlarni rivojlantirish va ularni IT-industriya bilan integratsiya qilish chora-tadbirlari to‘g’risida»gi qarorlari hamda mazkur faoliyatga tegishli boshqa me’riy-huquqiy xujjatlarda belgilangan vazifalar jamiyat rivojlanishi istiqbolini strategik rejalashtirish tizimiga sifat jihatdan yangi yondashuvlarni boshlab berdi.

Ushbu o‘quv qo‘llanmani yozishdan maqsad C++ tilida dasturlashning asosiy tamoyillarini ixcham va aniq taqdim etishdir. Materialni tanlashda amalda eng ko‘p ishlatiladigan dizaynlarga e’tibor qaratildi. Shuning uchun, bu erda keltirilgan C++ dasturlash tilining taqdimoti to‘liq tavsif deb da’vo qilmaydi, lekin muallifning fikriga ko‘ra, undan talabalar o‘rta darajadagi ilovalarni yaratish uchun etarli malakaga ega bo‘ladilar.

Dasturlash - bu turli sohalardagi muammolarni yechimlarini kompyuterda amalga oshirish uchun ifodalash san’atidir. Dasturchining asosiy sa’y-harakatlari yechimni topish va aniqlashtirishga qaratilgan bo‘lib, ko‘pincha muammoni to‘liq tushunish faqat uning yechimini dasturlash jarayonida bo‘ladi.

Dasturlarni tez va sifatli yozish hozirgi kunda katta ahamiyat kasb etmoqda. Buni ta’minlash uchun obyektli dasturlash g’oyasi ilgari surildi. Huddi 1970 yillar

boshida strukturali dasturlash kabi, dasturlarni hayotdagi jismlarni modellashtiruvchi obyektlat orqali tuzish dasturlash sohasida inqilob qildi.

Hozirgi kunda juda ko‘p algoritmik tillar mavjud. Bular ichida Java, C++, C# (Si sharp) va Payton dasturlash tillari universal tillar hisoblanib, boshqa tillarga qaraganda imkoniyatlari kengroqdir. So‘ngi yillarda Java, C++,C# va Payton dasturlash tillari juda takomillashib, tobora ommalashib bormoqda. Mazkur tillardagi vositalar zamonaviy kompyuter texnologiyasining hamma talablarini o‘z ichiga olgan va unda dastur tuzuvchi uchun ko‘pgina qulayliklar yaratilgan.

Axborot kommunikatsion texnonologiyalarini taraqqiy etishida bevosita dasturlash tillarining o‘rni beqiyos. Ayniqsa, hozirgi davrga kelib C++, C#, Java va Payton dasturlash tillari yordamida shaxsiy kompyuterlar uchun amaliy dasturiy to‘plamlardan tashqari SmartPhone va Planshetlar uchun iOS, Android, Windows mobile, Symbian kabi operatsion tizimlar va ilovalar yaratilmoqda.

C++ dasturlash tili tarkibida bir nechta imkoniyatlar mavjud, ya’ni consol rejimi, forma ob’yekt rejimi, grafik muhiti va ma’lumotlar bazasi bilan ishlash imkoniyatlari keng joriy etilgan. Ushbu qo‘llanmada keltirilgan misol va masalalarning yechimi, ya’ni dasturining *int main ()* funksiyasi tarkibini C++ dasturlash tilinining ixtiyoriy versiyalarida ishlatib ko‘rish mumkin. Qo‘llanma oliy oquv yurtlari talabalari va magistrantlari, litsey va kasb hunar kollej oquvchilari hamda mustaqil organuvchilar uchun qulay vosita hisoblanadi.

Ushbu taklif etilayotgan qo‘llanma asosan C++ dasturlash tilini o‘rganmoqchi bo‘lganlar uchun mo‘ljallangan. Shu sababli qo‘llanmada C++ tiliga bog’liq boshlang’ich ma’lumotlar yoritilgan. Bu qo‘llanmadan C++ dasturlash tilini o‘rganuvchilar, dastur tuzishni o‘rganayotganlar hamda “Dasturlash asoslari”, “Zamonaviy dasturlash tillari” fanlaridan olingan nazariy bilimlarni mustahkamlash uchun foydalanishlari hisobga olingan. Ushbu qo‘llanmaga kiritilgan ma’lumotlar dasturlashning bazaviy kursidagi deyarli barcha bo‘limlarini, ya’ni skalyar turlar va boshqaruv operatorlaridan tortib, ma’lumotlarning murakkab turlari kabilarni o‘z ichiga oladi.

I-BOB. C++ DASTURLASH TILI VA UNING ASOSIY TUSHUNCHALARI

1.1. C++ dasturlash tili va uning tuzilmasi

Hozirgi kunda juda ko‘p algoritmik tillar mavjud. Bular ichida Java, C++ va **C#** (Si sharp) dasturlash tillari universal tillar hisoblanib, boshqa tillarga qaraganda imkoniyatlari kengroqdir. So‘ngi yillarda Java, C++ va C# dasturlash tillari juda takomillashib, tobora ommalashib bormoqda. Mazkur tillardagi vositalar zamonaviy kompyuter texnologiyasining hamma talablarini o‘z ichiga olgan va unda dastur tuzuvchi uchun ko‘pgina qulayliklar yaratilgan.

C++ 1980 yillar boshida Bjarne Stroustrup tomonidan C tiliga asoslangan tarzda tuzildi. **C++** juda ko‘p qo‘shimchalarni o‘z ichiga olgan,lekin eng asosiysi u obyektlar bilan dasturlashga imkon beradi.

Dasturlarni tez va sifatli yozish hozirgi kunda katta ahamiyat kasb etmoqda. Buni ta’minlash uchun obyektli dasturlash g’oyasi ilgari surildi. Huddi 1970 yillar boshida strukturali dasturlash kabi, dasturlarni hayotdagi jismlarni modellashtiruvchi obyektlat orqali tuzish dasturlash sohasida inqilob qildi.

C++ funksiya va obyektlarning juda boy kutubhonasiga ega. Yani **C++** dasturlash tilida dasturlashni o‘rganish ikki qismga bo‘linadi. Birinchisi bu **C++** tilini o‘zini o‘rganish, ikkinchisi esa **C++** ning standart kutubhonasidagi tayyor obyekt va funksiyalarni qo‘llashni o‘rganishdir.

C++ tiliga ko‘plab yangiliklar kiritilgan bo‘lib, tilning imkoniyati yanada kengaytirilgan. **C++** dasturlash tili ham boshqa dasturlash tillari kabi o‘z alfavitiga va belgilariga ega.

- **Tillarda mavjud alfavit va leksemalarga quyidagilar kiradi:**
  1. Katta va kichik lotin alfaviti harflari;
  2. Raqamlar - 0,1,2,3,4,5,6,7,8,9;
  3. Maxsus belgilar: " {} | [] () + - / % \ ; ‘ : ? <=>_ ! & ~ # ^ . *

- **Alfavit belgilaridan tilning leksemalari shakllantiriladi:**
  1. Identifikatorlar;
  2. Kalit (xizmatchi yoki zahiralangan) so‘zlar;
  3. O‘zgarmaslar;
  4. Amallar belgilanishlari;
  5. Ajratuvchilar.

1. Dastur tuzilmasi

C++ dasturlash tilida dastur quyidagi tarkibda tashkil topadi:

Direktivalar – funksiyalar kutubxonasini chaqirish. Ular maxsus **include** katalogida joylashgan va **.h** kengaytmali fayllar bo‘ladi. **C++** tilida masalaning qo‘yilishiga qarab kerakli kutubxonalar chaqiriladi. Bu esa dasturning xotirada egallaydigan joyini minimallashtiradi.

Masalan, ma’lumotlarni kiritish-chiqarish proseduralari uchun:

#include <stdio.h> tizimdan chaqirish
#include “stdio.h” joriy katalogdan chaqirish.

C++ dasturlash tili bilan ishlovchi eng sodda dasturlar Dev C++ va CodeBlocks dasturlaridir. Ularning tarkibida 300 dan ortiq kutubxonalar mavjud. Eng ko‘p ishlatiladigan kutubxonalar quyidagilar:

#include<iostream.h>,
#include <math.h>
#include <conio.h>
#include <graphics.h>
#include <memory.h> va boshqalar

Makroslar (#define) – dastur bajarilishi davomida o‘zgaruvchi ko‘rsatilgan qiymatni qabul qilishi uchun (const). Unda makroning nomi va qiymati ko‘rsatiladi.

Масалан:

```
#define  pi  3.1415
#define  x  556
#define  s[100]
#define M  x*x*x
```

main () funksiyasi-asosiy degan ma’noni anglatadi. Bu funksiya **“{“** belgisidan boshlanadi va dasturning asosini tashkil etuvchi o‘zgaruvchilarningtoifalari ko‘rsatiladi. Dastur **“}”** belgisi bilan yakunlanishi shart. Agar dasturda qism dasturlardan foydalanilayotgan bo‘lsa, ularning nomlari va haqiqiqy parametrlari keltiriladi. So‘ngra dasturning asosiy buyruqlari yoziladi. Agar buyruqlar murakkab bo‘lsas, ular alohida **“{ }”** belgilari orasiga olingan bo‘lishi kerak.

C++ tilida dasturning asosi bo‘lmish buyruqlar kichik harflar bilan yoziladi. Buyruqlar nuqta-verguk bilan **(;)** yakunlanadi. Buyruqlar bir qator qilib yozilishi ham mumkin.

C++ dasturlash tilida dastur funksiya va funksiyalardan tashkil topadi.Agar dastur bir nechta funksiyalardan tashkil topgan bo‘lsa, bir funksiyaning nomi main deb nomlanishi shart. Dastur aynan main funksiyasining birinchi operatoridan boshlab bajariladi.

2. Kiritish va chiqarish operatorlari

C++ tilidagi dastur ko‘rinishini quyidagi misol yordamida keltirib o‘tamiz.

```
#include <iostream.h>              // sarlavha faylni qo‘shish
int main  ()                      //  bosh funksiya tavsifi
{                                 //  blok boshlanishi
cout << “Salom Dunyo! “\n”;       //  satrni chop etish
return 0;                         //  funksiya qaytaradigan qiymat
}                                 //  blok tugashi
```

Dasturning 1-satrida #include direktivasi bo‘lib, dastur kodiga oqimli o‘qish/yozish funksiyalari va uning o‘zgaruvchilari e’loni joylashgan **iostream.h** sarlavha faylini qo‘shadi. Keyingi qatorlarda dasturning yagona, asosiy funksiyasi **main()** funksiyasi tavsifi keltirilgan. Shuni qayd etish kerakki, **C++** dasturida albatta **main()** funksiyasi bo‘lishi shart va dastur shu funksiyani bajarish bilan o‘z ishini boshlaydi.

Dastur tanasida konsol rejimi (***Consol*** *– rejimi bu MS DOS oynasi ko‘rinishiga o‘xshash oyna bo‘lib, unda foydalanuvchi dastur tuzuishda faqat dastur kodlari bilan ishlaydi. Graphic interface – rejimida esa faqat tilning kodlari bilangina emas muhitning menyulari, komponentalari bilan ham ishlashi mumkin bo‘ladi)* da belgilar ketma-ketligini oqimga chiqarish amali qo‘llanilgan. Ma’lumotlarni standart oqimga (ekranga) chiqarish uchun quyidagi format ishlatilgan:

cout << <ifoda>;

Bu yerda **<ifoda>** sifatida o‘zgaruvchi yoki sintsksisi to‘g’ri yozilgan va qandaydir qiymat qabul qiluvchi til ifodasi kelishi mumkin (keyinchalik, burchak qavs ichiga olingan o‘zbekcha satr ostini til tarkibiga kirmaydigan tushuncha deb qabul qilish kerak).

cin << a;

Ma’lumotlarni klaviatura yordamida kiritish buyrug’i bo‘lib, u ham **iostream.h** kutubxonasi tarkibidagi funksiya hisoblanadi.

Identifikatorlar va kalit so‘zlar:

Programmalash tilining muhim tayanch tushunchalaridan biri identifikator tushunchasidir.

Identifikator- *identifikator deganda kata va kichik lotin harflari, raqamlar va tag chiziq (‘_’) belgilaridan tashkil topggan va raqamdan boshlanmaydigan belgilar ketma-ketligi tushuniladi.*

Identifikatorlar-*kalit so‘zlar, o‘zgaruvchilar, funksiyalar, nishonlar va boshqa ob’ektlarni nomlashda ishlatiladi.*

C++ tilining kalit so‘zlari quyidagilardan iborat: auto, break, case, char, const, do, if, else, float, for, goto, int, short, void va ..

Dasturlash tillarida dastur bajarilishi vaqtida qiymati o‘zgarmaydigan identifikatorlar **o‘zgarmaslar** deyiladi. **O‘zgarmaslar *beshta guruhga bo‘linadi*** *– butun, haqiqiy (suzuvchi nuqtali), sanab o‘tiluvchi, belgi (literli) va satr («string», literli satr).*

C++ tilida o‘zgarmas **(cons)** – bu fiksirlangan sonni, satrni va belgini ifodalovchi leksema hisoblanadi.

Kompilyator o‘zgarmasni leksema sifatida aniqlaydi, unga xotiradan joy ajratadi, ko‘rinishi va qiymatiga (turiga) qarab mos guruhlarga bo‘ladi.

Butun o‘zgarmaslar: ular quyidagi formatlarda bo‘ladi

- o‘nlik son;

- sakkizlik son;

- o‘n oltilik son.

O‘nlik o‘zgarmas 0 raqamidan farqli raqamdan boshlanuvchi raqamlar ketma-ketligi va 0 hisoblanadi: 0; 123; 7987; 11.

Manfiy o‘zgarmas – bu ishorasiz o‘zgarmas bo‘lib, unga faqat ishorani o‘zgartirish amali qo‘llanilgan deb hisoblanadi.

Sakkizlik o‘ 0 raqamidan boshlanuvchi sakkizlik sanoq sistemasi (0,1,..,7) raqamlaridan tashkil topgan raqamlar ketma-ketligi:

023; 0777; 0.

O‘n oltilik o‘zgarmas 0x yoki 0X belgilaridan boshlanadigan o‘n oltilik

sanoq sistemasi raqamlaridan iborat ketma-ketlik hisoblanadi:

0x1A; 0X9F2D; 0x23.

Harf belgilar ixtiyoriy registrlarda berilishi mumkin.

Kompilyator sonning qiymatiga qarab unga mos turni belgilaydi. Agar tilde belgilangan turlar dastur tuzuvchini qanoatlantirmasa, u oshkor ravishda turni ko‘rsatishi mumkin. Buning uchun butun o‘zgarmas raqamlari oxiriga, probelsiz l yoki **L (1ong),** u yoki **U (unsigned)** yoziladi. Zarur hollarda bitta o‘zgarmas

uchun bu belgilarning ikkitasini ham ishlatish mumkin: 451u, 012Ul, 0xA2L.

Haqiqiy o‘zgarmaslar: Haqiqiy o‘zgarmaslar – suzuvchi nuqtali son bo‘lib, u ikki xil formatda berilishi mumkin:

- O‘nlik fiksirlangan nuqtali formatda. Bu ko‘rinishda son nuqta orqali

ajratilgan butun va kasr qismlar ko‘rinishida bo‘ladi. Sonning butun yoki kasr

qismi bo‘lmasligi mumkin, lekin nuqta albatta bo‘lishi kerak. Fiksirlangan nuqtali o‘zgarmaslarga misollar: 24.56; 13.0; 66.; .87;

- eksponensial shaklda haqiqiy o‘zgarmas 6 qismdan iborat bo‘ladi:

1) butun qismi (o‘nli butun son);

2) o‘nli kasr nuqta belgisi;

3) kasr qismi (o‘nlik ishorasiz o‘zgarmas);

4) eksponenta belgisi ‘e’ yoki ‘E’;

5) o‘n darajasi ko‘rsatkichi (musbat yoki manfiy ishorali o‘nli butun son);

6) qo‘shimcha belgisi (‘F’ yoki f , ‘L’ yoki ‘l’).

- Eksponensial shakldagi o‘zgarmas sonlarga misollar:

1E2; 5E+3; .25E4; 31.4E-1.

Belgi o‘zgarmaslar: Belgi o‘zgarmaslar qo‘shtirnoq (‘,’-apostroflar) ichiga olingan alohida belgilardan tashkil topadi va u char kalit so‘zi bilan aniqlanadi.Bitta belgi o‘zgarmas uchun xotirada bir bayt joy ajratiladi va unda butun son ko‘rinishidagi belgining ASCII kodi joylashadi. Quyidagilar belgi o‘zgarmaslarga misol bo‘ladi:

‘e’, ‘@’ , ‘7’, ‘z’, ‘w’, ‘+’, ‘sh’, ‘*’ , ‘a’, ‘s’.

Turlangan o‘zgarmaslar: Turlangan o‘zgarmaslar xuddi o‘zgaruvchilardek ishlatiladi va initsializatsiya qilingandan (boshlang’ich qiymat berilgandan) keyin ularning qiymatini o‘zgartirib bo‘lmaydi. Turlangan o‘zgarmaslar const kalit so‘zi bilan e’lon qilinadi, undan keyin o‘zgarmas turi va albatta initsializatsiya qismi bo‘lishi kerak.

3. Konsul orqali muloqat qilish

Misol tariqasida turlangan va literli o‘zgarmaslardan foydalangan holda radius berilganda aylana yuzasini hisoblaydigan dasturni keltiramiz.

```
#include <iosream.h>
  int main ()
  {
        const double pi=3.1415;
        const int radius=3;
```

```
    double square=0;
    square=pi*radius*radius;
    cout<<square<<‘\n’;
    return 0;
}
```

Dastur bosh funksiyasining boshlanishida ikkita – pi va radius o‘zgarmaslari e’lon qilingan. Aylana yuzasini aniqlovchi square o‘zgarmas deb e’lon qilinmagan, chunki u dastur bajarilishida o‘zgaradi. Aylana radiusini dastur ishlashida o‘zgartirish mo‘ljallanmagan, shu sababli u o‘zgarmas sifatida e’lon qilingan.

Sanab o‘tiluvchi tur: Ko‘p miqdordagi, mantiqan bog’langan o‘zgarmaslardan foydalanganda sanab o‘tiluvchi turdan foydalanish ma’qul. Sanab o‘tiluvchi o‘zgarmaslar enum kalit so‘zi bilan aniqlanadi. Mazmuni bo‘yicha bu o‘zgarmaslar oddiy butun sonlardir. Sanab o‘tiluvchi o‘zgarmaslar C++ standarti bo‘yicha butun turdagi o‘zgarmaslar hisoblanadi. Har bir o‘zgarmasga (songa) mazmunli nom beriladi va bu identifikatorni dasturning boshqa joylarida nomlash uchun ishlatilishi mumkin emas. Sanab o‘tiluvchi tur quyidagi ko‘rinishga ega:

enum <Sanab o‘tiladigan tur nomi> { <nom1> =<qiymat1>, <nom2> = <qiymat2>,… <nom3>=<qiymat3> } ;

bu yerda, enum – kalit so‘z (inglizcha enumerate – sanamoq);

<Sanab o‘tiladigan tur nomi> – o‘zgarmaslar ro‘yxatining nomi;

<nom> – butun qiymatli konstantalarning nomlari;

<qiymati> – shart bo‘lmagan initsializatsiya qiymati (ifoda).

Dastur ishlashi mobaynida qiymatlari o‘zgarishi mumkin bo‘lgan identifikatorga o‘zgaruvchilar deyiladi.

Dasturlash tillarida dastur bajarilishi paytida qandaydir berilganlarni saqlab turish uchun o‘zgaruvchilar va o‘zgarmaslardan foydalaniladi.O‘zgaruvchi-dastur obyekti bo‘lib, xotiradagi bir nechta yacheykalarni egallaydi va berilganlarni saqlash uchun xizmat qiladi. O‘zgaruvchi nomga, o‘lchamga va boshqa atributlarga – ko‘rinish sohasi, amal qilish vaqti va boshqa xususiyatlarga ega bo‘ladi. O‘zgaruvchilarni ishlatish uchun ular albatta e’lon qilinishi kerak. E’lon natijasida o‘zgaruvchi uchun xotiradan qandaydir soha zahiralanadi, soha o‘lchami esa o‘zgaruvchining aniq turiga bog’liq bo‘ladi. Shuni qayd etish zarurki, bitta turga turli apparat platformalarda turlicha joy ajratilishi mumkin.

Dasturlash tillarida kalit so‘zlar mavjud bo‘lib ulardan boshqa maqsadlarda foydalanilmaydi. Quyida **C++** tilining kalit so‘zlarini alfavit

tartibida keltiramiz.

C++ tilida: asm, auto, break, case, catch, char, class, const, continue, default, delete, do, double, else, enum, explicit, extern, float, for, friend, goto, if, inline, int, long, mutable, new, operator, private, protected, public,register, return, short, signed, sizeof, static, struct, swith, template, this, throw, try, typedef, typename, union, unsigned, virtual, void, volatile,while.

4. Amallar jadvali

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

Misol.

```
#include <iostream>;
using namespace std;
int main ()
{
   double a,b,c;
   cout <<"a="; cin>>a;
   cout <<"b="; cin>>b;
   c=a+b;
   cout<<c;
   return 0;
}
```

Nazorat savollari

1. C++ tilidagi programma tuzilishi va uning kompilyatsiyasi
2. C++ tili alfaviti va leksemalar
3. Identifikatorlar va kalit so‘zlar
4. Direktivalar tushunchasi

5. Main () funksiyasi
6. Iostream.h sarlavha fayli
7. Butun o‘zgarmaslar
8. Haqiqiy o‘zgarmaslar
9. Turlangan o‘zgarmaslar
10. Sanab o‘tiluvchi tur

1.2. C++ tilida berilganlar va ularning turlari

1. **C++ ning asosiy skalyar turlari.** Ma’lumotlar turlari kompyuter xotirasida kuplab kiymatlar, kuplab operatsiyalar va ama’ulomotlarni takdim kilish usulini belgilaydi. Ma’lumotlar turi konsepsiyasi yukori darajadagi dasturlash tilininig muxim tayanch tushunchasi xisoblanadi. Kat’iy turlash (tasniflash) tillarida, masalan bular katoriga Paskal tilini kiritish mumkin, bu tilda dasturlash kodining ishonchliligi sezilarli darajada oshadi, chunki mos kelmaslik xatolarining aksariyati kompilyatsiya boskichida aniklash imkoni mavjud.

S++ tilining sodda darajadagi imkoniyatlari bu tilni kat’iy turlash (tasniflash) tili deb xisoblashga yul kuymaydi, birok ma’lumotlar turi bu til uchun tayanch tushunchalaradan biri xisoblanadi.

Skalyar turlar

C++ning asosiy skalyar turlari:
char – simvollarga oid,
int – butun (yaxlit), **float** – moddiy,
double – ikki barobar aniklikka ega
bool – mantikiy.

Shuningdek ushbu tayanch turlarning modifikatsiyalari kullanilishi mumkin. Xosila (modifikatsiyalangan) turlar tayanch tur kiymati diapazonini uzgartiradigan 4ta tasniflagichi yordamida shakllantiriladi.

Uzunlik tasniflagichlari:

short – kalta, **long** – uzun. Belgi tasniflagichi:
signed –belgili (ijobiy va salbiy kiymatlar),
unsigned–belgisiz (fakat ijobiy kiymatlar).

Bundan tashkari **C++void**turi bilan belgilanadi –bu turning kuplab kiymatlari bush.

Dastlab xolatlarda **short** va**signed**tasniflagichlari kullaniladi, ya’ni uzunlik tasniflagichining bulmaganligi «kiska» degan ma’noni anglatadi, «belgi» tasniflagichining bulmaganligi – «belgili» ma’nosini anglatadi.

C++da **char** turining uziga xosligi talkin kilinishining ikkilamchiligida. Ushbu turdagi kiymatlar tegishli operatsiyalarni amalga oshirish mumkin bulgan

butun sonlar sifatida olib karalishlari mumkin, yoki simvollarning baytli kodi sifatida olib karalishi mumkin. **Char** (simvolli konstantalar/uzgarmaydigan kiymatlar) turidagi kiymatlar kuyidagi aporstoflarni uz ichichga kiritidi: **‘g’, ‘a’**; **char**turiga ayrim ikkisimvolli kiymatlar (maxsus kiymatlar) kiradi, masalan, **‘\n’-** keyingi katorga utish.

Muloxaza: **C++da katorli konstantalar ikkilamchi kushtirnok ichiga olinadi, masalan** “kator”. Shuning uchun ‘a’ – bu simvolli kiymat (literal), a “a” – katorli kiymat, bir simvoldan tashkil topgan kator.

Int turining ulchamlari standart tomonidan belgilanmagan va protsessor turkumiga va kompidyator xususiyatlariga boglik.

Short tasniflagichi protsessorning tugrkumiga bogliq bo‘lmagan ravishda butun tur uchun xotira kattaligini 2 bayt deb belgilaydi, **long** tasniflagichi – 4 bayt.

Ma’lumotlar turiga xos xotira xajmini sizeof funksiyasi yordamida aniklash mumkin:

sizeof(<tur nomi>) – kursatilgan tur kiymatining baytdagi ulchami, **sizeof<**uzgaruvchan nomi> – kursailgan uzgaruvchanga xos turning baytdagi ulchami.

Ma’lumotlar turlari uchun xotira xajmining anik bir kiymatini belgilamgan xolda, standart, ular urtasida kuyidagi nisbatlarni belgilaydi:

sizeof(**char**)<=**sizeof**(**short**)<=**sizeof**(**int**)<=**sizeof**(**long**), **sizeof**(**float**) <= **sizeof**(**double**)<= **sizeof**(**long double**).

16-turkumli protsessor uchun skalyar turlarning xarakteristikalari

Tur	**Kiymatlar diapazoni**	**Baytdagi ulchami**
bool	false (0), true (1)	1
char	-128 .. 127 (256 simvollar)	1
unsigned char	0 .. 255	1
signed char	-128 .. 127	1
int	-32768 .. 32767	2
unsignedint	0 .. 65535	2
signed int	-32768 .. 32767	2
short int (short)	-32768 .. 32767	2
unsigned shortint	0 .. 65535	2
signed short int	-32768 .. 32767	2
long int (long)	-2147483648 .. 2147483647	4
signed long int	-2147483648 .. 2147483647	4
unsigned longint	0 .. 4294967295	4
float	3.4e-38 .. 3.4e+38	4

short float	3.4e-38 .. 3.4e+38	4
long float	1.7e-308 .. 1.7e+308	8
double	1.7e-308 .. 1.7e+308	8
short double	1.7e-308 .. 1.7e+308	8
long double	3.4e-4932 .. 3.4e+4932	10

Muloxaza: *C++* dagi ayrim turdagi dasturlash tizimlarida **char** turi uchun 0…255 gacha bulgan kiymatlar diapazoni belgilanishi mumkin, bunday xolatda **char** turi **signed char**bilanemas **, unsigned char** bilan mos keladi.

Ushbu jadvalda turli tasniflarga bulinadigan ma'lumotlar turi mavjud, birok mazmunan olib karaganda, masalan **int**va **signed int** dan fark kilmaydilar. Ushbu turning nisbatan soda tasniflarini terib chikish bilan chegaralanib kolish imkonini beradi.

C++ tilida ham o'zgaruvchilarning turlari bir necha guruhlarga ajraladi. Ularni quyida qarab chiqamiz.

Butun son turlari. Butun son qiymatlarni qabul qiladigan o'zgaruvchilar **int** (butun), **short** (qisqa) va **long** (uzun) kalit so'zlar bilan aniqlanadi. O'zgaruvchi qiymatlari ishorali bo'lishi yoki unsigned kalit so'zi bilan ishorasiz son sifatida qaralishi mumkin.

Belgi turi. Belgi turidagi o'zgaruvchilar char kalit so'zi bilan beriladi vaular o'zida belgining ASCII kodini saqlaydi. Belgi turidagi qiymatlar nisbatan murakkab bo'lgan tuzilmalar – satrlar, belgilar massivlari va hokazolarni hosil qilishda ishlatiladi.

Haqiqiy son turi. Haqiqiy sonlar float kalit so'zi bilan e'lon qilinadi. Bu turdagi o'zgaruvchi uchun xotiradan 4 bayt joy ajratiladi va <ishora><tartib><mantissa> qolipida sonni saqlaydi. Agar kasrli son juda katta (kichik) qiymatlarni qabul qiladigan bo'lsa, u xotirada 8 yoki 10 baytli ikkilangan aniqlik ko'rinishida saqlanadi va mos double va long double kalit so'zlari bilan e'lon qilinadi. Oxirgi holat **32-razryadli** platformalar uchun o'rinli.

Mantiqiy tur. Bu turdagi o'zgaruvchi bool kalit so'zi bilan e'lon qilinib, xotiradan **1 bayt** joy egallaydi va **0** (**false**-yolg'on) yoki (**true-rost**) qiymat qabul qiladi. Mantiqiy tur o'zgaruvchilar qiymatlar o'rtasidagi munosabatlarni ifodalaydigan mulohazalarni **rost** (true) yoki **yolg'on** (false) ekanligi tavsifida qo'llaniladi va ular qabul qiladigan qiymatlar matematik mantiq qonuniyatlariga asoslanadi.

Mantiqiy mulohazalar ustida uchta amal aniqlangan:

1) **inkor** – A mulohazani inkori deganda A rost bo'lganda yolg'on yoki yolg'on bo'lganda rost qiymat qabul qiluvchi mulohazaga aytiladi. C++

tilida inkor – ‘!’ belgisi bilan beriladi. Masalan, A mulohaza inkori «!A» ko‘rinishida yoziladi;

2) **konyunksiya**- ikkita A va B mulohazalar konyunksiyasi yoki mantiqiy ko‘paytmasi «A && B» ko‘rinishga ega. Bu mulohaza faqat A va B mulohazalar rost bo‘lgandagina rost bo‘ladi, aks holda yolg’on bo‘ladi (odatda «&&» amali «va» deb o‘qiladi). Masalan «bugun oyning 5- kuni va bugun chorshanba» mulohazasi oyning 5- kuni chorshanba bo‘lgan kunlar uchungina rost bo‘ladi;

3) **dizyunksiya** – ikkita A va B mulohazalar dizyunksiyasi yoki mantiqiy yig’indisi «A || B» ko‘rinishda yoziladi. Bu mulohaza rost bo‘lishi uchun A yoki Bmulohazalardan biri rost bo‘lishi yetarli. Odatda «||» amali «yoki» deb o‘qiladi. Yuqorida keltirilgan fikrlar asosida mantiqiy amallar uchun rostlik jadvali aniqlangan (1.1-jadval).

1.1-jadval. Mantiqiy amallar uchun rostlik jadvali

Mulohazalar		**Mulohazalar ustida amallar**		
A	**B**	**NotA**	**A and B**	**A or B**
		!A	**A&&B**	**A||B**
False	False	True	False	False
False	True	True	False	True
True	False	False	False	True
True	True	False	True	True

Mantiqiy tur qiymatlari ustida mantiqiy ko‘paytirish, qo‘shish va inkor mallarini qo‘llash orqali murakkab mantiqiy ifodalarni qurish mumkin. Misol uchun, ***«x – musbat va uning qiymati [1..3] sonlar oralig’iga tegishli emas»*** mulohazasini mantiqiy ifoda ko‘rinishi quyidagicha bo‘ladi:

(x>0)&&(y<1|| y>3)

Void turi. C++ tilida **void** turi aniqlangan bo‘lib bu turdagi dastur obyekti hech qanday qiymatga ega bo‘lmaydi va bu turdan qurilmaning til sintsksisiga mos kelishini ta’minlash uchun ishlatiladi. Masalan, **C++** tili sintsksisi funksiya qiymat qaytarishini talab qiladi. Agar funksiya qiymat qaytarmaydigan bo‘lsa, u void kalit so‘zi bilan e’lon qilinadi.

Misollar.

```
int a=0 A=1; float abc=17.5;
double Ildiz;
bool ok=true;
char LETTER=‘z’;
Void mening_funksiyam(); /*funksiya qaytaradigan qiymat inobatga olinmaydi*/
```

Turni boshqa turga keltirish: C++ tilida bir turni boshqa turga keltirishning oshkor va oshkormas yo‘llari mavjud. Umuman olganda, turni boshqa turga oshkormas keltirish ifodada har xil turdagi o‘zgaruvchilar qatnashgan hollarda amal qiladi (aralash turlar arifmetikasi). Ayrim hollarda,xususan tayanch turlar bilan bog’liq turga keltirish amallarida xatoliklar yuzaga kelishi mumkin. Masalan, hisoblash natijasining xotiradan vaqtincha egallagan joyi uzunligi, uni o‘zlashtiradigan o‘zgaruvchi uchun ajratilgan joy uzunligidan katta bo‘lsa, qiymatga ega razryadlarni yo‘qotish holati yuz beradi.

Oshkor ravishda turga keltirishda, o‘zgaruvchi oldiga qavs ichida boshqa tur nomi yoziladi:

```
#include <iostream.h>
int main()
{
int integer_1=54;
int integer_2;
float floating=15.854;
integer_1=(int) floating; // oshkor keltirish;
integer_2=(int) floating // oshkormas keltirish;
cout<<‘‘yangi integer (oshkor): ‘‘<<integer_1<<‘‘\n’’;
cout<<‘‘yangi integer (oshkormas): ‘‘<<integer_2<<‘‘\n’’;
return 0;
 }
```

Dastur natijasi quyidagi ko‘rinishda bo‘ladi:

*Yangi integer (oshkor):**15***

*Yangi integer (oshkormas):**15***

Masala. Berilgan belgining ASCII kodi chop etilsin. Masala belgi turidagi qiymatni oshkor ravishda butun son turiga keltirib chop qilish orqali yechiladi.

Dastur matni:

```
#include <iostream.h>
int main()
{
Unsigned char A;
Cout<<‘‘belgini kiriting:’’;
Cin>>A;
Cout<<‘\’’<<A<<‘‘-belgi ASCII kodi=‘‘(int)A<<‘\n’;
Return 0;
}
```

Dasturning belgini kiriting so‘roviga

A <enter> amali bajarilsa, ekranga ‘A’-belgi ASCII kodi=65 satri chop etiladi.

Nazorat savollari

1. C++ ning asosiy skalyar turlari
2. Uzunlik tasniflagichlari
3. Sizeof funksiyasi
4. Butun va haqiqiy sonli turlar
5. Belgili va mantiqiy turlar
6. Mantiqiy mulohazalar ustida amallar
7. Turni boshqa turga keltirish

1.3. C++ tilida ifodalar va operatorlar

1. Arifmetik amallar

Berilganlarni qayta ishlash uchun dasturlash tillarida amallarning juda keng majmuasi aniqlangan.

Amal - bu qandaydir harakat bo‘lib, u bitta (**unar**) yoki ikkita (binar) operandlar ustida bajariladi, hisob natijasi uning qaytaruvchiqiymati hisoblanadi. Tayanch arifmetik amallarga qo‘shish **(+)**, ayirish **(-)**, ko‘paytirish **(*)**, bo‘lish **(/)** va bo‘lishdagi qoldiqni olish **(%)** amallarini keltirish mumkin.

Amallar qaytaradigan qiymatlarni o‘zlashtirish uchun **C++** tilida “=” va uning turli modifikatsiyalari ishlatilib, quyidagilar hisoblanadi: qo‘shish, qiymat berish bilan (+=); ayirish, qiymat berish bilan (-=); ko‘paytirish, qiymat berish bilan (***=**); bo‘lish, qiymat berish bilan (**/=**); bo‘lish qoldig’ini olish, qiymat berish bilan (**%=**) va boshqalar. Bu holatlarning umumiy ko‘rinishi:

<o‘zgaruvchi><amal>=<ifoda>;

Quyidagi dastur matnida ayrim amallarga misollar keltirilgan.

```
#include <iostream.h>

int main()

{

    int a=0 , b=4, c=90; char z=‘\t’;
    a=b; cout<<a<<z;                    //a=4
    a=b+c+c+b; cout<<a<<z;              //a=4+90+90+4=188
    a=b-2; cout<<a<<z;                  //a=2
    a=b*3 cout<<a<<z;                   //a=4*3=12
    a=c/(b+6); cout<<a<<z;              //a=90/(4+6)=9
    cout<<a%2<<z;                       //9%2=1
```

```
    a+=b; cout<<a<<z;          //a=a+b=9+4=13
    a*=c-50; cout<<a<<z;       //a=a*(c-50)=13*(90-50)=520
    a-=38; cout<<a<<z;         //a=a-38=520-38=482
    a%=8; cout<<a<<z;          //a=a%8=482%8=2
    return 0;
}
```

Dastur bajarilishi natijasida ekranda quyidagi sonlar satri paydo bo‘ladi:

4 188 2 12 9 1 482 2

2. Razryadli mantiqiy amallar

Dastur tuzish tajribasi shuni ko‘rsatadiki, odatda qo‘yilgan masalani yechishda biror holat ro‘y bergan yoki yo‘qligini ifodalash uchun **0** va **1** qiymat qabul iluvchi bayroqlardan foydalaniladi. Bu maqsadda bir yoki undan ortiq baytli o‘zgaruvchilardan foydalanish mumkin. Masalan, mantiqiy turdagi o‘zgaruvchini shu maqsadda ishlatsa bo‘ladi. Boshqa tomondan, bayroq sifatida baytning razryadlaridan foydalanish ham mumkin. Chunki razryadlar faqat ikkita qiymatni - **0** va **1** sonlarini qabul qiladi. Bir baytda **8 razryad** bo‘lgani uchun unda **8** ta bayroqni kodlash imkoniyati mavjud.

Quyidagi jadvalda C++ tilida bayt razryadlari ustida mantiqiy amallar majmuasi keltirilgan:

1.2- jadval. Bayt razryadlari ustida mantiqiy amallar

Amallar		**Mazmuni**
And	**&**	Mantiqiy VA (ko‘paytirish)
Or	|	Mantiqiy YOKI (qo‘shish)
Xor	^	Istisno qiluvchi YOKI
Not	~	Mantiqiy INKOR (inversiya)

C++ tilida razryadli mantiqiy amallarni qiymat berish operatori birgalikda bajarilishining quyidagi ko‘rinishlari mavjud:

&= – razryadli VA qiymat berish bilan;

| = – razryadli YOKI qiymat berish bilan;

^= – razryadli istisno qiluvchi YOKI qiymat berish bilan.

1.3-jadval. Ifodalar yozilganida qullaniladigan asosiy operatsiyalar

Arifmetik unarli	
+	unarli plyus
-	unarli minus (arifmetik inkor etish)

++	1ga oshirish (inkrement)
--	1ga kamaytirish (dekrement)
Arifmetik binarli	
*	Kupaytirish
/	Bulish
%	Modul buyicha bulish (bulishdan kolgan koldik)
+	Kushish
-	Ayrish
mantikiy unarli	
!	Inkor etish
mantikiy binarli	
&&	Mantikiy VA (kon'yunksiya)
||	Manikiy YoKI (diz'yunksiya)
nunosabatlar operatsiyalari	
<	Kichik
<=	Kichikyoki teng
==	Teng
!=	Teng emas
>=	Katta yoki teng
>	Katta

1.4-jadval. Taqqoslash amallari

Amallar		**Qo'llanilishi**		**Mazmuni (o'qilishi)**
<	<	a<b	a<b	"a kichik b"
<=	<=	a<=b	a<=b	"a kichik yoki teng b"
>	>	a>b	a>b	"a katta b"
>=	>=	a>=b	a>=b	"a katta yoki teng "
=	==	a=b	a==b	"a teng b"
< >	!=	a< >b	a!=b	"a teng emas b"

Inkrement i dekrementning unar operatsiyalari ikkita shaklad kullanilishi mumkin: prefiksli va postfiksli.

Prefiks shakli

++x - inkrement, operand kiymatini u kullanilguncha 1 ga kupaytirish;

--x - dekrement,operand kiymatini u kullanilguncha 1 ga kamaytirish; Dastlab (x) uzgaruvchani uzgartiriladi, sungra esa ushbu uzgarishni inobtaga olgan xolda tarkibiga inkrement (dekrement) kiritilgan ifoda xisoblab chikiladi.

Postfik shakli

x++ - inkrement, operand kiymatini u kullanilguncha 1 ga kupaytirish;

x-- - dekrement,operand kiymatini u kullanilguncha 1 ga kamaytirish;

Dastlab tarkibiga inkrement (dekrement) kiritilgan ifoda xisoblab chikiladi, bunda uzgaruvchan (x)ning eski kiymati kullaniladi, sungra esa uzgaruvchan uzgartiriladi.

Misol.

```
#include <iostream>;
using namespace std;
int main ()
{
   double a,b,c;
   cout <<"a="; cin>>a;
   cout <<"b="; cin>>b;
   c=a+b;
   cout<<c;
   return 0;
}
```

3. C++ da operatorlar

C++ tilida operand qiymatini birga oshirish va kamaytirishning samarali vositalari mavjud. Bular inkrement (++) va decrement(--) unar amallardir.

1) **inkrement (++)**
2) **decrement (--)**

Qiymat berish, inkrement va dekrement operatorlari

Qiymat berish amallari guruhi, xususan, qiymat berish operatorlari ifoda operatorlari hisoblanadi, C++ da qiymat berish operatori ‘=‘ ko‘rinishida bo‘ladi:

k=8;.

O‘zgaruvchilarga qiymat berish, ya’ni ularning oldingi qiymatini o‘zgartirishning boshqa usullari ham mavjud. Bu kabi ishlarni (***i++;), (--j;), (k+=i;)*** operatorlari yordamida amalga oshirish mumkin.

Quyida keltirilgan jadvalda **C++** tilida o‘zgaruvchilarga qiymat kiritish tasvirlangan.

cin>>S1>>S2;
cin>>x1>>x2>>x3>>"\n";
cin>>"\n";

Bu yerda birinchi operator Sl va S2 o‘zgaruvchilar qiymatini klaviaturadan kiritadi. Ikkinchi operator esa xl, x2, x3 o‘zgaruvchilar qiymatini klaviauradan qabul qiladi va kiritishni keyingi qatorga o‘tkazadi. Oxirgi operator esa kiritishni kutadi va kursorni navbatdagi qatorga o‘tkazadi.

4. Kiritish va chiqarish operatorlari

C++ tilida o‘zgaruvchilar qiymatini klaviaturadan kiritish uchun ***cin>><o‘zgaruvchi 1> >> <o‘zgaruvchi 2> >>… >> <o‘zgaruvchi n>*** kiritish oqimidan foydalaniladi.

Quyida keltirilgan jadvalda C++ tilida o‘zgaruvchilarga qiymat kiritish tasvirlangan.

cin>>S1>>S2;
cin>>x1>>x2>>x3>>"\n";
cin>>"\n";

Chop etish operatori:

C++ tilida oddiy ma’lumotlar, o‘zgaruvchilar va ifodalar qiymatini chop etish uchun ***cout<< <o‘zgaruvchi 1> << <o‘zgaruvchi 2> <<…<< <o‘zgaruvchi n>*** kiritish oqimidan foydalaniladi.

cout<<Summa<<"\n";
cout<<"Natija yo‘q";
cout<<"Tenglama yechimi x1="<<x1<<"x2=" <<x2;

1-masala.

$$y = \frac{2{,}5Sinx + 0{,}75Tg^2x^3}{0{,}65\sqrt[3]{x}\,e^{Sinx} + Cos^2x^3}$$

Dastur kodi	Natija

#include <iostream>
#include <cmath>
using namespace std;
int main()
{
float y,x;
cout<<"x="; cin>>x;
	x=0
	y=0

```
y=(2.5*sin(x)+0.75*pow(tan(pow(x,3)),2))/(0.65*cbrt(x)*exp(sin(x))+
pow(cos(pow(x,3)),2));
   cout<<"y="<<y;
   return 0;
}
```

2-masala. Berilgan belgining ASCII kodi chop etilsin. Masala belgi turidagi qiymatni oshkor ravishda butun son turiga keltirib chop qilish orqali yechiladi.

Dastur matni:

```
#include <iostream.h>
int main()
{
Unsigned char A;
Cout<<''belgini kiriting:'';
Cin>>A;
Cout<<'\''<<A<<''-belgi ASCII kodi=''(int)A<<'\n';
Return 0;
}
```

Dasturning belgini kiriting so‘roviga

A <enter> amali bajarilsa, ekranga ‘A’-belgi ASCII kodi=65 satri chop etiladi.

Nazorat savollari

1. C++ tilining tarkibi nimalardan iborat?
2. C++ tilining asosiy kalit so‘zlari
3. Konstantalar nima uchun xizmat qiladi?
4. C++ tilida o‘zgaruvchilarning nomlanishi
5. C++ da arifmetik amallar
6. Razryadli mantiqiy amallar
7. Inkrement va dekrementning unar operatsiyalari
8. C++ da operatorlar
9. Konsoldan kiritish va chiqarish operatorlari
10. Identifikatorlar nima?

II-BOB. C++ TILIDA JARAYONLARNI DASTURLASH OPERATORLARI

2.1. C++ tilida chiziqli jarayonlarni dasturlash

Masalani hal etish uchun tuzilgan algoritm tarkibidagi buyruqlar ketma ketligi uzluksiz bo‘lishi mumkin yoki qandaydir holatlarda shartlar asosida uzluksizlik tarqatilishi mumkin. Chiziqli algoritmlarda esa buyruqlar ketma-ketligi doim uzluksiz bo‘ladi.

Algoritmni ijro etishda uning buyruqlari qanday tartibda berilgan bo‘lsa, o‘sha tartibda bajarilsa, bunday algoritmlarni bajarish - buyruqlar tabiiy tartibiga bo‘ysunadi — deyiladi. Agar algoritmlarning buyruqlarini bajarish tabiiy tartibga bo‘ysunsa, bunday algoritmlar ***chiziqli algoritmlar*** deyiladi [1].

C++ dasturlash tilida tuzilgan dasturlar albatta uchta jarayonga asoslanib tuziladi. Dasturlash tili operatorlari yechilayotgan masala algoritmini amalga oshirish uchun ishlatiladi. Operatorlar chiziqli va boshqaruv operatorlariga bo‘linadi. Aksariyat holatlarda operatorlar nuqtali vergul (;) belgisi bilan tugallanadi va u kompilyator tomonidan alohida operator deb qabul qilinadi.

Kompilyator dasturni ishga tushirish vaqtida dasturni kodini mashina tiliga tarjima qiladi. Dastur tuzish vaqtida buyruqlar ketma-ketligi uzluksiz bajarilib boshqa shartlar talab etilmasa, dastur chiziqli hisoblanadi.

Tarif: Chiziqli algoritmlarga asoslanib ixtiyoriy dasturlash tilida tuzilgan dasturlar **chiziqli dasturlar** deyiladi.

Chiziqli dasturlar tarkibiy qismi bo‘lgan operator va buyruqlarda hech qanday shart yoki takrorlanish bajarilmaydi. Chiziqli dasturlar tarkibidagi boyruqlar, albatta, bir marta bajariladi.

C++ dasturi ko‘rsatmalar to‘plamidan iborat. Har bir ko‘rsatma muayyan harakatni amalga oshiradi. C++ ko‘rsatma nuqta-vergul (**;**) bilan tugaydi. Bu belgi kompilyatorga buyruqni bajarishni bildiradi.Masalan:

```
std::cout << "Hello World!";
```

Bu satr konsolga “Hello World!” qatorini chop etadi, bu ko‘rsatmadir va shuning uchun nuqtali vergul bilan tugaydi. Ko‘rsatmalar to‘plami kod blokini ko‘rsatishi mumkin. Kod bloki figurali **{,}** qavslar ichiga olingan va ko‘rsatmalar ochilish va yopish figurali qavslar orasiga joylashtirilgan:

```
{
   std::cout << "Hello World!";
   std::cout << "Bye World!";
}
```

Main funksiyasi

C++ har bir dasturida kamida bitta funktsiya, **main()** funksiyasi bo‘lishi kerak. Aynan shu funksiya bilan dasturning bajarilishi boshlanadi. Uning asosiy nomi-**main** aniq va barcha C++ dasturlari uchun har doim bir xil. Funksiya ham kod blokidir, shuning uchun uning tanasi figurali qavslar bilan hoshiyalanadi, ular orasida ko‘rsatmalar to‘plami aniqlanadi. Xususan, birinchi dasturni yaratishda quyidagi asosiy funktsiyadan foydalanilgan:

```
#include <iostream>                 // iostream sarlavha faylini o‘z ichiga oladi

int main()                          // main funktsiyasini aniqlaymiz
{                                   // funktsiyani boshlash
    std::cout << "Salom Dunyo!";  // konsolga satrni chop etish
    return 0;                       // funksiyadan chiqish
}                                   // funktsiyaning oxiri
```

Konsolga chiqishi

Konsolga chiqarish uchun **<<** operatori ishlatiladi. Bu operator ikkita operandni oladi. Chap operand **ostream** tipidagi obyektni ifodalaydi, bu holda **cout**. O‘ng operand esa konsolga chiqariladigan qiymatdir. **<<** operatori chap operand, **cout**ni qaytarganligi sababli, operatorlar zanjiri yordamida konsolga bir nechta qiymatlarni uzatishimiz mumkin. Masalan, oddiy dasturini konsolga chiqarish ko‘ramiz:

```
#include <iostream>

int main()

{

   int age {33};

   double weight {81.23};

   std::cout << "Name: " << "Tom" << "\n";

   std::cout << "Age: " << age << std::endl;

   std::cout << "Weight: " << weight << std::endl;

}
```

Dasturning konsolga chiqishi:

Name: Tom
Age: 33

Weight: 81.23

C++ tilida chiziqli dasturlarga oid misollar

1) . *Teng tomonli uchburchakning a tomoni berilgan. Shu uchburchakni yuzini hisoblovchi dastur tuzing.*

Dastur kodi:	Natija:

#include<iostream>
#include<math.h>
using namespace std;
int main()
{
int a, s ;
cout<<"a=";cin>>a;
s=(sqrt(3)*pow(a,2))/4;
cout<<"uchburchak
yuzi=="<<s;
return 0;
}
	a=3
	uchburchak yuzi==3

2). a, b va c sonlari berilgan, shu sonlarni yig'indisini topuvchi dastur tuzing.

Dastur kodi:	Natija:

#include<iostream>
using namespace std;
int main()
{
int a, b, c, summ;
cout<<"a=";cin>>a;
cout<<"b=";cin>>b;
cout<<"c=";cin>>c;
summ=a+b+c;
cout<<"summ=="<<summ;
return 0;
}
	a=4
	b=5
	c=6
	summ==15

3). a va b sonlari berilgan ularni o'rta arifmetikigini hisoblovchi dastur tuzing.

Dastur kodi:	Natija:
#include<iostream> using namespace std; int main() { int a, b, summ;	a=4 b=5 summ==15

cout<<"a=";cin>>a;
cout<<"b=";cin>>b;
summ=(a+b)/2;
cout<<"summ=="<<summ;
return 0;
}	

4). Uzunlik l santimetrda berilgan. Undagi to‘liq metrlar sonini aniqlovchi dastur tuzing.

Dastur kodi:	Natija:

#include<iostream>
using namespace std;
int main()
{
int l, m;
cout<<"l=";cin>>l;
m=l/100;
cout<<"uzunligi=="<<m<<"
m";
return 0;
}
	l=450
	uzunligi==4 m

5). Uch xonali son berilgan. Uning yuzlar xonasidagi raqamni aniqlovchi dastur tuzing.

Dastur kodi:	Natija:

#include<iostream>
using namespace std;
int main()
{
int l, m;
cout<<"l=";cin>>l;
m=l/100;
cout<<"yuzlar xonasidagi
raqam=="<<m;
return 0;
}
	l=789
	yuzlar xonasidagi raqam==7

Ushbu dasturlar chiziqli dastur hisoblanadi. Chunki unung buyruqlari berilgan tartibda birin-ketin bajariladi. C++ dasturlash tilida chiziqli dasturlar

tuzilganda uning tarkibida matematik funksiyalar ishtirok etsa, albatta, matematik funksiyalar paketini chaqirish kerak.

Chiziqli dasturlashda matematik funksiyalardan foydalanish mumkin. Matemetik funksiyalardan foydalanish uchun **cmath** jutibxonasini e'lon qilish lozim. Quyidagi 2.1-jadvalda matematik funksiyalarni C++ tilida yozilishi keltirilgan.

2.1-jadval. Matemetik funksiyalar va ularni C++ tilida yozilishi.

Funksiya	Tavsifi	Misol
abs(a) fabs(a)	**a** ning moduli **a** haqiyqiy son moduli	abs(-3)= 3 abs(5)= 5
sqrt(a)	**a** ning kvadrat ildizi	sqrt(9)=3.0
pow(a, b)	**a** ni **b** darajaga ko'tarish	pow(2,3)=8
ceil(a)	**a** ni o'zidan kichik bo'lmagan eng kichik butun songa yaxlitlash	ceil(2.3)=3.0 ceil(-2.3)=-2.0
floor(a)	**a** ni o'zidan katta bo'lmagan eng kichik butun songa yaxlitlash	floor(12.4)=12 floor(-2.9)=-3
fmod(a, b)	a/b ni hisoblashdagi qoldiqni olish	fmod(4.4, 7.5) = 4.4 fmod(7.5, 4.4) = 3.1
exp(a)	$\mathbf{e^a}$ ni hisoblash	exp(0)=1
sin(a)	***sina,*** **a** radiyanda beriladi.	
cos(a)	***cosa,*** **a** radiyanda beriladi.	
tan(a)	***tga,*** **a** radiyanda beriladi.	
log(a)	**a** natular logarifmi	log(1.0)=0.0
log10(a)	**a** ning o'nlik logarifmi	Log10(10)=1
asin(a)	**arcsin*a***, bunda **-1.0 < a < 1.0.** Natija radiyanda xosil bo'ladi	asin(1)=1.5708
acos(a)	**arccos*a***, bunda **-1.0 < a < 1.0.** Natija radiyanda xosil bo'ladi	
atan(a)	**arctg*a***, bunda **-1.0 < a < 1.0.** Natija radiyanda xosil bo'ladi	

Bo'linmaning haqiqiy qismi kerak bo'lga, agar o'zgaruvchilar butun son bo'lsa bo'lish amaliga e'tibor qaratish lozim.

1. Misol. Asosining uzunligi ***a*** va balandligi ***h*** ga teng bo'lgan uchburchakning yuzasini hisoblovchi dastur tuzing.

Yechish.

Kiruvchi ma’lumot a va h butun sonlari. Uchburchak yuzasi formulasi: $S = \frac{ah}{2}$

```
#include <iostream>
using namespace std;
int main()
{
	int a, h;
	cin>>a>>h;
	double s = a * h / 2;
	cout<<s;
}
```

Dasturda hatolik mavjud. Bu hatolik shundan iboratki, butun sonlarni bo‘lganda bo‘linmaning butun qiymati hisoblanadi. Bo‘linmaning haqiqiy qiymatini hisoblash uchun bo‘linuvchilardan birining qiymati haqiqiy bo‘lishi kerak. Yuqoridagi masalada buni

double s = a * h / 2.0;

yoki

double s = 1.0 * a * h / 2;

ko‘rinishida yozish orqali to‘girlash kiritishimiz mumkin.

2. Misol. Murakkab ifodani yechish.

$$AF = 2^{-x} \cdot \sqrt{x + \sqrt[4]{|y| + 2}} \cdot \sqrt[3]{e^{x-1} / \sin(z + 2) + 2};$$

3.

Bunda kiruvchi ma’lumotlar ***x, y, z*** haqiqiy sonlari. Chiquvchi ma’lumot ***AF***.

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

using namespace std;

int main()
{
	double x, y, z;
	cin>>x>>y>>z;
double AF = pow(2, -x)*sqrt(x+sqrt(sqrt(fabs(y)+2))) * pow(exp(x-
1) / sin(z+2) + 2, 1. / 3);
```

```
    printf("%.2f", AF);
}
```

printf () funksiyasi xaqiqiy sonni nuqtadan so‘ng biror xona aniqlikda chiqarish uchun hizmat qiladi. Agar sonning qiymati **3.5689** ga teng bo‘lsa yaxlitlab chiqarilganda **3.57** soni chiqariladi.

Nazorat savollari

1. Butun sonlar toifalarini sanab bering. Ular nimasi bilan farq qiladi?
2. Haqiqiy sonlar toifalarini sanab bering. Ular nimasi bilan farq qiladi?
3. Matematik funksiyalarni sanab bering.
4. Mantiqiy toifalar qanday e’lon qilinadi?
5. Mantiqiy amallarni tushuntirib bering.
6. Munosabat amallarini tushuntiring.
7. Mantiqiy amallar jadvalini tuzib bering.

2.2. C++ tilida tarmoqlanuvchi jarayonlarni dasturlash

1. Shart operatorlari

Yuqorida mavzularda keltirilgan dasturlarda amallar yozilish tartibida ketma-ket va faqat bir marta bajariladigan holatlar, ya’ni chiziqli algoritmlar keltirilgan. Amalda esa kamdan-kam masalalar shu tariqa yechilishi mumkin. Aksariyat masalalar yuzaga keladigan turli holatlarga bog’liq ravishda mos qaror qabul qilishni (yechimni) talab etadi. C++ tilida dasturning alohida bo‘laklarining bajarilish tartibini boshqarishga imkon beruvchi qurilmalarning yetarlicha katta ajmuasiga ega.

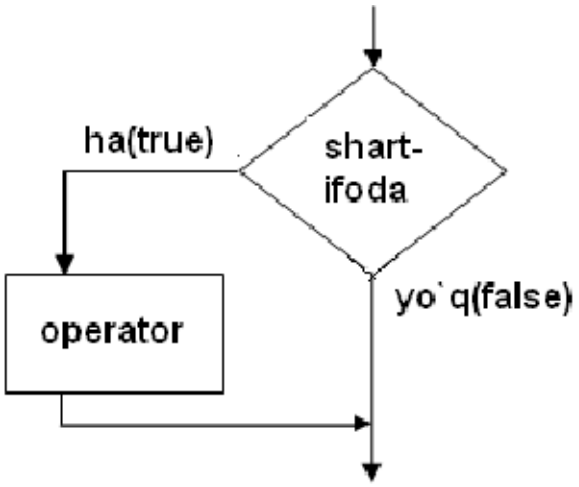
Masalan, dastur bajarilishining birorta qadamida qandaydir shartni tekshirish natijasiga ko‘ra boshqaruvni dasturning u yoki bu bo‘lagiga uzatish mumkin (tarmoqlanuvchi algoritm). Tarmoqlanishni amalga oshirish uchun shartli operatordan foydalaniladi.

If operatori: If operatori qandaydir shartni rostlikka tekshirshi natijasiga ko‘ra dasturda tarmoqlanishni amalga oshiradi:

IF (<SHART>) <OPERATOR>;

Bu yerda <shart> har qanday ifoda bo‘lishi mumkin. Odatda u taqqoslash amali bo‘ladi. Agar shart ***0*** qiymatidan farqli yoki ***ros***t (***true***) bo‘lsa, **<*operator*>** bajariladi, aks holda, ya’ni shart ***0*** yoki ***yolg’on*** (false) bo‘lsa, hech qanday amal bajarilmaydi va boshqaruv ***if operatoridan*** keyingi operatorga o‘tadi ***(i (agar u***

mavjud bo‘lsa). Ushbu holat 2.1 –rasmda ko‘rsatilgan.



2.1-rasm. If () shart operatorining blok sxemasi

C++ tillarining qurilmalari operatorlarni blok ko‘rinishida tashkil qilishga imkon beradi. Blok C++ tilida ‘**{**‘ va ‘**}**’ belgi oralig’iga olingan operatorlar ketma-ketligi ko‘rinishida bo‘ladi. Blok kompilyator tomonidan yaxlit bir operator deb qabul qilinadi. C++ tilida blok ichida e’lon operatorlari ham bo‘lishi mumkin va ularda e’lon qilingan o‘zgaruvchilar faqat shu blok ichida ko‘rinadi (amal qiladi), blokdan tashqarida ko‘rinmaydi. Blokdan keyin **‘;’** belgisi qo‘yilmasligi mumkin, lekin blok ichidagi har bir ifoda **‘;’** belgisi bilan yakunlanishi shart.

Quyida keltirilgan dasturda if operatoridan foydalanish ko‘rsatilgan.

```
#include <iostream.h>
 int main()
{
     int b;
     cin>>b;
     if (b>0)
           {
            cout<<‘‘b- musbat son’’;
           }
      if (b<0)
            cout<<‘‘b – manfiy son’’;
      return 0;
 }
```

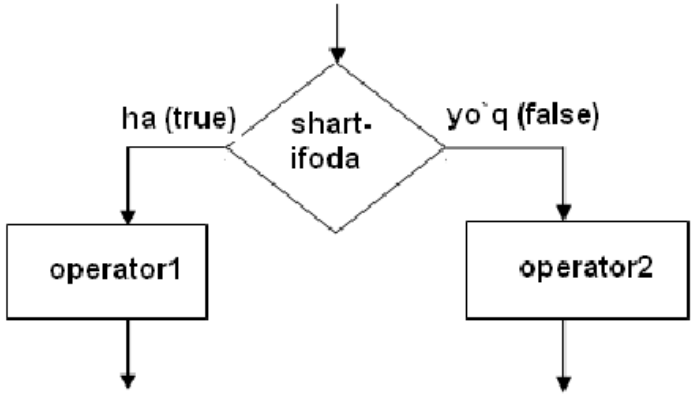
Dastur bajarilishi jarayonida butun turdagi ***b*** o‘zgaruvchi e’lon qilingan va uning qiymati klaviaturadan o‘qiladi. Keyin ***b*** qiymatini **0** sonidan kattaligi

tekshiriladi, agar shart bajarilsa ***(true)*** , u holda ekranga ***“b – musbat son”*** xabari chiqadi. Agar shart bajarilmasa, bu operatorlar cheklab o‘tiladi. Navbatdagi shart operatori ***b*** o‘zgaruvchi qiymatini anfiylikka tekshiradi, agar shart bajarilsa, ekranga ***“b – manfiy son”*** xabari chiqadi.

If – else operatori: Shart operatorining **if – else** ko‘rinishi quyidagicha**:**

if (<shart-ifoda>) <operator1>; **else** <operator2>;

Bu yerda **<*shart-ifoda*> 0** qiymatidan farqli yoki ***true*** bo‘lsa **<*operator1*>**, aks holda **<*operator2*>** bajariladi. ***If-else*** shart operator mazmuniga ko‘ra algoritmning tarmoqlanuvchi blokini ifodalaydi: **<*shart-ifoda*>** - shart bloki (romb) va **<*operator1*>** blokning ***«ha»*** tarmog’iga, **<*operator2*>** esa blokning ***«yo‘q»*** tarmog’iga mos keluvchi amallar bloklari deb qarash mumkin.



2.2-rasm. If (), else shart operatorining blok sxemasi

Misol tariqasida diskriminantni hisoblash usuli yordamida ax2+bx+c=0 ko‘rinishidagi kvadrat tenglama ildizlarini topish masalasini ko‘raylik:

```
#include <iostream.h>
#include <math.h>
int main()
{
        float a,b,c;
        float D,x1,x2;
        cout<<‘‘ax^2+bx+c=0; tenglama ildizini topish.’’;
        cout<<‘‘\n a- koiffitsiyentni 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 ildizga ega emas!’’;
              return 0;
        }
        if (D==0)
        {
              cout<<‘‘tenglama yagona ildizga ega:’’;
              x1=-b/(2*a);
              cout<<‘‘\nx=‘‘<<x1;
              return o;
        }
        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;
}
      return 0;
}
```

Dastur bajarilganda, birinchi navbatda tenglama koeffitsientlari- ***a, b, c*** o‘zgaruvchilar qiymatlari kiritiladi, keyin diskriminant – ***D*** o‘zgaruvchining qiymati hisoblanadi. Keyin ***D*** o‘zgaruvchining manfiy ekanligi tekshiriladi. Agar shart o‘rinli bo‘lsa, yaxlit operator bajariladi va ekranga ***“Tenglama haqiqiy ildizlarga ega emas”*** xabari chiqadi va dastur o‘z ishini tugatadi ***(“return 0;” operatorini bajarish orqali)***. Diskriminant noldan kichik bo‘lmasa, navbatdagi shart operatori uni nolga tengligini tekshiradi. Agar shart o‘rinli bo‘lsa, keyingi qatorlardagi operatorlar bloki bajariladi – ekranga ***“Tenglama yagona ildizga ega:”*** xabari, hamda x1 o‘zgaruvchi qiymati chop qilinadi va dastur shu yerda o‘z ishini tugatadi, aks holda, ya’ni ***D*** qiymatni noldan katta holati uchun else kalit so‘zidan keyingi operatorlar bloki bajariladi va ekranga ***“Tenglama ikkita ildizga ega:“*** xabari, hamda ***x1*** va ***x2*** o‘zgaruvchilar qiymatlari chop etiladi. Shu bilan shart operatoridan chiqiladi va asosiy funksiyaning return ko‘rsatmasini bajarish orqali dastur o‘z ishini tugatadi.

O‘z navbatida <***operator1***> va <***operator2***> ham shartli operator bo‘lishi mumkin. Ifodadagi har bir else kalit so‘zi, oldindagi eng yaqin ***if*** kalit so‘ziga tegishli hisoblanadi (xuddi ochiluvchi va yopiluvchi qavslardek). Buni inobatga olmaslik mazmunan xatoliklarga olib kelishi mumkin.

Masalan:

```
if (x==1)
if (y==1) cout<<”x=1 va y=1”;
else cout <<‘‘x< >1”;
```

Bu misolda ***«x< >1»*** xabari ***x*** qiymatini ***1*** ga teng va y qiymatini ***1*** ga teng bo‘lmagan holda ham chop etiladi. Quyidagi variantda ushbu mazmunan xatolik bartaraf etilgan:

```
if (x==1)
{
    if (y==1) cout<<”x=1 va y=1”;
}
else  cout<<”x< >1”;
```

C++ tilida shart operatorida umumiy bo‘lgan o‘zgaruvchilarni e’lon qilish man etiladi, lekin undagi bloklarda o‘zgaruvchilarni e’lon qilish mumkin va bu o‘zgaruvchilar faqat blok ichida amal qiladi. Quyidagi misolda bu holat bilan
bog’liq xatolik ko‘rsatilgan:

```
if (j>0) {int i; i=2*j;}
else i=-j;        //xato, chunki I blokdan tashqarida ko‘rinmaydi
```

Masala. Berilgan to‘rt xonali ishorasiz sonning boshidagi ikkita raqamining yig’indisi qolgan raqamlar yig’indisiga teng yoki yo‘qligi aniqlansin (raqamlar yig’indisi deganda ularga mos son qiymatlarining yig’indisi tushuniladi). Sonning raqamlarini ajratib olish uchun butun sonlar arifmetikasi amallaridan foydalaniladi:

```
#include  <iostream.h>
int main()
{
      Unsigned int n,a3,a2,a1,a0;      //n=a a a a  ko‘rinishida
      cout<<‘‘\nn-qiymatini kiriting:’’;
      cin>>n;
      If (n<1000| n>9999)
```

```
        {
         cout<<‘‘kiritilgan son 4 xonali emas!’’;
         return 1;
         }
    a3=n/1000;
    a2=n%1000/100;
    a1=n%100/10;
    a0=n%10;
    if (a3+a2==a1+a0) cout<<‘‘a3+a2=a1+a0’’;
    else cout<<‘‘a3+a2<>a1+a0’’;
    return 0;
}
```

2. SWITCH - Tanlash operatori

Shart operatorining yana bir ko‘rinishi tanlash tarmoqlanish operatori bo‘lib, uning sintsksisi quyidagacha:

switch (<ifoda>)

{

case <o‘zgarmas ifoda > : <operatorlar guruhi >; break;

case <o‘zgarmas ifoda > : <operatorlar guruhi >; break;

…

case <o‘zgarmas ifoda > : <operatorlar guruhi >; break;

default:: <operatorlar guruhi >;

}

Misol. Hafta kunlarin hosil qilish.

```
#include <iostream>
using namespace std;

int main()
}
    int n;
    cout<<”n=”; cin>>n;
switch (n)
    {
case 1:
```

```
            cout<<"Dushanba"; break;
        case 2:
            cout<<"Seshanba"; break;
        case 3:
            cout<<"Chorshanba"; break;
        case 4:
            cout<<"Payshanba"; break;
        case 5:
            cout<<"Juma"; break;
        case 6:
            cout<<"Shanba"; break;
        case 7:
            cout<<"Yakshanba"; break;
    }
    return o;
}
```

C++ tilida bu operator quyidagicha amal qiladi: birinchi navbatda **<*ifoda*>** qiymati hisoblanadi, keyin bu qiymat case kalit so‘zi bilan ajratilgan **<*o‘zgarmas ifodai*>** bilan solishtiriladi. Agar ular ustma-ust tushsa, shu qatordagi ***‘:’*** belgisidan boshlab, toki break kalit so‘zigacha bo‘lgan **<*operatorlar guruhi*>** bajariladi va boshqaruv tarmoqlanuvchi operatordan keyingi joylashgan operatorga o‘tadi. Agar **<*ifoda*>** birorta ham **<*o‘zgarmas ifoda*>** bilan mos kelmasa, qurilmaning default qismidagi **<*operatorlar guruhi*>** bajariladi. Shuni qayd etish kerakki, qurilmada default kalit so‘zi faqat bir marta uchrashi mumkin.

Tanlash operatorini qo‘llashga doir misolni qarab chiqaylik. Klaviaturadan kiritilgan ***“Jarayon davom etilsinmi?”*** so‘roviga foydalanuvchi tomonidan javob olinadi. Agar ijobiy javob olinsa, ekranga ***“Jarayon davom etadi!”*** xabari chop etiladi va dastur o‘z ishini tarmoqlanuvchi operatordan keyingi operatorlarni bajarish bilan davom ettiradi, aks holda “Jarayon tugadi!” javobi beriladi va dastur o‘z ishini tugatsin. Bu masala uchun tuziladigan dastur foydalanuvchining ***‘y’*** yoki ***‘Y’*** javoblari jarayonni davom ettirishni bildiradi, boshqa belgilar esa tugatishni anglatadi.

```
#include <iostream.h>
int main()
{
    char javob=‘ ‘;
```

```
        cout<<‘‘jarayon davom etsinmi?(‘y’,’Y’):’’;
        cin>>javob;
        switch (javob)
        {
                case ‘y’:
                case ‘Y’:
                        cout<<‘‘jarayon davom etadi!\n’’;
                        break;
                default:
                       cout<<‘‘jarayon tugadi!\n’’;
        return 0;
         }     //jarayon
 return 0;
}
```

Tanlash operatorining C++ tilidagi ko‘rinishida break va default kalit so‘zlarini ishlatmasa ham bo‘ladi. Ammo bu holda operatorning mazmuni buzilishi mumkin. Masalan, default qismi bo‘lmagan holda, agar <ifoda> birorta <o‘zgarmas ifoda > bilan ustma-ust tushmasa, operator hech qanday amal bajarmasdan boshqaruv tanlash operatordan keyingi operatorga o‘tadi. Agar break bo‘lmasa, <ifoda> birorta <o‘zgarmas ifoda > bilan ustma-ust tushgan holda, unga mos keluvchi operatorlar guruhini bajaradi va «to‘xtamasdan» keyingi qatordagi operatorlar guruhini bajarishga o‘tib ketadi. Masalan, yuqoridagi misolda break operatori bo‘lmasa va jarayonni davom ettirishni tasdiqlovchi (‘Y’) javob bo‘lgan taqdirda ekranga

Jarayon davom etadi!
Jarayon tugadi!

xabarlari chiqadi va dastur o‘z ishini tugatadi (return operatorining bajarilishi natijasida).

Tanlash operatori sanab o‘tiluvchi turdagi o‘zgarmaslar bilan birgalikda ishlatilganda samara beradi. Quyidagi dasturda ranglar gammasini toifalash masalasi yechilgan.

```
#include <iostream.h>
int main()
{
        enum Ranglar{qizil, tuq_sariq,sariq,yashil,kuk,zangori,binafsha};
        Ranglar rang;         //...
        switch(rang)
```

```
{
        case qizil:
        case tuq_sariq:
        case sariq;
                cout<<‘‘Issiq gamma tanlandi. \n’’;
                break;
        case yashil:
        case kuk:
        case zangori:
        case binafsha:
                cout<<‘‘Sovuq gamma tanlandi.\n’’;
                break;
        default:
                cout<<‘‘Kamalak bunday rangga ega emas. \n’’;
}
return 0;
}
```

Dastur bajarilishida boshqaruv tanlash operatorga kelganda, rang qiymati qizil yoki tuq_sariq yoki sariq bo‘lsa, ***“Issiq gamma tanlandi’’*** xabari, agar rang qiymati yashil yoki kuk yoki zangori yoki binafsha bo‘lsa, ekranga ***“Sovuq gamma tanlandi”*** xabari, agar rang qiymati sanab o‘tilgan qiymatlardan farqli bo‘lsa, ekranga ***“Kamalak bunday rangga ega emas”*** xabari chop etiladi va dastur o‘z ishini tugatadi. C++ tilidagi tanlash operatorida e’lon operatorlari ham uchrashi mumkin. Lekin ***switch*** operatori bajarilishida “***Sakrab o‘tish”*** holatlari bo‘lishi hisobiga blok ichidagi ayrim e’lonlar bajarilmasligi va buning oqibatida dastur ishida xatolik ro‘y berishi mumkin:

```
int k=0,n=0;
cin>>n;
switch (n)
{
int=10;       //xato,bu operator hech qachon bajarilmaydi
case 1:
        int j=20;     //agar n=2 bo‘lsa,bu e’lon bajarilmaydi
case 2:
        k+=i+j;   //xato, chunki i,j o‘zgaruvchilar noma’lum
  }
  cout<<k;
```

Tanlash operatorini qo‘llashga doir yana bir masalani qarab chiqamiz.

Masala. Quyida, sanab o‘tiluvchi turlar va shu turdagi o‘zgaruvchilar e’lon qilingan:

enum Birlik{detsimetr, kilometr, metr, millimetr, santimetr}
float y; Birlik r;

Birlikda berilgan x o‘zgaruvchisining qiymat metrlarda chop qilinsin.

```
#include <iostream.h>
 int main()
}
        enum Birlik {detsimetr, kilometr, metr, millimetr, santimetr};
                float x,y;
                Birlik r;
                cout<<‘‘uzunlikni kiriting: x=‘‘;
                cin>>x;
                cout<<‘‘ uzunlik birliklari\n’’;
                cout<<‘‘ 0-detsimetr\n’’;
                cout<<‘‘1-kilometr\n’’;
                cout<<‘‘2-metr\n’’;
                cout<<‘‘3-millimetr\n’’;
                cout<<‘‘4-santimetr\n’’;
                cout<<‘‘uzunlikni birligini tanlang; r=‘‘;
                cin>>r;

        switch (r)
          }
                case detsimetr:y=x/10; break;
                case kilometr: y=x*1000; break;
                case metr: y=x; break;
                case millimetr: y=x/1000; break;
                case santimetr: y=x/100; break;
                default:
                cout<<‘‘uzunlik birligi noto‘g’ri kiritildi!’’;
                return 0;
         }
        cout<<y<<‘‘metr’’;
 return 0;
 }
```

Nazorat savollari

1. Shart operatorlari
2. If operatori
3. If () shart operatorining blok sxemasi
4. #Include <iostream.h> va #include <math.h> deriktivalari tavfsifi
5. Switch tanlash operatori tavfsifi

2.3. C++ tilida takrorlanuvchi jarayonlarni dasturlash

Dasturlash jarayonida ba'zi bir masalalarni algoritmlari tarkibidagi buyruqlar bir necha marta bajarilishiga to‘g'ri keladi. Agar algoritm tarkibidagi bir necha marta takrorlanishi kerak bo‘lgan buyruqlarni takrorlanuvchi jarayonlar asosida dasturlash tillarida tasvirlanmasa, bu buyruqlarni barchasini bajarish murakkablashadi. Elektron hisoblash mashinalarini insoniyatdan farqi shundaki, insoniyatda bir nechta buyruqlarni bajarish davomida toliqish holatlari bo‘lishi mumkin elektron mashinalarga takrorlanishni qanchaligini ma'lum bir buyruqlar asosida berilsa, ular barchasini charchamasdan bajaradi. Tarmoqlanuvchi jarayonlarni masalan matematikada ixtiyoriy ketma ketliklarni yig'indisini hisoblash oddiy usullar bilan hal etilmaydigan holatlarda qo‘llash mumkin.

Tarif: *Algoritmning qandaydir qismidagi buyruqlar ikki va undan ortiq bajarilishiga* ***takrorlanuvchi jarayonlar*** *deyiladi.*

Yuqoridagi ta'rifga etibor qaratsak, demak algoritmning qandaydir qismi ikki va undan ortiq bajarilishi mumkin bo‘lgan holatlar ham mavjud. Bunda dasturchiga shunday vazifa qo‘yiladiki takrorlanish holatini bir yaxlit buyruq asosida kompyuterga qulay usulda berish kerak.

Takrorlanuvchi jarayonlarni quyidagi blok sxema ko‗rinishda C++ dasturlash tilida tasvirlash mumkin.

TSIKL OPERATORLARI

Tsikl - operatsiyalar blokining takrorlanishini o‘z ichiga oladi, ya'ni tsikl tanasi deyiladi.

- **Tsikl o‘zi nima?**
- **Tsikl** *(yun. Kyklos (kayklos) - doyra) - ma'lum vaqt ichida takrorlanib turadigan hodisa, jarayon va b. ning har bir davrasi (mas, yillik* ***Tsikl****);*

C++ tilida tsikl 3 turi mavjud:

• **While** - old sharti bilan sikl;
• **Do**…while – keyingi shart;
• **For** - parametrik sikl;

1. While- old shart bilan tsikl operatori

Old shartli sikl sintaksisi. **“While”** so‘zi o‘zbek tilida “***mobaynida***”, “***bo‘lgunga qadar***” degan ma’noni anglatadi.

Tarif: *Agar takrorlanishlar soni ma’lum bir shartlar asosida aniqlansa, bunday jarayonlar shartli takrorlanuvchi jarayonlar deyiladi.*

Shartsiz o‘tish operatori va tarmoqlanuvchi operatorlar yordamida ham shartli takrorlanuvchi jarayonlarni dasturlash imkoniyati mavjud. Lekin bunday holatlarda bitta amalni bajarish uchun bir nechta operatorlarni ishlatish kerak bo‘ladi. Shartli takrorlanuvchi operatorlar bajarilish holatlariga qarab turlarga ajratiladi. Shartli takrorlanuvchi operatorlar quyidagi turlari mavjud:

-Shart oldi takrorlanuvchi operatorlar;

- Shart keyin takrorlanuvchi operatorlar.

Shart oldi takrorlanuvchi jarayonlar bajarilish holati har bir takrorlanish oldidan shart tekshirilib keyin takrorlanish tanasidagi operatorlar bajariladi. Agar takrorlanish holati boshidan shart yolg’on qiymat qabul qilsa, takrorlanish bir marta ham bajarilmaydi. Shart oldi takrorlanuvchi operatorlarning C++ dasturlash tilida ifodalash uchun while operatori yordamida tasvirlanadi.

Old shartga ega bo‘lgan sikl - bu boshlanishidan oldin belgilangan ba’zi shartlar **rost** bo‘lganda bajariladigan tsikl. Bu shart **tsikl** tanasining bajarilishidan oldin tekshiriladi, shuning uchun tana bir marta ham bajarilmasligi mumkin (agar shart boshidanoq noto‘g’ri bo‘lsa).

Ta’rif: *Agar shartli takrorlanuvchi jarayonlar tarkibidagi shart takrorlanishdan oldin tekshirilsa, shart oldi takrorlanuvchi jarayonlar deyiladi.*

Takrorlash tanasi – bu takrorlash operasiyasi ichida amalga oshiriladigan amallar jamlanmasi hisoblanadi. Takrorlash tanasi ***takrorlash operasiyasi*** - ***{ }*** qavslar ichiga yoziladi.

Takrorlanuvchi operator tarkibiga beriladigan shart tahlil qilinib yozilish kerak, chunki shart hech qachon yolg’on qiymat qabul qilmasa, dastur cheksiz ishlashga to‘g’ri keladi. Takrorlanish hech qachon cheksiz bo‘lishi mumkin emas, aks holda algoritmning diskretlik hossasi buziladi.

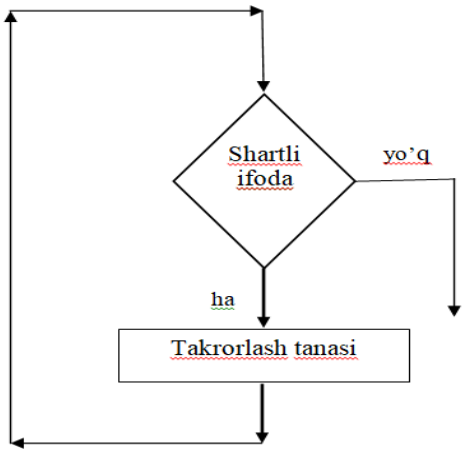
While takrorlash operatori quyidagi ko‘rinishga ega:

```
While (shart)
 {
   takrorlash tanasi;
}
```

Agar shart chin qiymat qabul qilib tursa **{}** ichidagi operatorlar bajarilaveradi, qachonki shart yolg’on bo‘lgandagina takrorlanish o‘z ish faoliyatini to‘xtatadi.

Takrorlash strukturasi bir ifoda yoki blokni ma’lum bir shart to‘g’ri (true) bo‘lishi davomida qaytarish imkonini beradi. Qaytarilayotgan ifoda shartga ta’sir ko‘rsatishi kerak. Ma’lum bir vaqt o‘tgandan keyin shart false ga o‘zgartilishi kerak. Bo‘lmasa while (davomida) tugatilmaydi. while faqat o‘zidan keyin kelgan ifodaga ta’sir qiladi. Agar biz bir guruh amallarni qaytarmoqchi bo‘lsak, ushbu blokni {} qavslar ichiga olishimiz kerak. Shart takrorlanuvchi blokning boshida tekshirilgani sababli, agar shart noto‘g’ri bo‘lib chiqsa, blokni hech ijro ko‘rmasligi ham mumkin.

While operatori tarkibidagi shart yolg’on qiymat qabul qilganda operatorlar bajarilmasdan qoladi, shart chin qiymat qabul qilgandagina operatorlar bajariladi. Ba’zi hollarda shart takrorlanish boshidan yolg’on qiymat qabul qiladi, bunda takrorlanish bir marta ham bajarilmaydi. Shart chin qiymat qabul qilib, lekin takrorlanish tanasida shart tarkibi o‘zgartirilmasa, takrorlanish cheksiz bo‘lib qoladi.



2.3-rasm. While takrorlash operatorining blok-sxemas

1-misol. While yordamida 1 dan 1000 gacha bo‘lgan barcha butun sonlar yig’indisini hisoblaydigan dastur kodini yozamiz.

```
#include <iostream>
using namespace std;
int main ( )
{
      int i = 0; // sikl hisoblagichini ishga tushiramiz.
      int sum = 0; // summa hisoblagichini ishga tushiramiz.
      while (i < 1000)
      {
         sum += i
         i++;
      }
```

```
        cout << "1 dan 1000 gacha sonlar yig'indisi= " << sum << endl;
        return 0;
    }
```

2-misol. While yordamida faktorialni hisoblaydigan dastur kodini yozamiz.

```
#include <iostream>
using namespace std;
int main ( )
{
    int factorial = 1;
    int i = 1;
    while(i <= n)   // Faktorialni hisoblash.
    {
       factorial *= i;
       i++; // Schyotchikni oshirish.
       cout << "factorial = " << factorial << endl;
    }
return 0;
}
```

2. Do… while - takrorlash operatori

Do takrorlash operatori – ***while*** operatoriga o‘xshash. ***Do*** takrorlash operatorining ***while*** operatoridan farqi: undagi shartli ifoda takrorlash tanasidan keyin ketadi.

Shu sabali, takrorlanuvchi jarayon shart tekshirilgunga qadar bir martta bajariladi. Shart rost bo‘lsa, takrorlash jarayoni qaytariladi va bu jarayon toki shart bajarilmay qolgunga qadar davom etadi. Agar **do** operatori tanasidagi qaytarilishi kerak bo‘lgan amallar ko‘p bo‘lsa, ular ***{ }*** qavslar ichiga olib qo‘yiladi.

Shartli takrorlanuvchi jarayonlar tarkibida takrorlanish oxirida shart tekshirilish holatlarida shart keyin takrorlanuvchi algoritmlar qo ‘llaniladi. Shart keyin takrorlanuvchi jarayonlar bajarilish holati har bir takrorlanish bir qadam bajarilgandan keyin shart tekshiriladi. Agar takrorlanish holati boshidan shart yolg’on qiymat qabul qilsa takrorlanish bir marta bajarilib qoladi.

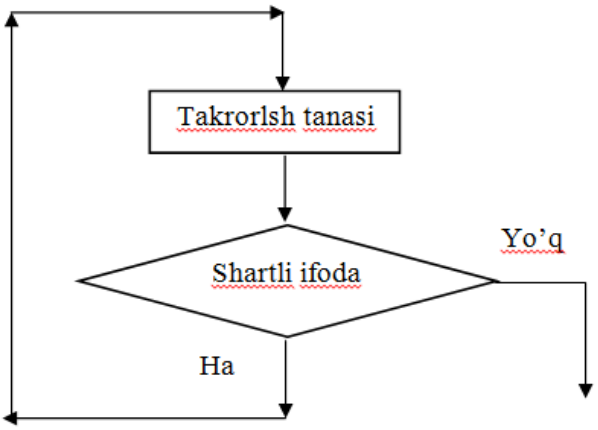
Ta’rif: *Agar shartli takrorlanuvchi jarayonlar tarkibidagi shart takrorlanish oxirida tekshirilsa, bunday jarayonlar shart keyin takrorlanuvchi jarayonlar deyiladi.*

Shart keyin takrorlanuvchi jarayonlarni takrorlanish oldindan shart tekshirilish iloji bo‗lmagan vaqtlarda foydalanish mumkin. Shart keyin

takrorlanuvchi jarayonlarda takrorlanish tanasi tarkibidagi operatorlar bir marta bajariladi va keyin shart tekshiriladi.Agar shart yolg'on bo'lsa, takrorlanish to'xtatiladi.

Do takrorlash operatori quyidagi ko'rinishga ega:

```
Do
 {
   takrorlash tanasi;
 }
While (shartli ifoda);
```



2.4-rasm. **Do** takrorlash operatorining blok-sxemasi.

3-misol. Do while sikli yordamida 1 dan 1000 gacha bo'lgan sonlar yig'indisini topish masalasini yechish.

```
#include <iostream>
using namespace std;
int main ()
{
   int i = 0;    // sikl hisoblagichini ishga tushiramiz.
   int sum = 0; // summa hisoblagichini ishga tushiramiz.
 do
    {    // tsiklni bajaramiz.
       sum += i;
        i++;
    }
 while (i < 1000);   // toki shart bajariladi.
 cout << "1 dan 1000 gacha sonlar yig'indisi = " << sum << endl;
 return 0;
}
```

***4*- misol.** *1- hadi b va maxraji q berilganda yig'indisi n dan kichik bo'lgan geometrik progressiyani eng katta hadini toping.*

```
#include <iostream.h>
int main()
{
  int b,q,s=0,n;
        cin>>n>>b>>q;
        do
         {
            s=s+b;          // yig'indini hisoblash
            b=b*q;          // umumiy hadini hisoblash
         }
        while(s<n);
cout<<b/q;
return 0;
}
```

do/while ifodasi while strukturasiga o'xshashdir. Bitta farqi shundaki while da shart boshiga tekshiriladi. **do/while** da esa takrorlanish tanasi eng kamida bir marta ijro ko'radi va shart strukturaning so'ngida test qilinadi. Shart true bo'lsa blok yana takrorlanadi. Shart false bo'lsa **do/while** ifodasidan chiqiladi. Agar **do/while** ichida qaytarilishi kerak bo'lgan ifoda bir dona bo'lsa **{}** qavslarning keragi yo'qdir.

3. For - parametrik sikl

Amalda shunday masalalar ham uchraydiki, ularni hal qilish dasturlarini hozirgacha tanishgan buyruqlar yordamida tuzib bo'lmaydi. Ular shundaymasalalarki, natijaga erishish uchun qo'yilgan shartga bogliq holda bir yoki bir nechta amal yoki buyruqlar ketma-ketligini takror-takror bajarishga to'gri keladi. Takrorlanish buyruqlaridan, ayniqsa, aniq fanlarga doir masalalarni hal qilishda ko'p foydalaniladi.

Takrorlanuvchi jarayonlarni takrorlanish soni aniq bo'lgan holatlardagina parameter bo'yicha takrorlash usulidan foydaliniladi. Takrorlanuvchi jarayonlarni parametr bo'yicha C++ dasturlash tilida tasvirlash uchun, albatta, takrorlanish soniga e'tibor berish kerak. Parametr bo'yicha takrorlanuvchi jarayonlarga, masalan, birdan *n* gacha sonlarning kvadratlarini yig'indisini topish bunda takrorlanish soni aniq, ya'ni birdan dan *n* gacha deb berilyapti.

Parametr bo‘yicha takrorlanuvchi jarayonlar takrorlanish oshishi yoki kamayishiga qarab ikki turga bo‘linadi:

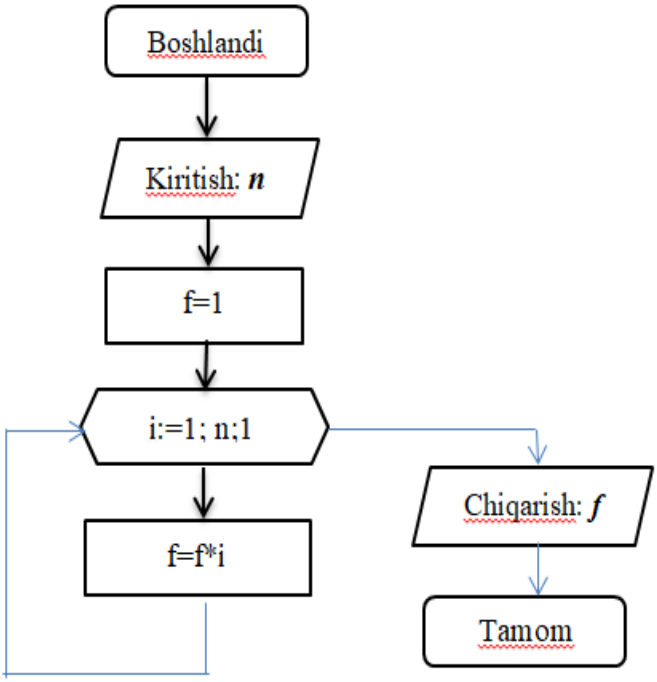
-qadam +1ga teng b’lgan takrorlanish;

- qadam -1ga teng bo‘lgan takrorlanish.

For tsiklga kirishdan oldin takroriy takrorlashlar soni ma’lum bo‘lgan holatlarda qo‘llanilishi mumkin. U quyidagi shaklga ega:

For (<takrorlash parametri qiymati >; <shart>; <inkrement>)
{
<Takrorlash tanasi >
}

Takrorlash parametri qiymati - hisoblagichning dastlabki parametrlarini o‘rnatish;
Shart - tsikldan chiqish sharti, u noto‘g’ri bo‘lishi bilanoq, tsikl chiqib ketadi;
Inkrement - hisoblagich ko‘paytirish buyrug’i.



2.5-rasm. **For** takrorlash operatorining blok-sxemasi

1-misol. For yordamida 1 dan 1000 gacha bo‘lgan barcha butun sonlar yig’indisini hisoblaydigan dastur kodini yozamiz.

```
#include <iostream>
using namespace std;
int main ( )
{
    int Summa = 0; // summa hisoblagichini ishga tushiramiz.
    for (int i=1;i<=1000;i++)
```

```
    {
        Summa += i;
    }
    cout << "yig'indi= " << Summa << endl;
    return 0;
}
```

Izoh:

Dasturdagi takrorlash operatori o‘z ishini, ***i takrorlash parametriga*** (***takrorlash sanagichiga***) boshlang’ich qiymat – ***1*** sonini berishdan boshlaydi va har bir takrorlash qadamidan (***itaratsiyadan***) keyin uning qiymati bittaga oshadi. Har bir takrorlash qadamida takrorlash tanasidagi operator bajariladi, ya’ni summa o‘zgaruvchisiga ***i*** ning qiymati qo‘shiladi. Takrorlash sanagichi ***i*** ning qiymati ***1001*** bo‘lganda “***i<=1000***” takrorlash sharti (***0***-qiymati) bo‘ladi va takrorlash tugaydi.

Natijada oshqaruv takrorlash operatoridan keyingi operatorga o‘tadi va ekranga yig’indi chop etiladi.

2-misol. N = 1 * 2 * 3 * ... * N faktorialni hisoblang.

```
#include <iostream>
using namespace std;
int main ( )
{
   int  n, factorial = 1;
   for (int i = 1; i <= n; i++) // tsikl
   {
      factorial *= i;
      i++;
   }
      cout << "factorial = " << factorial << endl;
      return 0;
```

Nazorat savollari

1. Tsikl operatorlari
2. While- old shart bilan tsikl operatori
3. Takrorlash tanasi nima?
4. While takrorlash operatorining blok-sxemasi
5. Do… while - takrorlash operatori
6. Do takrorlash operatorining blok-sxemasi
7. Tsilk schyotchigi nima?

8. For tsikl operatorining bajarilish tartibi
9. Tsikl tanasini bajarilish tartibi
10. Tsiklda schyotchikni e’lon qilish
11. Tsikl qadami nima?
12. For tsikl operatorini bajarilishiga misol

2.4. C++ tilida funksiyalar. Lokal va global o‘zgaruvchilar

1. Funksiyalar

Dasturlash mobaynida bir xil ifodalarni, hisoblash jarayonlarini qayta – qayta hisoblashga to‘g’ri keladi. Dasturlash tillarida, kompyuter hotirasini va dasturchining vaqtini tejash maqsadida, bunday takkorlanuvchi jarayonlarni dasturda ajratib yozib, unga asosiy daturdan, boshqa funksiyalardan murojaat qilish imkoniyatlari keltirilgan. Dasturning istalgan qismidan murojaat qilib, bir necha bor ishlatish mumkin bo`lgan operatorlar guruhiga funksiya deyiladi.

Funksiya faqat chaqirilgan vaqtda ishlaydigan kod blogi hisoblanadi. Funksiyaga parametrlar sifatida ma’lumotlarni uzatish mumkin. Funksiyalar muayyan bir vazifani bajarish uchun ishlatiladi. Kod yozish jarayonida yaratgan funksiyadan bir yoki bir nechta marta ishlatish yoki umuman ishlatmaslik imkoniyatiga egasiz.

Funksiya yaratish uchun birinchi funksiya qaytaradigan tip yoki funksiya turi keyin nomi va **()** ochiladi. Qavs ichida siz ma’lumotlar ya’ni parametrlarni qabul qilasiz.

C++ da dasturlashning asosiy bloklaridan biri funksiyalardir. Funksiyalarning foydasi shundaki, katta masala bir necha kichik bo‘laklarga bo‘linib, har biriga alohida funksiya yozilganda, masala yechish algoritmi ancha soddalashadi. Bunda dasturchi yozgan funksiyalar C++ ning standart kutubhonasi va boshqa firmalar yozgan kutub-honalar ichidagi funksiyalar bilan birlashtiriladi. Bu esa ishni osonlashtiradi.

- *ko‘p holda dasturda takroran bejariladigan amalni funksiya sifatida yozish va kerakli joyda ushbu funksiyani chaqirish mumkin*;
- *funksiyani programma tanasida ishlatish uchun u chaqiriladi, yani uning ismi yoziladi va unga kerakli argumentlar beriladi.*

Funktsiyani standart ta’rifi:

<natija turi> <funktsiya nomi>
<formal parametrlar ro‘yxati>
{
    < ob’ektlarni aniqlash >;

< bajariladigan operatorlar >;
}

( ) qavslar ushbu funksiya chaqirig'ini ifodalaydi.

Masalan: **foo (); k = square (l);**

Standart kutubhonaning matematik funksiyalari ko‘pgina amallarni bajarishga imkon beradi. Biz bu kutubhona 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 kutubhonadan “*sin” funksiyasini chaqiradi. Kirish qiymati* sifatida *m* ni berdik. Javob, yani funksiyadan qaytgan qiymat *k* ga berildi.

Funktsiyani tipi sifatida faqat butun va haqiqiy tiplar olinishi mumkin:

- *char,*
- *int,*
- *long,*
- *float,*
- *double*
- *long double.*

Bundan tashqari bu erda signed va unsigned larni ham ishlatish mumkin. Agar funktsiyani tipi ko‘rsatilmagan bo‘lsa, u holda uning tipi **int** deb qabul qilinadi. Natijani qaytarmaydigan funktsiyalar C++ da mavjud. Bu holda ularni oldiga **void** qo‘yiladi.

* *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;
```

1-misol. To‘g’ri burchakli uchburchakning katetlari berilgan. (3, 4), (6, 8), (12, 5) bo‘lgan xollar uchun uchburchak gipotenuzasini hisoblovchi dastur tuzilsin.

1) Parametrli funksiya:

```
#include <iostream.h>
#include <math.h>
// funksiya prototipi
float hisobla(float , float );
int main()
{
     float c;
     c = hisobla(3, 4);
     cout << c << endl;
     c = hisobla(6, 8);
     cout << c << endl;
c = hisobla(12, 5);
cout << c << endl;
system ("pause");
return 0;
}
   float hisobla(float a, float b)
{
    //lokal o‘zgaruvchi
    float natija;
    natija = sqrtf(a*a + b*b);
return natija;
}
```

2) Void toifasidagi parametrli funksiya:

```
#include <iostream.h>
#include <math.h>
// funksiya prototipi
void hisobla(float , float );
int main()
{
      hisobla(3, 4);
      hisobla(6, 8);
      hisobla(12, 5);
      system ("pause");
      return 0;
}
     void hisobla(float a, float b)
```

```
{
    float c;
    c = sqrtf(a*a + b*b);
    cout << c << endl;
}
```

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 funksiyalar kutubhonasining bazi bir a’zolarini beraylik. **x** va **y** o‘zgaruvchilari ***double*** tipiga ega.

Funksiyalar dasturchi ishini juda yengillashtiradi. Funksiyalar yordamida programma ***modullashadi***, qismlarga bo‘limaydi. 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 argumentlaridir***, ular faqat ushbu funksiya ichida ko‘rinadi va qo‘llaniladi. Bunday o‘zgaruvchilar ***local*** (***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.

Bitta parametrni qabul qiluvchi funksiyalarni tavsiflash va ulardan foydalanish misollari.

2-misol. Butun son tipidagi bitta parametrni olib, uni 5 ga ko‘paytiruvchi va natijani qaytaruvchi funksiya. Funktsiya natijani chop etmaydi.

```
// parametrni 5 ga ko‘paytiruvchi funksiya
int Mult5(int d)
{
   int res;
   res = d * 5;
   return res; //  natijani qaytarish
}
```

Funktsiyani boshqa dastur kodidan chaqirish:

…

```
// funktsiyani boshqa dastur kodidan chaqirish
int x, y;
x = 20;
y = Mult5(x);       // y = 100
y = Mult5(-15);   // y = -75

...
```

2. Lokal va global o‘zgaruvchilar

O‘zgaruvchilar funktsiya tanasida yoki undan tashqarida e’lon qilinishi mumkin. ***Funktsiya ichida e’lon qilingan o‘zgaruvchilarga lokal o‘zgaruvchilar deyiladi.*** Bunday o‘zgaruvchilar xotiradagi programma stekida joylashadi va faqat o‘zi e’lon qilingan funktsiya tanasida amal qiladi. Boshqaruv asosiy funktsiyaga qaytishi bilan lokal o‘zgaruvchilar uchun ajratilgan xotira bo‘shatiladi (o‘chiriladi). Har bir o‘zgaruvchi o‘zining amal qilish sohasi va yashash vaqti xususiyatlari bilan xarakterlanadi.

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

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

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

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

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

```
int x=a+b;   // lokal o‘zgaruvchi

 return x;
}
```

Global o‘zgaruvchilar programma matnida funktsiya aniqlanishidan tashqarida e’lon qilinadi va e’lon qilingan joyidan boshlab programma oxirigacha amal qiladi.

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

Yuqorida keltirilgan programmada kompilyatsiya xatosi ro‘y beradi, chunki **f1()** funktsiya uchun x o‘zgaruvchisi noma’lum hisoblanadi.

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

Shuni qayd etish kerakki, tajribali programma tuzuvchilar imkon qadar global o‘zgaruvchilarni ishlatmaslikka harakat qilishadi, chunki bunday o‘zgaruvchilar qiymatini programmaning ixtiyoriy joyidan o‘zgartirish xavfi

mavjudligi sababli programma ishlashida mazmunan xatolar yuzaga kelishi mumkin. Bu fikrimizni tasdiqlovchi programmani ko‘raylik.

```
# include <iostream.h> //global o‘zgaruvchi e’loni
int test=100;
void Chop_qilish(void );
int main()
{
     int test=10;     // lokal o‘zgaruvchi e’loni

      Chop_qilish();  // global o‘zgaruvchi chop qilish funktsiyasini
chaqirish
     cout<<”Lokal o‘zgaruvchi: ”<<test<<‘\n’;
 return 0;
  }
    void Chop_qilish(void)
  {
     cout<<”Global o‘zgaruvchi: ”<<test<<‘\n’;
}
```

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

Global o‘zgaruvchi: 100
Lokal o‘zgaruvchi: 10

:: amali

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

quyidagi programani keltiramiz:

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

3. C++ tilidada ko‘rsatkichlar

Ko‘rsatkichlar. Ko‘rsatkich - xotira uyasining ***unikal adresini*** saqlaydigan o‘zgaruvchi. ***Ko‘rsatkich*** operativ xotiradagi biron-bir o‘zgaruvchi mavjud bo‘lishi mumkin bo‘lgan biron-bir joyni belgilaydi. Ko‘rsatkichlarning qiymatlarini o‘zgartirish, turli variantlarda qo‘llash mumkinki, bu dasturning moslashuvchanligini oshiradi.

Ko‘rsatkich. O‘zining qiymati sifatida xotira manziliini ko‘rsatuvchi (saqlovchi) o‘zgaruvchilarga - ko‘rsatkich o‘zgaruvchilar deyiladi.

Ko‘rsatkich xotiradagi o‘zgaruvchining manzilidir. O‘zgaruvchiga ko‘rsatgich - bu ma’lum turdagi ob’ektga ko‘rsatgichni ushlab turish uchun maxsus yaratilgan o‘zgaruvchi. O‘zgaruvchining manzilini bilish ba’zi dasturlarning ishini ancha soddalashtirishi mumkin. Ko‘rsatkichlar C++ tilida uchta asosiy qo‘llaniladi:

1. Massiv elementlariga tezkor kirishni ta’minlash.
2. Funksiyalarga uzatilgan parametrlarni o‘zgartirishga ruxsat bering.
3. Ro‘yxatlar kabi dinamik ma’lumotlar tuzilmalarini qo‘llab-quvvatlash.

Ko‘rsatkich odatda tipga ega bo‘lib quyidagicha e’lon qilinadi:

<turning nomi>*<ko‘rsatkichning nomi>=<dastlabki qiymat>

Misol uchun:

```
int *pr;
char *alfa;
```

Bu holda ko‘rsatkichlar noaniq qiymatga ega bo‘ladi.

- *Ko‘rsatkichlar ta’riflanganda ularning tiplari ko‘rsatilishi shart.*
- *Ko‘rsatkichlarni initsializatsiya kilish ya’ni boshlang’ich qiymatlarini kiritish mumkin.*

Ma’lum turdagi biron-bir o‘zgaruvchi adresi yoki NULL qiymat dastlabki qiymat bo‘lishi mumkin. Ko‘rsatkichlarga boshlang’ich maxsus NULL qiymati berilsa bunday ko‘rsatkich bo‘sh ko‘rsatkich deb ataladi.

Biron-bir o‘zgaruvchi adresini olish hamda uni ko‘rsatkichga qiymat sifatida berish uchun «&» operatori qo‘llanadi.

Ampersand (**&**, ba’zan - ***ampersand***; ingliz ***ampersand***) - "**va**" birlashmasi o‘rnini bosuvchi logogramma.

Misol:

```
int I=100;
int*p=&I;
unsigned longint *ul=NULL;
```

Boshqa o‘zgaruvchilar kabi, ko‘rsatkichlardan foydalanish uchun ularni e’lon qilish, toifasini aniqlash shart.

Masalan:

int ***countPtr**, count;

bu yerda **countPtr** - int toifasidagi ob’ektga ko‘rsatkich, **count** (hisoblash) esa oddiy butun **(int)** toifasidagi o‘zgaruvchi.

Ko‘rsatkichlarni e’lon qilishda har bir o‘zgaruvchi oldigan <*> qo‘yilishi shart.

Izoh: Ko‘rsatkich o‘zgaruvchilarini e’lon qilishning uchta usuli mavjud, ammo ***birinchi usul*** afzal ko‘riladi:

String* mystring;

string * mystring;

string * mystring;

*-operator(**dereference-*imtiyoz*** operatori) yordamida o‘zgaruvchining qiymatini olish uchun ko‘rsatgichdan foydalanishingiz mumkin:

Misol 1:

```
#include <iostream>
#include <string>
using namespace std;

int main()
```

```
{
    string food = "Palov";
    string* ptr = &food;
    cout << ptr << "\n";
    cout << *ptr << "\n";
    return 0;
}

    0x6dfed4
    Palov
```

Misol 2: Ko‘rsatkich ko‘rsatayotgan manzili qiymatini dasturi

```
#include <iostream.h>
int main()
{
    int n = 5;
    int * nPtr;
    // & manzilini olish amali
    nPtr = &n;
   cout << "n=" << n << endl;
      *nPtr = 15;
  cout << "n=" << n << endl;
  cout << "\nKo‘rsatkich qiymati,\n";
  cout << "ya’ni ko‘rsatkich ko‘rsatayotgan manzili=" <<
nPtr<<endl;
  cout << "Ko‘rsatkich ko‘rsatayotgan manzili qiymati="
      <<*nPtr<<endl;
  system ("pause");
   return 0;
}
```

Manzil operatori & (ampersend)

O‘zgaruvchi ishga tushirilganda avtomatik ravishda bo‘sh xotira manzili tayinlanadi va biz o‘zgaruvchiga tayinlagan har qanday qiymat xotirada shu manzilda saqlanadi. Masalan:

int b = 8;

Ushbu ko‘rsatma protsessor tomonidan bajarilganda, operativ xotiraning bir qismi ajratiladi. Misol tariqasida, ***b*** o‘zgaruvchisiga xotira joylashuvi raqami ***150***

berilgan deylik. Dastur ifoda yoki ko‘rsatmada ***b*** o‘zgaruvchiga duch kelganda, qiymatni olish uchun ***150*** xotira manziliga qarash kerakligini tushunadi.

Yaxshi xabar shundaki, biz qaysi aniq xotira manzillari qaysi o‘zgaruvchilarga ajratilganligi haqida tashvishlanishimiz shart emas. Biz shunchaki o‘zgaruvchiga unga tayinlangan identifikator orqali murojaat qilamiz va kompilyator bu nomni tegishli xotira manziliga o‘zgartiradi. Biroq, bu yondashuv ba’zi cheklovlarga ega, biz ushbu va keyingi darslarda muhokama qilamiz.

Manzil operatori & ma’lum bir o‘zgaruvchiga qaysi xotira manzili tayinlanganligini aniqlash imkonini beradi. Bu juda oddiy:

```
#include <iostream>
 int main()
{
   int a = 7;
   std::cout << a << ‘\n’;      // a o‘zgaruvchining qiymatini chop etish
   std::cout << &a << ‘\n’;    // a o‘zgaruvchining xotira manzilini chop etihs

   return 0;
}
```

Natija:

7
0046FCF0

Misol:

1. $A=\sqrt[3]{xtg(5y)+arcctg(\frac{x-2}{y+5})}$; $x=\begin{cases}3; & y>5 \\ 2^{y}; & y=5 \\ \cos^{3}(5y); & y<5\end{cases}$; $y=c^{2}b+\cos^{2}(b^{c})$ $b\in R$; $c=0{,}04$;

Dastur kodi	Natija

#include <iostream>
#include <cmath>
using namespace std;
int main()
{
  float c=0.04, b, y, x, A;
  cout<<"b="; cin>>b;
	b=1
	A=-0.711358

```
   y=pow(c,2)*b+pow(cos(pow(b,c)),2);
   if(y>5)
   {
     x=3;
   }
    else
   {if(y=5)
   {x=pow(2,y);
  } else
   {x=pow(cos(5*y),3);
  }
}
      A=pow((x*1/tan(5*y)+atan((x-2)/(y+5))),1/3);
      cout<<"A="<<A;
    return 0;
}
```

Nazorat savollari

1. Funktsiya tushunchasi
2. Dasturda funktsiyani e’lon qilish
3. Lokal o‘zgaruvchilar
4. Funktsiyani yaratishga misollar
5. Funktsiyalarni aniqlash
6. Funksiya parametrlari va argumentlari

III-BOB. C++ TILIDA MASSIVLAR VA ULAR BILAN ISHLASH

3.1. Massiv tushunchasi. Massivlar bilan ishlash

Ko‘p holarda, bir nacha sonlarni yoki o‘zgaruvchilarni bitta nom ostida tuzib chiqishga to‘g’ri keladi. Buning uchun quyidagicha ish bajariladi. Bu sonlar bitta to‘plamga (**massivga)** tuzib chiqiladi va unga biror nom beriladi.

Ta’rif. *Bitta nom bilan ataladigan va ma’lum tartibda joylashtirilgan sonlar (o‘zgaruvchilar) to‘plami massiv deb ataladi. Massivga kirgan har bir son (o‘zgaruvchi) massiv elementi deb ataladi.*

Massivning elementlari massivda tuzilgan o‘rniga qarab nomerlab chiqiladi va nomerlar massiv elementlarining **indeksi** deb ataladi.

Massivga murojat qilinayotganda massiv nomi ko‘rsatiladi va uning yoniga **[ ] qavslar** ichiga bu massivning jami elementlar soni ham yozib qo‘yildi.

Odatda massivlarga zarurat, katta hajmdagi, lekin cheklangan miqdordagi va tartiblangan qiymatlarni qayta ishlash bilan bog’liq masalalarni yechishda yuzaga keladi.

Massivlarni matematikadagi sonlar vektoriga o‘xshatish mumkin, chunki vektor ham o‘zining individual nomiga ega va u fiksirlangan miqdordagi bir turdagi qiymatlardan - sonlardan iboratdir.

Demak, **massiv** - bu fiksirlangan miqdordagi ayrim qiymatlarning (massiv elementlarining) tartiblangan majmuasidir. Barcha elementlar bir xil turda bo‘lishi kerak va bu tur element turi yoki massiv uchun tayanch tur deb nomlanadi.

Dasturda ishlatiladigan har bir aniq massiv o‘zining individual nomiga ega bo‘lishi kerak. Bu nomni to‘liq o‘zgaruvchi deyiladi, chunki uning qiymati massivning o‘zi bo‘ladi.

1. Massivlarni e’lon qilish

C++ tilida massivlar quyidagich e’lon qilinadi:

<tur> <nom> [<uzunlik>]={boshlang’ich qiymatlar};
<tur> <nom> [<uzunlik>];

Bu erda <uzunlik> - o‘zgarmas ifoda.

Misollar:

```
int m[6]={1,4,-5,2,10,3];
float a[4];
```

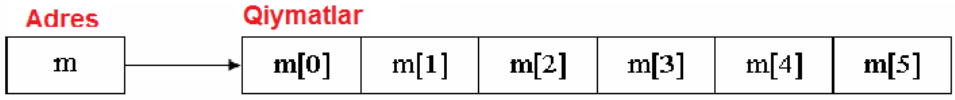
Massiv **statik** va **dinamik** bo‘lishi mumkin. **Statik** massivning uzunligi oldindan ma’lum bo‘lib, u xotirada ma’lum adresdan boshlab ketma-ket

joylashadi. **Dinamik** massivni uzunligi programma bajarilish jarayonida aniqlanib, u dinamik xotiradagi ayni paytda bo‘sh bo‘lgan adreslarga joylashadi. **Masalan**,

int m[6];

ko‘rinishida e’lon qilingan bir o‘lchamli massiv elementlari xotirada quyidagicha joylashadi:

int **m**[20]={1,4,-5,2,10,3,0,60];



3.1-rasm. Bir o‘lchamli massivning xotiradagi joylashuvi

Massiv elementiga quyidagicha murojat qilinadi:

<massiv nomi >[<indeks>]

Massiv uzunligini **sizeof(m)** (sayz of) amali orqali aniqladi.

Massiv e’lonida uning elementlariga boshlang’ich qiymatlar berish mumkin va uning bir nechta variantlari mavjud.

1) o‘lchami ko‘rsatilgan massiv elementlarini to‘liq initsializatsiyalash:

int t[5]={-10,5,15,4,3};

Bunda 5 ta elementdan iborat bo‘lgan t nomli butun turdagi bir o‘lchamli massiv e’lon qilingan va uning barcha elementlariga boshlang’ich qiymatlar berilgan. Bu e’lon quyidagi e’lon bilan ekvivalent:

int **t**[5];

t[0]=-10; t[1]=5; t[2]=15; t[3]=4; t[4]=3;

2) o‘lchami ko‘rsatilgan massiv elementlarini to‘liqmas initsializatsiyalash:

int **t**[5]={-10,5,15};

Bu erda faqat massiv boshidagi uchta elementga boshlang’ich qiymatlar berilgan. Shuni aytib o‘tish kerakki, massivning boshidagi yoki o‘rtasidagi elementlariga qiymatlar bermasdan, uning oxiridagi elementlarga boshlang’ich qiymat berish mumkin emas.

3) o‘lchami ko‘rsatilmagan massiv elementlarini to‘liq initsializatsiya-lash:

int **t**[]={-10,5,15,4,3};

Bu misolda massivni barcha elementlariga qiymatlar berilgan hisoblanadi, massiv uzunligi kompilyator tomonidan boshlang'ich qiymatlar soniga qarab aniqlanadi. Agarda massiv uzunligi berilmasa, boshlang'ich qiymati berilishi shart.

Massivni e'lon qilishga misollar:

```
shar ch[4]= {'a','b','c','d'}; // belgilar massivi
int in[4] = {10,20,30,40};     // butun sonlar massivi
char str[]= "abcd";
```

Bitta ko'rsatmada bir nechta massivlarni e'lon qilish mumkin, lekin ularning har biri uchun o'lcham ko'rsatilishi kerak.

```
int a[4], b[8];
```

2. Amaliy misollar

1-misol. Massivni konsoldan kiritish va chiqarish.

```
#include <iostream>
 using namespace std;
 int main()
{
    int i, m[10]; // Bitta butun son va bitta massivni e'lon qilamiz
      for(i = 0; i < 10; i++)
     {
        cin >> m[i]; // Klaviaturadan massiv qiymatlarini kiritamiz
     }
     for(i = 0; i < 10; i++)
    {
      cout << "m[" << i << "]= " << m[i] << '\n'; // Ekranga massiv
                elementlari chiqadi
    }
     return 0;
}
```

2-misol. 100 ta butun sondan iborat A massivi berilgan. Ushbu massiv elementlarining yig'indisini toping.

```
// 100 ta butun A massiv elementlari yig'indisi
   int A[100];
```

```
    int suma;   // summani o‘z ichiga olgan o‘zgaruvchi
    int i;      //  qo‘shimcha o‘zgaruvchi
// A massivni kiritish
// ...

//  Yig’indini hisoblash
    suma = 0; // Yig’indini nolga tenglashtirish
    for (i=0; i<100; i++)
      suma += A[i];
```

3-misol. 50 ta butun sondan iborat massiv berilgan. Massivning musbat elementlari yig’indisini toping.

```
//  massivning musbat elementlari yig’indisini toppish
int A[50];
int sum; // summani o‘z ichiga olgan o‘zgaruvchi
int i; // qo‘shimcha o‘zgaruvchi

// massivni kiritish
// ...

// Summani hisoblash
sum = 0; // yig’indini nolga tenglashtirish
for (i=0; i<50; i++)
   if (A[i]>0)
      sum = sum + A[i];
```

4-misol. 50 ta butun sonli massiv berilgan. Toq sonli massiv elementlarining ko‘paytmasini toping.

```
// massivning toq elementlarini ko‘paytirish
int A[50];
int d;   // ko‘paytmani o‘z ichiga olgan o‘zgaruvchi
int i;   //  yordamchi o‘zgaruvchi

//  massivni kiritish
// ...

// Ko‘paytmani hisoblash
   d = 1;    //  o‘zgaruvchining dastlabki o‘rnatilishi d
```

```
for (i=0; i<50; i++)
if ((A[i]%2)==1)
d = d * A[i];
```

5-misol. Ikki o‘lchovli massivni ishga tushirish.

```
// ikki o‘lchovli massivni ishga tushirish:
int a[5][3] = { {4, 7, 8}, {9, 66, -1}, {5, -5, 0}, {3, -3, 30}, {1, 1, 1} };
```

Nazorat savollari

1. Massivlar haqida tushuncha
2. Massiv elementi nima?
3. Massiv elementlarining indeksi
4. Massivlarni e’lon qilish
5. Massivning boshlang’ich qiymatlari
6. Massivni ishga tushirishning noto‘g’ri usullari qanday?
7. Satik va dinamik massivlar va ularning farqlari
8. Bir o‘lchamli massivning xotiradagi joylashuvi
9. Massiv elementlarini eng kattasini toppish
10. Massiv elementlarini juft elementlarini toppish

3.2. Ddinamik massivlar va ulardan foydalanish

1. Ko‘p o‘lchovli massivlarni aniqlash

C++ tilidagi ko‘p o‘lchovli massivlar 2 yoki undan ortiq indeksga ega bo‘lgan massivlardir. Ular massiv identifikatoridan keyin o‘zgarmas ifodalar ro‘yxati sifatida rasmiylashtiriladi. Bundan tashqari, har bir o‘zgarmas ifoda kvadrat qavs ichiga olingan. Kvadrat qavs ichidagi o‘zgarmas ifoda massivning berilgan o‘lchami bo‘yicha elementlar sonini aniqlaydi, shuning uchun ikki o‘lchovli massiv e’lonida ikkita o‘zgarmas ifoda, uch o‘lchovli massiv e’lonida uchta va hokazo.

Ko‘p o‘lchovli massivlarning oddiy ko‘rinishi bu **satr** va **ustunlardagi** ma’lumotlarni o‘z ichiga olgan **qiymatlar jadvallaridir**. Bitta jadval elementini aniqlash uchun ikkita indeks ko‘rsatilishi kerak: birinchisi **satr raqamini**, ikkinchisi **ustun raqamini** bildiradi.

Bitta elementni belgilash uchun ikkita indeks kerak bo‘lgan jadvallar yoki massivlar ikki o‘lchovli massivlar deb ataladi. C++ kompilyatorlari kamida 12 o‘lchovli massivlarni qo‘llab-quvvatlaydi.

Ko‘p o‘lchovli massivlarni bir o‘lchovli massivlar kabi ishga tushirish mumkin. **Masalan:**

```
int  a[2][3];            /* matritsa sifatida taqdim etiladi
                            a[0][0] a[0][1] a[0][2]
                            a[1][0] a[1][1] a[1][2] */
double b[10];        // double tipidagi 10 ta elementdan iborat vector
int w[3][3] = {       { 2, 3, 4 },
                      { 3, 4, 8 },
                      { 1, 0, 9 }
                  };
int ia[4][3] = { {0}, {3}, {6}, {9} }; // satrlarning birinchi elementlarini
ishga tushirish.
```

Figurali qavslar ichiga olingan ro‘yxatlar massiv satrlariga mos keladi. Qavslar bo‘lmasa, ishga tushirish ketma-ket elementlar tomonidan amalga oshiriladi, etishmayotgan elementlar bilvosita ishga tushiriladi. **ia** massivi uchun har bir satrning faqat birinchi elementlari ishga tushiriladi. **Qolgan elementlar nolga teng bo‘ladi**. Agar siz barcha massiv qiymatlarini nolga tenglashtirishingiz kerak bo‘lsa, siz quyidagilarni bajarishingiz mumkin:

long arr[2][3] = { 0 };

2. Massivni qiymatlar bilan to‘ldirish

C++ tilida massivlar elementining turiga cheklovlar qo`yilmaydi , lekin bu turlar chekli o`lchamdagi obyektlarning turi bo`lishi kerak.Chunki kompliyator massivning hotiradan qancha joy (bayt) egallashini xisoblay olish kerak. Xususan, massiv komponentasi massiv bo`lish mumkin (“vektorlar - vektori”) , natijada matritsa deb nomlanuvchi ikki o`lchamli massiv xosil bo`ladi. Agar matritsaning elementi xam vektor bo`lsa , uch o`lchamli massivlar - kub xosil bo`ladi. Shu yo`l bilan yechilayotgan masalaga bog`liq ravishda ixtiyoriy o`lchamdagi massivlarni yaratish mumkin. Ikki o`lchamli massivda birinchi indeks satrlar sonini , ikkinchisi esa ustunlar sonini bildiradi. Birinchi satrning dastlabki elementi **a10** – a biri nol element deb o`qiladi . a o`n deyilmaydi. **m** ta satr **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.

Ikki o`lchamli massivning sintaksi quyidagi ko`rinishda bo`ladi:

<tur><nom>[<uzunlik>][<uzunlik>]

Masalan, 10X20 o`lchamli xaqiqiy sonlar massivning e`loni:

Float a[10][20];

E`lon qilingan a matritsa ko`rinishi quyidagicha ko`rinishda bo`ladi.

Ikki o‘lchamli massiv e’loniga misol:

int a[3][3], b[2][4];
a matritsa a matritsa
a_{00} a_{01} a_{02} b_{00} b_{01} b_{02} b_{03}
a_{10} a_{11} a_{12} b_{10} b_{11} b_{12} b_{13}
a_{20} a_{21} a_{22}

Masalan:

1-misol. O‘lchami ko‘rsatilgan massivning barcha elementlariga boshlang’ich qiymat berish.

```
#include <iostream>
using namespace std;
int main(int argc, char *argv[]) {
     int k[5]={0};   // massivning barcha elementlariga 0 qiymat berish.
        for (int i=0; i<5; i++ )
      cout<<"k["<<i<<"]="<<k[i]<<endl;
    return 0;
}
```

Ekranga quyidagicha natija chiqadi:

```
C:\Users\Zahriddin\Desktop\dgdgf.exe
k[0]=0
k[1]=0
k[2]=0
k[3]=0
k[4]=0
Для продолжения нажмите любую клавишу . . .
```

2-misol. O‘lchami ko‘rsatilgan massivni to‘liq initsializatsiyalash.

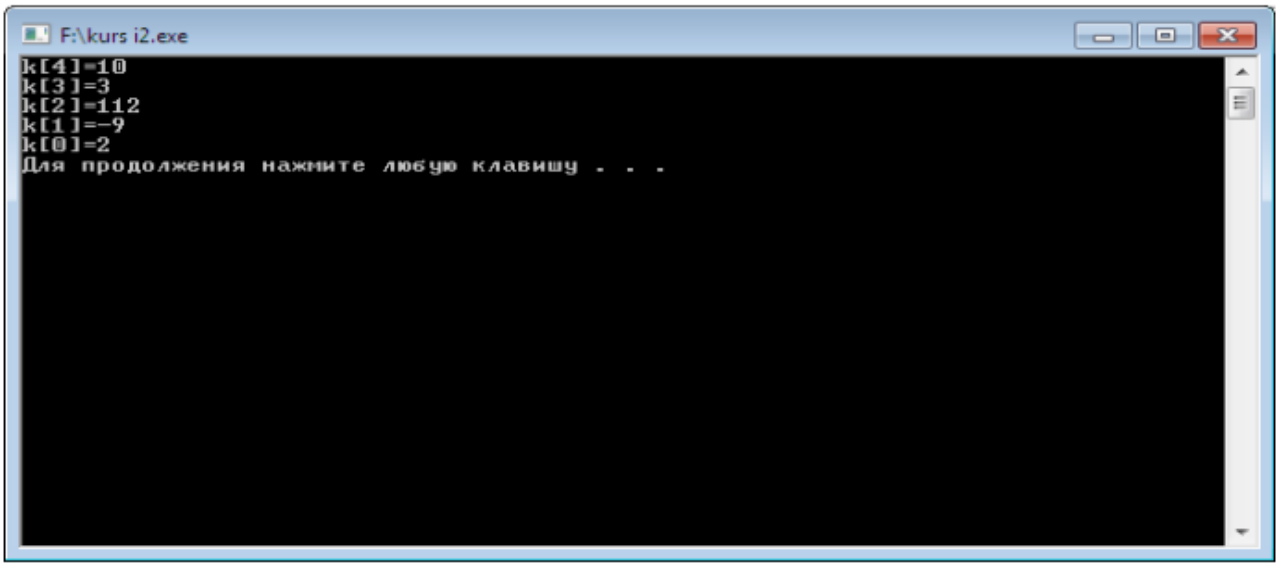
```
#include <iostream>
using namespace std;
int main(int argc, char *argv[]) {
    int k[5] = { 2, -9, 112, 3, 8 };

      for (int i=4;  i>=0;  i-- )  // indekslarini  teskari  tartibda  chop  etish.

       cout<<"k["<<i<<"]="<<k[i]<<endl;

    return 0;
}
```

Ekranga quyidagicha natija chiqadi:



Xulosa qilib, ko‘p o‘lchovli massivlar bilan ishlash misolini ko‘rib chiqamiz. Minimal elementni topish va uning joylashgan joyini eslab qolish kerak (ustun va qator raqami):

```
int  Matr[N][M];
int  iMin = Matr [0] [0];    // eng kichik element
int  iCol = 0;               // ustun
int  iRow = 0;               // satr
   for(int i = 0; i < N; i++)
       for(int j = 0; j < M; j++)
         {
           if(iMin > Matr[i][j])
             {
```

```
                iMin = Matr[i][j];
                    // eng kichik elementni toppish va uni joyini
to‘ldirish
                  iCol = i;
                  iRow = j;
                }
            }
```

3. DINAMIK MASSIVLAR

Massiv dinamik deb ataladi, uning o‘lchami dasturni bajarish jarayonida o‘zgarishi mumkin. O‘lchamni o‘zgartirish imkoniyati dinamik massivni statik massivdan ajratib turadi, uning o‘lchami dasturni kompilyatsiya qilish vaqtida o‘rnatiladi.

Dinamik massivni ajratish va u bilan ishlash uchun **new** va **delete** operatorlarining alohida shakllaridan foydalaniladi: **new[]** va **delete[].**

```
1  int *numbers = new int[4]; // 4 sondan
   iborat dinamik massiv
```

Bundan tashqari, bu holda, new operatori **int (**yaratilgan massivning birinchi elementi**)** tipidagi ob’ektga ko‘rsatgichni qaytaradi.

Ushbu holda, to‘rtta **int** elementli **massiv** aniqlanadi, lekin ularning har biri aniqlanmagan qiymatga ega. Biroq, biz qiymatlar bilan massivni ishga tushirishimiz ham mumkin:

```
1  int *n1 = new int[4];     // har bir element aniqlanmagan qiymatga ega
2  int *n2 = new int[4]();   // har bir elementni standart qiymati 0 ga ega
3  int *n3 = new int[4]{ 1, 2, 3, 4 };   // massiv 1, 2, 3, 4 raqamlaridan iborat
```

Ikkinchi holda, ma’lum qiymatlarga ega massivni ishga tushirishda shuni hisobga olish kerakki, agar figurali qavslarda massiv uzunligidan ko‘proq qiymatlar bo‘lsa, **new** operatori ishlamay qoladi va u **massiv** yarata olmaydi. Agar aksincha, kamroq qiymatlar o‘tgan bo‘lsa, unda hech qanday qiymat ko‘rsatilmagan elementlar standart qiymat bilan ishga tushiriladi.

Dinamik massivni yaratgandan so‘ng, biz qabul qilingan ko‘rsatgich yordamida u bilan ishlashimiz, uning elementlarini olishimiz va o‘zgartirishimiz mumkin:

```
1  int n = 5;  // massiv o‘lchami
2  int *p = new int[n]{ 1, 2, 3, 4, 5 };
```

```
3  for (int *q = p; q != p + n; q++)
4  {
5      std::cout << *q << "\t";
6  }
```

Dinamik massivni o‘chirish va uning xotirasini bo‘shatish uchun **delete** operatorining maxsus shakli qo‘llaniladi:

```
1  delete[] // dinamik massivga ko‘rsatgich
```

Masalan:

```
1   #include <iostream>
2
3   int main()
4   {
5       int n = 5;  //   massivning o‘lchami
6       int *p = new int[n]{ 1, 2, 3, 4, 5 }; // массив состоит из чисел 1, 2, 3, 4
7       for (int *q = p; q != p + n; q++)
8       {
9           std::cout << *q << "\t";
10      }
11
12      std::cout << std::endl;
```

Nazorat savollari

1. Ko‘p o‘lchovli massivlarni avtomatik aniqlash
2. Massiv elementlarini initializatsiya qilish
3. Massivni qiymatlar bilan to‘ldirish
4. Massiv qiymatlarini konsolga chop etish
5. Massivni xotiradagi joylashuvi
6. Dinamik massivlar va ularning berilishi
7. Dinamik massivlarda new va delete operatorlarini tavfsiflari

3.3. Satrlar va ular ustida amallar

1. Belgi va satrlar

Standart C++ tili ikki xildagi belgilar majmuasini qo‘llab-quvvatlaydi. Birinchi toifaga, an’anaviy, «tor» belgilar deb nomlanuvchi 8-bitli belgilar majmu asi kiradi, ikkinchisiga 16-bitli «keng» belgilar kiradi.Til kutubxonasida har bir guruh belgilari uchun maxsus funktsiyalar 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 0 kodli belgi bilan tugaydi (‘/’0’). Odatda, nol-terminator bilan tugaydigan satrlarni ASCIIZ-satrlar deyiladi. Quyidagi 3.1-jadvalda C++ tilida belgi sifatida ishlatilishi mumkin bo‘lgan o‘zgarmaslar to‘plami keltirilgan.

Belgilar sinflari	Belgi o‘zgarmaslar
Katta harflar	‘A’ …’Z’, ‘A’…’Ya’
Kichik harflar	a’ …’’z’, ‘a’…’ya’
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)	! ” # $ & ‘ ( ) * Q - , . / : ; < q > ? @ [g’] ^ _ { | } ~
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’

3.1-jadval. Belgi sifatida ishlatilishi mumkin bo‘lgan o‘zgarmaslar

Satr massivi e’lon qilinishida, satr oxiriga terminator qo‘yilishi va natijada satrga qo‘shimcha bitta bayt bo‘lishini inobatga olinishi kerak:

char satr[10];

Ushbu e’londa **satr** satri uchun jami 10 bayt ajratiladi – 9 ta satr hosil qiluv-chi belgilar uchun va 1 bayt terminator uchun. Satr o‘zgaruvchilar e’lon qilinishida boshlang’ich qiymatlarni qabul qilishi mumkin. Bu holda kompilyator avtomatik ravishda satr uzunligi hisoblaydi va satr oxiriga terminatorni qo‘shib qo‘yadi:

char Hafta_kuni[]=”Juma”;

Ushbu e’lon quyidagi e’lon bilan ekvivalent:

char Hafta_kuni[]={‘J’,’u’,’m’,’a’,’\’0’};

Satr qiymatini o‘qishda oqimli o‘qish operatori “>>” o‘rniga **getline()** funktsiyasini ishlatgan ma’qul hisoblanadi, chunki oqimli o‘qishda probellar inkor qilinadi (garchi ular satr belgisi hisoblansa ham) va o‘qilayotgan belgilar ketma-ketligi satrdan «oshib» ketganda ham belgilarni kiritish davom etishi mumkin. Natijada satr o‘ziga ajratilgan o‘lchamdan ortiq belgilarni «qabul» qiladi. Shu sababli, getline() funktsiyasi ikkita parametrga ega bo‘lib, birinchi parametr o‘qish amalga oshirilayotgan satrga ko‘rsatkich, ikkinchi parametrda esa o‘qilishi kerak bo‘lgan belgilar soni ko‘rsatiladi. Satrni **getline()** funktsiyasi orqali o‘qishga misol ko‘raylik:

```
#include <iostream.h>
    int main()
    {
     char satr[6];
     cout<<”Satrni kiriting: “<<‘\’n’;
     cin.getline(satr,6);
     cout<<”Siz kiritgan satr: “<<satr;
     return 0;
    }
```

Programmada ishlatilgan satr satri 5 ta belgini qabul qilishi mumkin ortiqcha lari tashlab yuboriladi. getline() funktsiyasiga murojaatda ikkinchi parametr qiymati o‘qilayotgan satr uzunligidan katta bo‘lmasligi kerak.

Satr bilan ishlaydigan funktsiyalarning aksariyati **«string.h»** kutubxonasi-da jamlangan. Nisbatan ko‘p ishlatiladigan funktsiyalarning tavsifini keltiramiz.

2. Satr uzunligini aniqlash funktsiyalari

Satrlar bilan ishlashda, aksariyat hollarda satr uzunligini bilish zarur bo‘ladi. Buning uchun **«string.h»** kutubxonasida **strlen()** funktsiyasi aniqlangan bo‘lib, uning sintaksisi quyidagicha bo‘ladi:

```
size_t strlen(const char* string)
```

Bu funktsiya uzunligi hisoblanishi kerak bo‘lgan satr boshiga ko‘rsatkich bo‘lgan yagona parametrga ega va u natija sifatida ishorasiz butun sonni qayta-radi. **strlen()** funktsiyasi satrning **real** uzunligidan bitta kam qiymat qaytaradi, ya’ni nol-terminator o‘rni hisobga olinmaydi.

Xuddi shu maqsadda **sizeof()** funktsiyasidan ham foydalanish mumkin va u **strlen()** funktsiyasidan 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()** funktsiyasidan **getline()** funktsiyasining ikkinchi argumenti sifati ishlatiladi va satr uzunligini yaqqol ko‘rsatmaslik imkonini beradi:

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

Masala. Faqat lotin harflaridan tashkil topgan satr berilgan.Undagi har xil harflar miqdori aniqlansin.

```
int main()
{
 const int n=80;
 char Satr[n];
 cout<<"Satrni kiriting:";
 cin.getline(Satr,sizeof(Satr));
 float s=0;
 int k;
 for(int i=0;i<strlen(Satr); i++)
  if(Satr[i]!=' ')
  {
   k=0;
for(int j=0;j<strlen(Satr); j++)
  if(Satr[i]==Satr[j]||abs(Satr[i]-Satr[j])==32)
      k=;
   s+=1./k;
  }
 cout<<"Satrdagi turli harflar miqdori:"<<(int)s;
  return 0;
```

```
}
```

Programmada satr uchun 80 uzunligidagi **Satr** belgilar massivi e'lon qilinga n va uning qiymati klaviaturadan kiritiladi. Masala quyidagicha yechiladi. Ichma-ich joylashgan takrorlash operatori yordamida **Satr** massivining har bir elementi - **Satr[i]** massivning barcha elementlari - **Satr[j]** bilan ustma-ust tushishi yoki ular birbiridan 32 soniga farq qilishi (katta va kichik lotin harflarining kodlari o'rtasidagi farq) holatlari k o'zgaruvchisida sanaladi va sumumiy yig'indiga **1/k** qiymati bilan qo'shiladi. Programma oxirida **s** qiymati butun turga aylantirilgan holda chop etiladi. Satrdagi so'zlarni bir-biridan ajratuvchi probel belgisi cheklab o'tiladi.

Programmaga

```
Satrdagi turli harflar miqdori
```

satri kiritilsa, ekranga javob tariqasida

```
Satrdagi turli belgilar miqdori:13
```

satri chop etiladi.

3. Satrlarni nusxalash va ulash

Satr qiymatini biridan ikkinchisiga nusxalash mumkin. Bu maqsadda bir qator standart funktsiyalar aniqlangan bo'lib, ularning ayrimlarining tavsiflarini keltiramiz.

```
strcpy() funktsiyasi prototipi
char* strcpy(char* str1, const char* str2)
```

ko'rinishga ega va bu funktsiya **str2** satrdagi belgilarni **str1** 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 (segmentda) **str1** satrdan keyin joylashgan berilganlar «ustiga» **str2** satrning «ortib qolgan» qismi yozilishi mumkin.

Navbatdagi programma qismi "Satrni nusxalash!" satrini **Str** satrga nusxalaydi:

```
char Str[20];

strcpy(Str, "Satrni nusxalash!");
```

Zarur bo'lganda satrning qaysidir joyidan boshlab, oxirigacha nusxalash 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!”,Str2[20];
 char* kursatkich=Str1;
 kursatkich+=7;
 strcpy(Str2,kursatkich);
 cout<<Str2<<endl;
 return 0;
}
```

strncpy() funktsiyasining **strcpy()** funktsiyasidan farqli joyi shundaki, unda bir satrdan ikkinchisiga nusxalanadigan belgilar soni ko‘rsatiladi. Uning prototipi 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()** funktsiyasi ishlatilishiga misol ko‘raylik:

```
#include <iostream.h>
#include <string.h>
int main()
{
 char Uzun_str[]="012345678901234567890";
 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, uning oldingi qiymatlari ustiga joylanadi va natijada ekranga

```
01234567890123456789
0123EF
```

satrlar chop etiladi.

strdup() funktsiyasiga yagona parametr sifatida satr manbaga ko‘rsatkich uzatiladi. Funktsiya, satrga mos xotiradan joy ajratadi, unga satrni nusxalaydi va yuzaga kelgan satr-nusxa adresini javob sifatida qaytaradi. **strdup()** funktsiya 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);
```

3.1. Satrlarni ulash

Satrlarni ulash (konkatenatsiya) amali yangi satrlarni hosil qilishda keng qo‘llaniladi. Bu maqsadda **«string.h»** kutubxonasida **strcat()** va **strncat()** funktsiyalari aniqlangan.

strcat() funktsiyasi sintaksisi quyidagi ko‘rinishga ega:

```
char* strcat(char* str1, const char* str2)
```

Funktsiya ishlashi natijasida **str2** satr, funktsiya qaytaruvchi satr - **str1** satr oxiriga ulanadi. Funktsiyani chaqirishdan oldin **str1** satr uzunligi, unga **str2** satri ulanishi uchun etarli bo‘lishi hisobga olingan bo‘lishi kerak.

Quyida keltirilgan amallar ketma-ketligining bajarilishi natijasida **satr** satriga qo‘shimcha satr ulanishi ko‘rsatilgan:

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

Амаллар кетма-кетлигини бажарилиши натижасида satr кўрсатаётган joyda “Bu satrga satr osti ulandi.” satri paydo bo‘ladi. **strncat()** funktsiyasi **strcat()** funktsiyadan farqli ravishda **str1** satrga **str2** satrning ko‘rsatilgan uzunlikdagi satr qismini ulaydi. Ulanadigan satr qismi uzunligi funktsiyaning uchinchi parametri sifatida beriladi.

Funktsiya sintaksisi:

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

Pastda keltirilgan programma bo‘lagida **str1** satrga **str2** satr-

ning boshlang’ich **10 ta** belgidan iborat satr qismini ulaydi:

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

Amallar bajarilishi natijasida ekranga

```
Programmalash tillariga misol bu-C++,Pascal
```

satri chop etiladi.

Nazorat savollari

1. C++ tilida belgi sifatida ishlatilishi mumkin bo‘lgan o‘zgarmaslar
2. Satr uzunligini aniqlash funktsiyalari
3. Satrlarni nusxalash va satrlarni ulash
4. Satr qismlarini izlash funktsiyalari
5. Satrni **getline()** funktsiyasi
6. **Strdup()** funktsiya sintaksisi
7. Satrlarni ulash funksiyalari

3.4. Strukturalar va birlashmalar

1. Strukturalar

Ma’lumki, biror predmet sohasidagi masalani yechishda undagi ob’ektlar bir nechta, har xil turdagi parametrlar bilan aniqlanishi mumkin. Masalan, tekislikdagi nuqta haqiqiy turdagi **x**-abtsissa va **y**-ordinata juftligi **-(x,y)** ko‘rinishida beriladi. Talaba haqidagi ma’lumotlar: satr turidagi talaba familiya, ismi va sharifi, mutaxassislik yo‘nalish, talaba yashash adresi, butun turdagi tug’ilgan yili, o‘quv bosqichi, haqiqiy turdagi reyting bali, mantiqiy turdagi talaba jinsi haqidagi ma’lumot va boshqalardan shakllanadi.

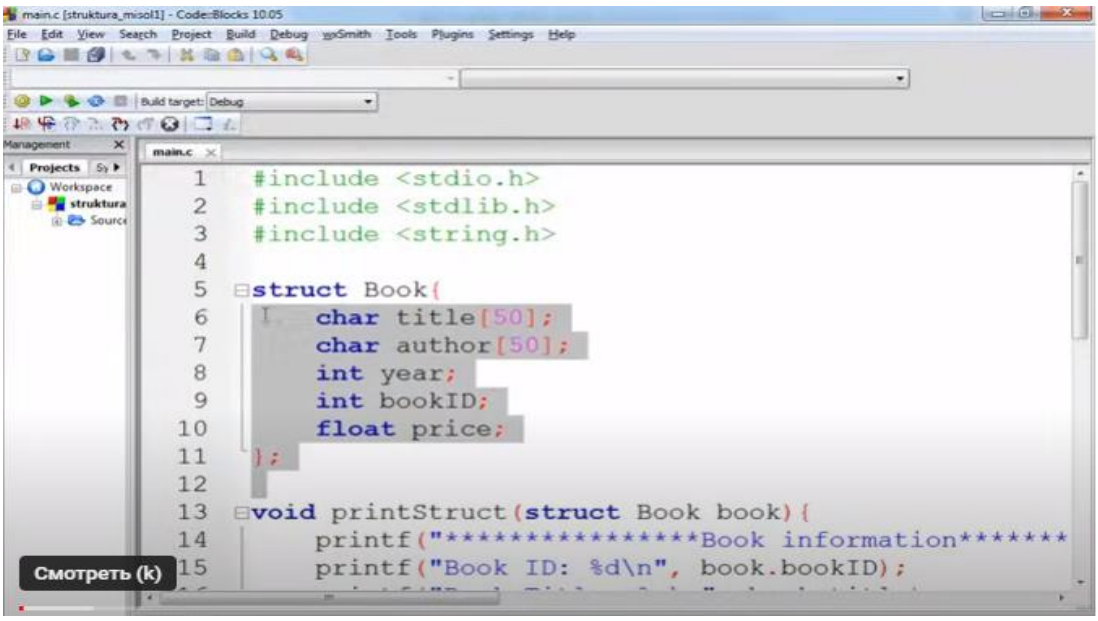
Programmada holat yoki tushunchani tavsiflovchi har bir berilganlar uchun alohida o‘zgaruvchi aniqlab masalani echish mumkin. Lekin bu holda ob’ekt haqidagi ma’lumotlar «tarqoq» bo‘ladi, ularni qayta ishlash murakkablashadi, ob’ekt haqidagi berilganlarni yaxlit holda ko‘rish qiyinlashadi.

C++ tilida bir yoki har xil turdagi berilganlarni jamlanmasi struktura deb nomlanadi. Struktura foydalanuvchi tomonidan aniqlangan berilganlarning yangi turi hisoblanadi (3.2-rasm).

Struktura quyidagicha aniqlanadi:

```
struct <структура номи>
{
   <tur1 > <nom1>;
   <tur2 > <nom2>;
  …
   <turn> <nomn>;
};

<tur1 > <nom1>;
   <tur2 > <nom2>;
  …
   <turn> <nomn>;
```



3.2-rasm. CodeBlocks kompilyatorida dastur kodi

Bu erda **<struktura nomi>** - struktura ko‘rinishida yaratilayotgan yangi turning nomi, **“<turi> <nomi>;”** - strukturaning **i**-maydonining (nomi) e’loni.

Boshqacha aytganda, struktura e’lon qilingan o‘zgaruvchilardan (maydonlardan) tashkil topadi. Unga har xil turdagi berilganlarni o‘z ichiga oluvchi qobiq deb qarash mumkin. Qobiqdagi berilganlarni yaxlit holda ko‘chirish, tashqi qurilmalar (binar fayllarga) yozish, o‘qish mumkin bo‘ladi.

Talaba haqidagi berilganlarni o‘z ichiga oluvchi struktura turining e’lon qilinishini ko‘raylik.

```
struct Talaba
 {
    char FISh[30];
    unsigned int Tug_yil;
    unsigned int Kurs;
    char Yunalish[50];
    float Reyting;
    unsigned char Jinsi[5];
    char Manzil[50];
    bool status;
 };
```

Programmada strukturalardan foydalanish, shu turdagi o‘zgaruvchilar e’lon qilish va ularni qayta ishlash orqali amalga oshiriladi:

Talaba talaba;

Struktura turini e’lonida turning nomi bo‘lmasligi mumkin, lekin bu holda struktura aniqlanishidan keyin albatta o‘zgaruvchilar nomlari yozilishi kerak:

```
struct
 {
  unsigned int x,y;
  unsigned char Rang;
 } Nuqta1, Nuqta2;
```

Keltirilgan misolda struktura turidagi ***Nuqta1, Nuqta2*** o‘zgaruvchilari e’lon qilingan. Struktura turidagi o‘zgaruvchilar bilan ishlash, uning maydonlari bilan ishlashni anglatadi.

Struktura maydoniga murojaat qilish **‘.’** (nuqta) orqali amalga oshiriladi. Bunda struktura turidagi o‘zgaruvchi nomi, undan keyin nuqta qo‘yiladi va maydon o‘zgaruvchisining nomi yoziladi. Masalan, talaba haqidagi struktura maydonlariga murojaat quyidagicha bo‘ladi:

```
talaba.Kurs=2;
talaba.Tug_yil=1988;
strcpy(talaba.FISh,”Abdullaev A.A.”);
strcpy(talaba.Yunalish,
”Informatika va Axborot texnologiyalari”);
strcpy(talaba.Jinsi,”Erk”);
strcpy(talaba.Manzil,
”Toshkent,Yunusobod 6-3-8, tel: 224-45-78”);
talaba.Reyting=123.52;
```

Keltirilgan misolda talaba strukturasining son turidagi maydonlariga oddiy ko‘rinishda qiymatlar berilgan, satr turidagi maydonlar uchun strcpy funktsiyasi orqali qiymat berish amalga oshirilgan.

Struktura turidagi ob’ektning xotiradan qancha joy egallaganligini **sizeof** funktsiyasi (operatori) orqali aniqlash mumkin:

```
int i=sizeof(Talaba);
```

Ayrim hollarda struktura maydonlari o‘lchamini bitlarda aniqlash orqali egallanadigan xotirani kamaytirish mumkin. Buning uchun struktura maydoni quyidagicha e’lon qilinadi:

<maydon nomi> : <o‘zgarmas ifoda>

Bu erda <maydon nomi>- maydon turi va nomi, <o‘zgarmas ifoda>- maydonning bitlardagi uzunligi. Maydon turi butun turlar bo‘lishi kerak (int, long, unsigned, char).

Agar foydalanuvchi strukturaning maydoni faqat 0 va 1 qiymatini qabul qilishini bilsa, bu maydon uchun bir bit joy ajratishi mumkin (bir bayt yoki ikki bayt o‘rniga). Xotirani tejash evaziga maydon ustida amal bajarishda razryadli arifmetikani qo‘llash zarur bo‘ladi.

Misol uchun sana-vaqt bilan bog’liq strukturani yaratishning ikkita variantini ko‘raylik. Struktura yil, oy, kun, soat, minut va sekund maydonlaridan iborat bo‘lsin va uni quyidagicha aniqlash mumkin:

```
struct Sana_vaqt
{
 unsigned short Yil;
 unsigned short Oy;
 unsigned short Kun;
 unsigned short Soat;
 unsigned short Minut;
 unsigned short Sekund;
};
```

Bunday aniqlashda Sana_vaqt strukturasi xotirada 6 maydon*2 bayt=12 bayt joy egallaydi. Agar e’tibor berilsa strukturada ortiqcha joy egallangan holatlar mavjud. Masalan, yil uchun qiymati 0 sonidan 99 sonigacha qiymat bilan aniqlanishi etarli (masalan, 2008 yilni 8 qiymati bilan ifodalash mumkin). Shuning uchun unga 2 bayt emas, balki 7 bit ajratish etarli. Xuddi shunday oy uchun 1..12 qiymatlarini ifodalashga 4 bit joy etarli va hakoza.

Yuqorida keltirilgan cheklovlardan keyin sana-vaqt strukturasini tejamli variantini aniqlash mumkin:

```
struct Sana_vaqt2
{
 unsigned Yil:7;
 unsigned Oy:4;
 unsigned Kun:5;
 unsigned Soat:6;
 unsigned Minut:6;
 unsigned Sekund:6;
};
```

Bu struktura xotiradan 5 bayt joy egallaydi.

2. Struktura funktsiya argumenti sifatida

Strukturalar funktsiya argumenti sifatida ishlatilishi mumkin. Buning uchun funktsiya prototipida struktura turi ko‘rsatilishi kerak bo‘ladi. Masalan, talaba haqidagi berilganlarni o‘z ichiga oluvchi **Talaba** strukturasi turidagi berilganlarni **Talaba_Manzili()** funktsiyasiga parametr sifatida berish uchun funktsiya prototipi quyidagi ko‘rinishda bo‘lishi kerak:

void Talaba_Manzili(Talaba);

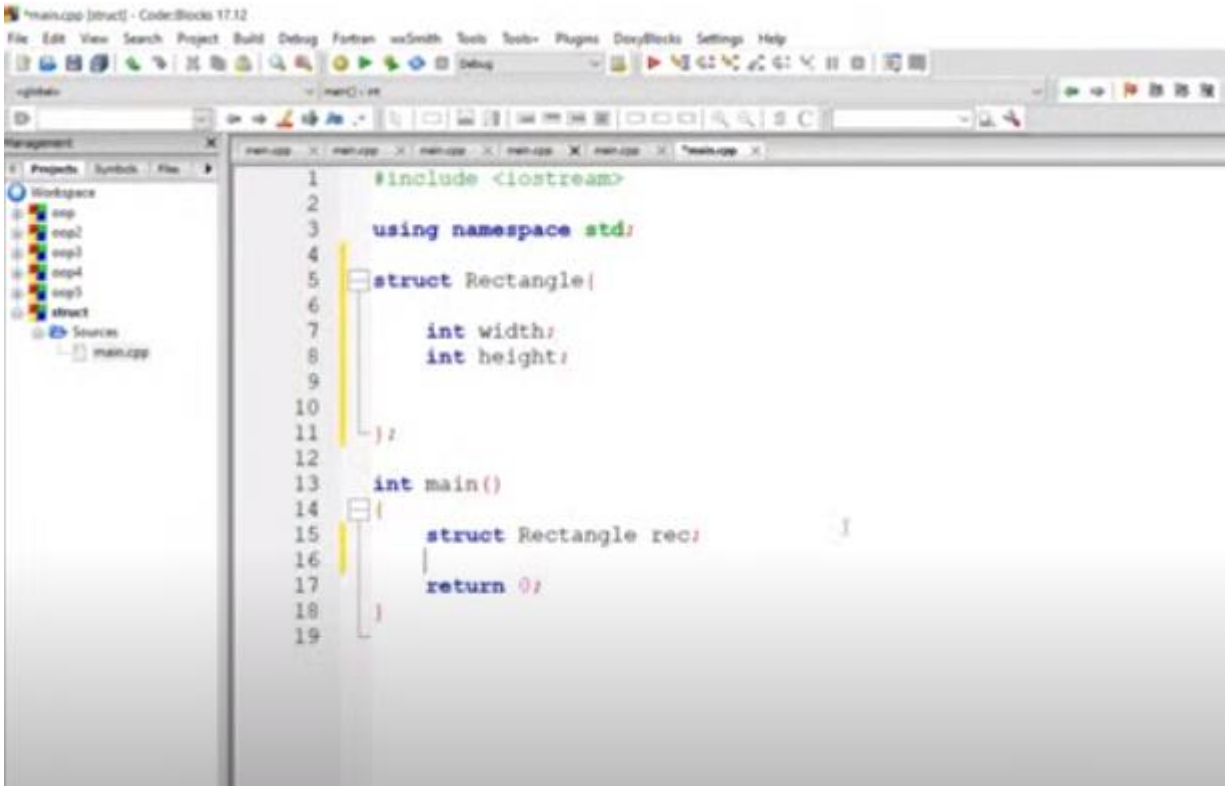
Funktsiyaga strukturani argument sifatida uzatishga misol sifatidagi programmaning matni:

```
#include <iostream.h>
#include <string.h>
struct Talaba
{
 char FISh[30];
 unsigned int Tug_yil;
 unsigned int Kurs;
 char Yunalish[50];
 float Reyting;
 unsigned char Jinsi[5];
 char Manzil[50];
 bool status;
};
void Talaba_Manzili(Talaba);
int main(int argc,char* argv[])
{
 Talaba talaba;
 talaba.Kurs=2;
talaba.Tug_yil=1988;
 strcpy(talaba.FISh,"Abdullaev A.A.");
 strcpy(talaba.Yunalish,
 "Informatika va Axborot texnologiyalari");
 strcpy(talaba.Jinsi,"Erk");
 strcpy(talaba.Manzil,
 "Toshkent, Yunusobod 6-3-8, tel: 224-45-78");
 talaba.Reyting=123.52;
 Talaba_Manzili(talaba);
return 0;
```

```
}
void Talaba_Manzili(Talaba t)
{
 cout<<"Talaba FIO: "<<t.FIO<<endl;
 cout<<"Manzili: "<<t.Manzil<<endl;
}
```

Programma bosh funktsiyasida talaba strukturasi aniqlanib, uning maydonlariga qiymatlar beriladi. Keyin **talaba** strukturasi **Talaba_Manzili()** funktsiyasiga argument sifatida uzatiladi. Programma ishlashi natijasida ekranga quyidagi ma’lumotlar chop etiladi (3.3-rasm).

```
Talaba FIO: Abdullaev A.A.
Manzili: Toshkent, Yunusobod 6-3-8, tel: 224-45-78
```



3.3-rasm. CodeBlocks kompilyatorida dastur kodi

3. Strukturalar massivi

O‘z-o‘zidan ma’lumki, struktura turidagi yagona berilgan bilan echish mumkin bo‘lgan masalalar doirasi juda tor va aksariyat holatlarda, qo‘yilgan masala strukturalar majmuasini ishlatishni talab qiladi. Bu turdagi masalalarga berilganlar bazasini qayta ishlash masalalari deb qarash mumkin.

Strukturalar massivini e’lon qilish xuddi standart massivlarni e’lon qilishdek, farqi massiv turi o‘rnida foydalanuvchi tomonidan aniqlangan struktura turining nomi yoziladi. Masalan, talabalar haqidagi berilganlarni o‘z ichiga olgan massiv yaratish e’loni quyidagicha bo‘ladi:

```
const int n=25;
Talaba talabalar[n];
```

Strukturalar massivining elementlariga murojaat odatdagi massiv elementlariga murojaat usullari orqali, har bir elementning maydonlariga murojaat esa **‘.’** orqali amalga oshiriladi.

Quyidagi programmada guruhidagi har bir talaba haqidagi berilganlarni klaviaturadan kiritish va guruh talabalarini familiya, ismi va sharifini chop qilinadi.

```
#include <iostream.h>
#include <conio.h>
const int n=3;
struct Talaba
{
   char FISh[30];
   unsigned int Tug_yil;
   unsigned int Kurs;
   char Yunalish[50];
   float Reyting;
   char Jinsi[6];
   char Manzil[50];
   bool status;
};
   void Talaba_Kiritish(Talaba t[]);
   void Talabalar_FISh(Talaba t[]);
   int main(int argc, char* argv[])
{
   Talaba talabalar[n];
   Talaba_Kiritish(talabalar);
   Talabalar_FISh(talabalar);
   return 0;
}
   void Talabalar_FISh(Talaba t[])
{
    for(int i=0; i<n; i++)
```

```
        cout<<t[i].FISh<<endl;
    }
      void Talaba_Kiritish(Talaba t[])
    {
       for(int i=0; i<n; i++)
     {
      cout<<i+1<<"-talaba malumotlarinkiriting:"<<endl;
      cout<<" Talaba FISh :";
      cin.getline(t[i].FISh,30);
      cout<<" Kurs:";
      cin>>t[i].Kurs;
      cout<<" Reyting bali:";
      cin>>t[i].Reyting;cout<<" Tug''ilgan yili:";
      cin>>t[i].Tug_yil;
      cout<<" Ta'lim yo'nalishi:";
      cin.getline(t[i].Yunalish,50);
     cout<<" Jinsi(erkak,ayol):";
      cin.getline(t[i].Jinsi,6);
      cout<<" Yashash manzili:";
      cin.getline(t[i].Manzil,50);
    }
}
```

4. Strukturalarga ko‘rsatkich

Struktura elementlariga ko‘rsatkichlar orqali murojaat qilish mumkin. Buning uchun strukturaga ko‘rsatkich o‘zgaruvchisi e’lon qilinishi kerak. Masalan, yuqorida keltirilgan misolda **Talaba** strukturasiga ko‘rsatkich quyidagicha e’lon qilinadi:

```
Talaba * k_talaba;
```

Ko‘rsatkich orqali aniqlangan struktura elementlariga murojaat **«.»** bilan emas, balki «->» vositasida amalga oshiriladi:

```
cout<<k_talaba ->FISh;
```

Nazorat savollari

1. C++ tilida strukturalar haqida tushunchalar
2. Strukturalar qanday aniqlanadi?
3. Sizeof funktsiyasi (operatori) ni tavsifi
4. Sana-vaqt bilan bog'liq strukturani dastur kodi
5. Struktura funktsiya argumenti sifatida qanday aniqlanadi?
6. Strukturalar massivi va ularning tavsifi

3.5. Fayllar va ular bilan ishlash

1. C++ da fayllar

Ko‘pgina kompyuter dasturlari fayllar bilan ishlaydi va shuning uchun fayllarni ***yaratish, o‘chirish, yozish, o‘qish, ochish*** zarurati tug'iladi.

Fayl nima? Fayl - ***bu ba'zi xotira qurilmalarida saqlanishi mumkin bo‘lgan baytlar to‘plami.*** Xo‘sh, endi ma'lum bo‘ldiki, fayl o‘ziga xos nomga ega bo‘lgan baytlar ketma-ketligi, masalan, ***.txt fayli***. Xuddi shu nomdagi fayllar bitta katalogda bo‘lishi mumkin emas. Fayl nomi nafaqat uning nomi, balki kengaytma sifatida ham tushuniladi, masalan: ***file.txt*** va ***file.dat*** bir xil nomga ega bo‘lsada, har xil fayllardir.

Fayllarning to‘liq nomi kabi narsa bor - bu fayl nomi bilan fayl katalogining to‘liq manzili, masalan: ***D:\docs\file.txt.*** Ushbu asosiy tushunchalarni tushunish muhim, aks holda fayllar bilan ishlash qiyin bo‘ladi.

Fayllar bilan ishlash uchun siz <***fstream***> sarlavha faylini kiritishingiz kerak. <***fstream***> bir nechta sinflarni belgilaydi va sarlavhali fayllarni o‘z ichiga oladi <***ifstream***> - fayl kiritish va <***ofstream***> - fayl chiqishi.

Faylli kiritish-chiqarish standart kiritish-chiqarishga o‘xshaydi, yagona farq shundaki, kiritish-chiqarish ekranga emas, balki **faylga** amalga oshiriladi. Agar standart qurilmalarga kiritish/chiqarish ***cin*** va ***cout*** obyektlari yordamida amalga oshirilsa, faylli kiritish/chiqarishni tartibga solish uchun ***cin*** va ***cout*** operatorlariga o‘xshash foydalanish mumkin bo‘lgan o‘z obyektlaringizni yaratish kifoya.

Masalan, matnli fayl yaratish va unga ***C++ da fayllar bilan ishlash*** qatorini yozish kerak. Buning uchun siz quyidagi amallarni bajarishingiz kerak:

1. **Ofstream** sinfining ob'ektini yaratish;
2. Sinf ob'ektini yoziladigan fayl bilan bog'lash;
3. Faylga qator yozish;
4. Faylni yoping.

Nima uchun **ifstream** sinfini emas, balki **ofstream** sinfining ob’ektini yaratish kerak? Chunki siz faylga yozishingiz kerak va agar fayldan ma’lumotlarni o‘qish kerak bo‘lsa, u holda **ifstream** sinfining ob’ekti yaratiladi.

```
1  // faylga yozish uchun ob’ekt yaratamiz
2  ofstream /*ob’ektni nomi*/; //ofstream sinfining ob’ekti
```

Ob’ektni - **fout** deb ataymiz, bu erda nima sodir bo‘ladi:

```
1  ofstream fout;
```

Nima uchun bizga ob’ekt kerak? Ob’ekt faylga yozish imkoniyatiga ega bo‘lishi uchun talab qilinadi. Ob’ekt allaqachon yaratilgan, lekin satr yozilishi kerak bo‘lgan fayl bilan bog’lanmagan.

```
1  fout.open("cppstudio.txt"); // ob’ektni fayl bilan bog’lash
```

Nuqta operatsiyasi orqali biz qavs ichida fayl nomini ko‘rsatgan ***open()*** klass usuliga kirishga erishamiz. Belgilangan fayl joriy katalogda dastur bilan yaratiladi. Agar bir xil nomdagi fayl mavjud bo‘lsa, mavjud fayl yangisi bilan almashtiriladi. Shunday qilib, ***fayl ochiq***, unga kerakli qatorni yozish qoladi. Bu shunday amalga oshiriladi:

```
1  fout <<"C++ da fayllar bilan ishlash";//qatorlarni faylga yozish
```

Fout ob’ekti bilan birgalikda oqimga uzatish operasiyasidan foydalanib C++ da faylga satr yoziladi. Endi fayl tarkibini o‘zgartirish shart emasligi sababli, u yopiq bo‘lishi kerak, ya’ni ob’ekt fayldan ajratilishi kerak.

```
1  fout.close(); //faylni yoping
```

Natijada C++ da fayllar bilan ishlash qatorli fayl hosil bo‘ladi.

1 va 2-bosqichlarni birlashtirish mumkin, ya’ni bir qatorda ob’ektni yaratish va uni fayl bilan bog’lash. Bu shunday amalga oshiriladi:

```
1   ofstream fout("cppstudio.txt"); //ofstream sinfining
    ob'ektini yaratamiz va uni cppstudio.txt fayli bilan bog'laymiz
```

Barcha kodlarni birlashtiramiz va quyidagi dasturni olamiz:

```
1   // file.cpp: konsol ilovasi uchun kirish nuqtasi belgilaydi.
2    #include "stdafx.h"
3   #include <fstream>
4   using namespace std;
5
6   int main(int argc, char* argv[])
7   {
8       ofstream fout("cppstudio.txt"); // biz yozish uchun
9   ofstream sinfining ob'ektini yaratamiz va uni cppstudio.txt fayli bilan bog'laymiz.
10      fout << "Работа с файлами в C++"; // faylga qator yozish
11      fout.close(); // faylni yopamiz
12      system("pause");
13      return 0;
14  }
```

C++ tilidagi standart va foydalanuvchi tomonidan aniqlangan turlarning muhim xususiyati shundan iboratki, ularning oldindan aniqlangan miqdordagi chekli elementlardan iboratligidir.

Agar programma yangidan ishga tushirilsa, bu berilganlarni yangidan hosil qilish zarur bo‘ladi. Aksariyat tadbiqiy masalalar esa berilganlarni doimiy ravishda saqlab turishni talab qiladi. Masalan, korxona xodimlarinning oylik maoshini hisoblovchi programmada xodimlar ro‘yxatini, shtat stavkalari va xodimlar tomonidan olingan maoshlar haqidagi ma’lumotlarni doimiy ravishda saqlab turish zarur. Bu talablarga fayl turidagi ob’ektlar (o‘zgaruvchilar) javob beradi.

Fayl - bu bir xil turdagi qiymatlar joylashgan tashqi xotiradagi nomlangan sohadir.

Yana bir muhim tushunchalardan biri fayl ko‘rsatkichi tushunchasidir. Fayl ko‘rsatkichi - ayni paytda fayldan o‘qilayotgan yoki unga yozilayotgan joy (yozuv o‘rnini) ko‘rsatib turadi, ya’ni fayl ko‘rsatkichi ko‘rsatib turgan joydan bitta yozuvni o‘qish yoki shu joyga yangi yozuvni joylashtirish mumkin.

2. Matn va binar fayllar

C++ tili C tilidan o‘qish-yozish amalini bajaruvchi standart funktsiyalar kutubxonasini vorislik bo‘yicha olgan. Bu funktsiyalar **<stdio.h>** sarlavha faylida e’lon qilingan. O‘qish-yozish amallari fayllar bilan bajariladi. Fayl matn yoki binar (ikkilik) bo‘lishi mumkin.

Binar fayllar - bu oddiygina baytlar ketma-ketligi. Odatda binar fayllardan berilganlarni foydalanuvchi tomonidan bevosita «ko‘rish» zarur bo‘lmagan hollarda ishlatiladi. Binar fayllardan o‘qish-yozishda baytlar ustida hech qanday konvertatsiya amallari bajarilmaydi.

C++ dasturlash tili nafaqat boshqa dasturlash tillarida ham fayllar bilan ishlash juda katta ahamiyatga ega hisoblanadi. C++ dasturlash tilida **fstream** standart kutubxonadan foydalaniladi. fstream dan foydalanish uchun **<iostream>** va **<fstream>** standart kutubxonalardan foydalaniladi.

```
#include <iostream>
#include <fstream>
```

fstream standart kutubxonasi ichida 3 ta obyekt mavjud:

Object/Data Type	Tavsifi
ofstream	Faylni yaratish va yozish uchun
ifstream	Faylni o‘qish uchun
fstream	Ofstream va ifstream larbirgalikda. O‘qish, yozish va yaratish imkonoyatini beradi.

3. Faylni yaratish va yozish

Fayl yaratish uchun, ofstream yoki fstream ob’ektdan foydalaniladi va fayl nomi ko‘rsatiladi. Faylga yozish uchun kiritish operatoridan (<<) foydalaniladi.

1-misol.

```
#include <iostream>
#include <fstream>
using namespace std;

int main()
{
```

```
    ofstream MyFile("filename.txt");// Faylni
  yaratadi
     yoki ochadi.
    MyFile << "Hello World.MasterSherkulov.Uz";
    // Faylga yozadi.
    MyFile.close();// Faylni yopadi.
}
```

2-misol.

```
1  #include <iostream>
2  #include <fstream>
3  using namespace std;
4  int main()
5  {
6      ofstream out; //yozish uchun oqim
7      out.open("D:\\hello.txt");//yozish uchun fayilni
   ochamiz
8      if (out.is_open())
9      {
10      cout <<"Hello World!"<<endl;
11     }
12
13     std::cout<<"End of program"<<endl;
14     return 0;
15 }
```

Ushbu usul faylni qayta yozadi. Agar fayl oxiriga matn qo‘shishingiz kerak bo‘lsa, faylni ochish uchun iOS::app rejimidan foydalanishingiz kerak:

```
1  std::ofstream out("D:\\hello.txt", std::ios::app);
2  if (out.is_open())
3  {
4      out << "Welcome to CPP" << std::endl;
5  }
6  out.close();
```

Nima uchun faylni yopamiz?

Bu yaxshi amaliyot deb hisoblanadi va keraksiz xotira joyini tozalaydi.

4. Faylni o‘qish

Fayldan o‘qish uchun, ***ifstream*** yoki ***fstream*** ob’ektdan va fayl nomidan foydalaning. E’tibor bering, biz funktsiyani (ob’ektga tegishli) funktsiya **while** bilan bir qatorda fayl satrini o‘qish va fayl tarkibini chop etish uchun ishlatamiz.

```
#include <iostream>
#include <fstream>
#include <string>
 using namespace std;

int main ()
{
 ofstream MyWriteFile("filename.txt");//Faylni
    yaratish
 MyWriteFile <<"Hello World. uzbekdevs.uz";

 MyWriteFile.close();//Faylni yopish

 string myText; //String tipiga tegishli
o'zgaruvchi
                  yaratish
 ifstream MyReadFile("filename.txt");//Text faylni
 o'qish

 //getline() funksiyasidan foydalanib faylni o'qish

  while (getline (MyReadFile, myText))
  {
   cout << myText;//O'qilgan faylni qora ekranga
                  chiqarish
  }

   MyReadFile.close();//Faylni yopish
```

```
}
```

Ekranda quyidagi matn hosil bo‘ladi:

Hello World. uzbekdevs.uz

Nazorat savollari

1. C++ da fayllar
2. Faylni o‘qish - yozish funktsiyalari
3. O‘qish-yozish oqimlari. Standart oqimlar
4. Matn va binar fayllar
5. Faylni yaratish va yozish
6. Faylni o‘qish
7. Belgilarni o‘qish-yozish funktsiyalari
8. Satrlarni o‘qish - yozish funktsiyalari

IV-BOB. BORLAND C++ BUILDER DASTURLASH MUHITIGA KIRISH

4.1. Borland C++ Builder dasturlash muhiti

Ixtiyoriy dasturlashdan minimal bilimga (dasturlashdan maktab kursidan) ega foydalanuvchi ham **Borland C++ Builder 6** muhitni tezda o‘rgana oladi. Bu muhitning soddaligi professional dasturchi bo‘lishga xalaqit qilmaydi.

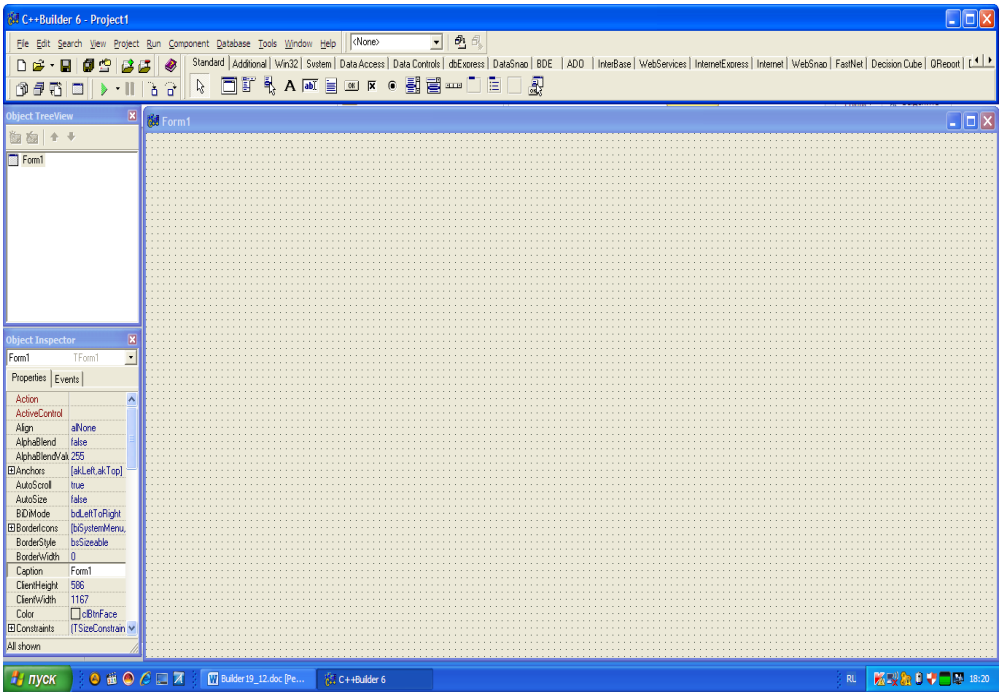
Dasturlash muhitlari ichida eng sodda muhitlardan biri hisoblanadi, bir nechta mashqlarni mustaqil bajargandan keyin, muhitdan kishining ajralgisi kelmaydi.

Umuman har qanday yangi texnologiyani o‘zlashtirayotganda bilgan o‘rgangan bilimlarni amalda tez-tez takrorlab turmasa xotirada saqlab qolishning iloji bo‘lmaydi. Yangi tushunchani takrorlab xotiraga joylash va uni tadbiq qilishni bilgandan keyingina xotirada mustahkam joylashadi. Har bir dasturlash tili, muhiti o‘zidan oldingi tillarning mukammal tomonlarini o‘zida mujassamlashtiradi, uncha muvaffaqiyat qozonmagan tomonlarini mukammallashtirish hisobiga rivojlantiriladi. Shunga ko‘ra bitta zamonaviy tilni mukammal bilgan dasturchi ikkinchi tilni katta kuch va vaqt sarflamasdan o‘rgana oladi. Dasturchilar tillarni o‘rganayotganda birini ikkinchisi bilan solishtirib o‘rganishadi. Masalan, Obyektga Yo‘naltirilgan Dasturlash (OYD) zamonaviy dasturlash tillarining asosini tashkil qiladi. Demak barcha zamonaviy dasturlash tillari umumiy qoidalarga bo‘ysunadi.

OYDda har bir dasturlash elementi obyekt sifatida qaraladi. Bunda obyekt, barcha hollarda ma’lum umumiy qoidalarga va xususiy qoidalarga ega bo‘ladi. Bu obyektlar oynalar, tugmalar, konteynerlar va Canvas lar ko‘rinishida bo‘lishi mumkin. Bunda har bir ilova kichik qismlarga ajratiladi va yakunida bu qismlar birlashtiriladi. Borland C++ Builder 6 muhitida Windows osti ilovalar yaratish juda oson. **Borland C++ Builder 6** interfeysi 4.1-rasmda keltirilgan.

OYD da qadamba qadam alohida uncha katta bo‘lmagan dasturlarni funktsiya metodlari amalga oshirsa, harakatlanuvchi jarayonlarni qayta ishlash uchun xodisa amalidan, chaqiriluvchi obyektlarni tugmalar va oynalar bilan ifodalaydi.

Dasturchilar tilida bu interfeysga tez qayta ishlovchi muhit RAD (Rapid Application Development) deb atashadi. Bunga sabab bu muhitda dastur, ilova tuzish va uning dizaynini qurishda tayyor obyektlar va kutubxonada mavjud metodlardan foydalanish mumkin.



4.1-rasm. Borland C++ Builder 6 interfeysi

Masalan, kompyuter avtomatik ravishda xodisani qayta ishlovchi funktsiya dasturi matnini hosil qiladi. Interfeysning tepasi 4.2-rasmda C++ Builder 6 – **Project1** asosiy oynasi joylashgan:

4.2-rasm. Interfeysning asosiy oynasi

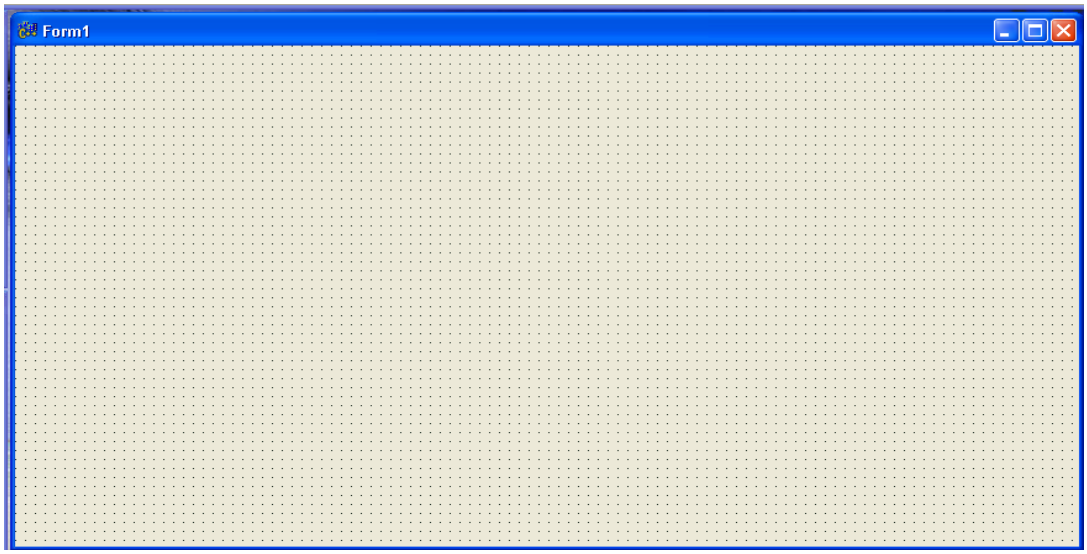
Ilova – bu tayyor bajariluvchi fayl hosil qilish uchun kerak bo‘ladigan barcha fayllar to‘plamidir. Masalan, ilova tarkibiga dastur matni, tovush fayllari, ikonka rasmlari va shu kabi ilovaga, kerakli fayllar kirishi mumkin. Bunda har bir ilova uchun alohida papka hosil qilish maqsadga muvofiq sanaladi. Chunki ilovani boshqa kompyuterga o‘tkazmoqchi bo‘lsak va papkada saqlamagan bo‘lsak, ularni yig’ishimizga to‘g’ri keladi. Bunda interfeysning o‘zi ilovani saqlashni talab qiladi, ammo papka hosil qilishni o‘zimiz bilishimiz kerak.

Interfeysning asosiy oynasi sarlavha oynasida ilova nomi, ilovani tiklash, berkitish tugmalari joylashgan. Sarlavha nomi tagida asosiy menyu joylashgan. Bu menyu orqali muhitning barcha funktsiya va komandalarini ishga tushirish mumkin. Asosiy menyuning tagida tez tugmalar joylashgan. Bu tugmalar ma’nolariga ko‘ra guruhlarga ajratilgan. Bu tugmalar orqali tez-tez ishlatiladigan komandalarni ishga tushirishimiz mumkin. Bu tugmalarning o‘ng tomonida vizual komponentalar VCL (Visual Component Library - vizual komponentlar kutubxonasi) palitrasi joylashgan. Bu shunday obyektlarki yoki shunday dasturlash komponentalariki,

bular yordamida Windows uchun vizual dasturlarni tezda yaratish mumkin. Komponentalar yordamida har xil tugmalar, rasmlar, yozuvlar, taymerlar, kalendar va hokazolarni ilovaga kiritishimiz mumkin. Vizual komponentalar palitrasi ma’nosiga va vazifasiga ko‘ra guruhlarga ajratilgan.

Bu muhitning barcha oynalarini berkitish mumkin faqat asosiy oynani berkitsa ilovadan chiqib ketiladi, boshqa barcha oynalar o‘lchamlarini kattalashtirish va kichiklashtirish imkoniyatiga egamiz.

Vizual komponentalar palitrasi. Vizual komponentalar palitrasini bir qismi monitorda ko‘rinib turadi, qolganlarini o‘ng va chapga siljituvchi tugmachalar vositasida ko‘rishimiz mumkin. Ekran (monitor) markazida forma (shakl) dizayneri joylashgan (4.3-rasm).

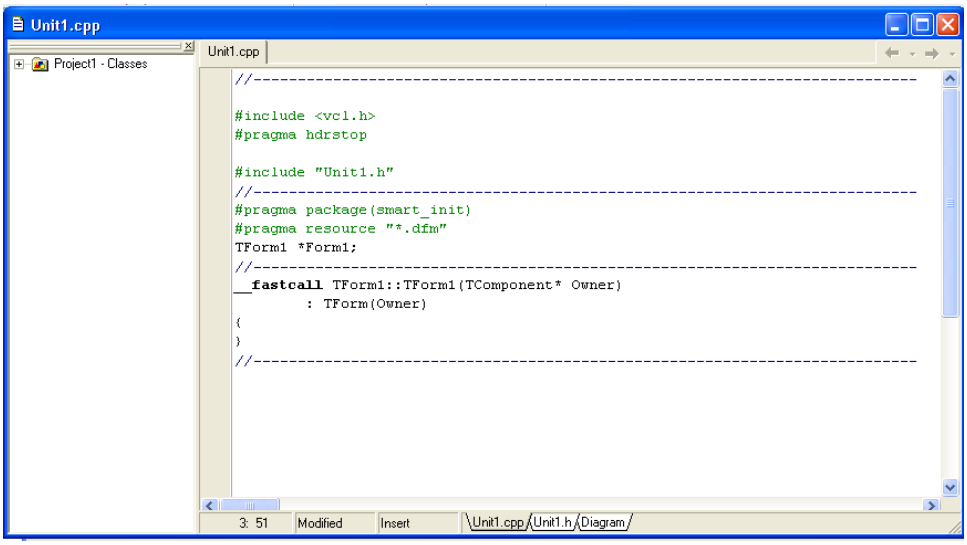


4.3-rasm. Forma (shakl) dizayneri

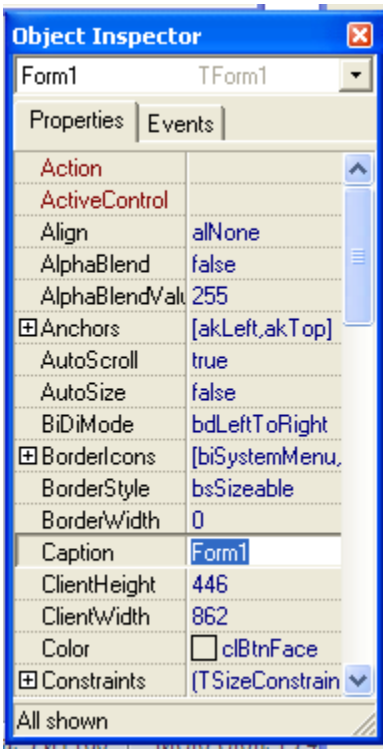
Bu bo‘lg’usi dasturning interfeysini hosil qiluvchi oynadir. Oynaning nomi va sarlavhasi uning tepasida yozilgan bo‘ladi. Odatda, Form l (Form 2, Form 3, Form 4) va shuningdek oynani berkituvchi va kichraytiruvchi tugmachalar ekranning o‘ng tepasida joylashgan bo‘ladi. Oynaning sathiga vizual dastur uchun zarur bo‘ladigan VCL komponentalar joylashtiriladi.

Forma (shakl) dizayneri ostida dastur kodi muharriri joylashgan, odatda u **Unit1.cpp** sarlavha bilan beriladi. Bu oyna dastur kodini kiritish va tahrir qilish uchun mo‘ljallangan (4.4- rasm).

Ekranning chap past qismida Obyektlar inspektori oynasi joylashgan. Bu oyna Object Inspector kabi sarlavxa bilan beriladi (4.5-rasm). Bu oynada vizual komponentalarga xususiyatlar o‘rnatiladi. Bu oynani dasturchi o‘zi xoxlagan ekran sathiga joylashtirishi mumkin.

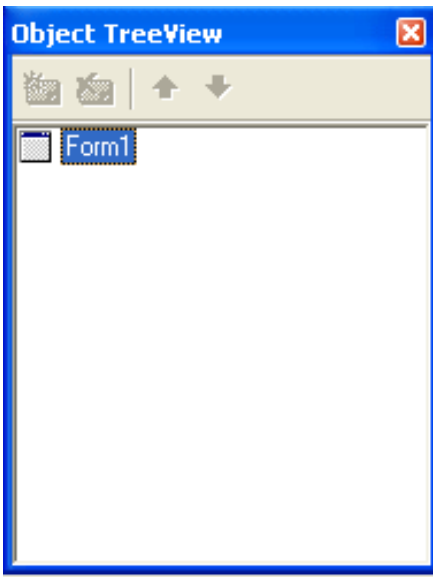


4.4-rasm. Dastur kodi muharriri



4.5-rasm. Obyektlar inspektori oynasi

Obyektlar inspektori oynasi ostida Obyektlarni daraxtsimon ko‘rinishi oynasi (4.6-rasm) joylashgan. Bu oynada ilovadagi barcha obyektlar daraxt strukturasi shaklida ifodalangan bo‘ladi. Formalar, dastur kodi va boshqa dastur tarkiblari fayllari berilgan bo‘ladi.



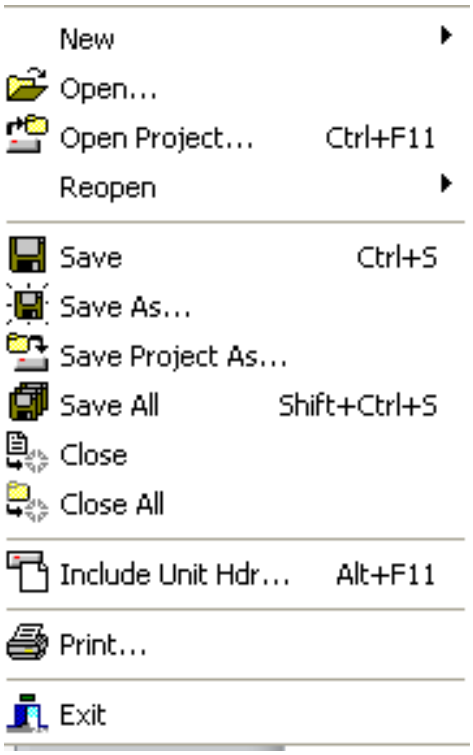
4.6-rasm. Obyektlarni daraxtsimon ko‘rish oynasi

Bu interfeys ilova tuzatayotganda eng asosiy foydalanayotgan qurolimiz bo‘ladi.

Borland C++ Builder 6 muhiti asosiy menyu buyruqlari

File buyruqlari guruhi:

Menyuning birinchi File (Fayl) buyrug’i guruhi 4.7-rasmda berilgan



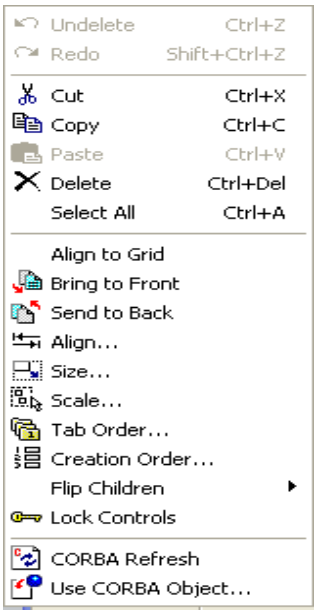
4.7-rasm. File buyrug’i guruhi

Bu menyu buyruqlari fayllar bilan ishlashga mo‘ljallangan va qo‘yidagi amallarni bajaradi: (**New**) yangi fayl hosil qilish, forma (oyna) ochish, mavjud

fayllarni (**Open**) ochish, fayllarni saqlash (**Save**) va (**Close**) yopish, ilova dastur kodi matnini (**Print**) chop qilish mumkin.

Edit buyruqlari guruhi:

Edit (Muharrir), 4.8-rasmda ifodalangan.

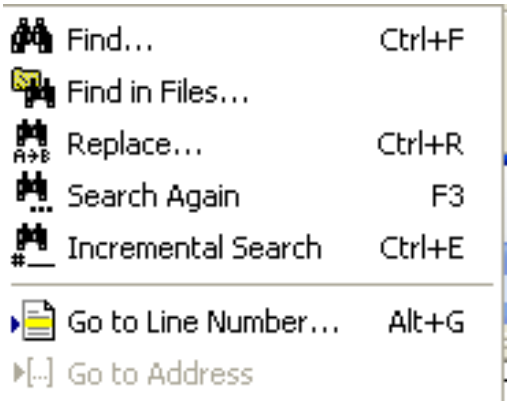


4.8-rasm. Edit buyruqlari guruhi

Menyuning bu buyruqlari guruhi tahrir qilish uchun zarur bo‘lgan buyruqlardir. Masalan inkor qilish (**Undelete**) va (**Redo**) takrorlash, qirqib olish(**Cut**), nusxa olish (**Copy**), nusxa olinganni qo‘yish (**Paste**) va o‘chirish (**Delete**) kabi amallarni o‘z ichiga olgan.

Search buyruqlari guruhi:

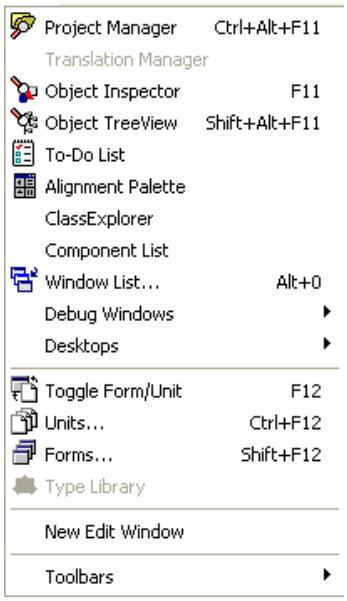
Search (Izlash)



4.9-rasm. Search (Izlash)

Menyuning bu buyruqlari guruhi fayllardan biror matnni izlash va avtomatik almashtirish, kerakli satrga o‘tish va uni o‘zgartish imkonini beradi.

View buyruqlari guruhi:

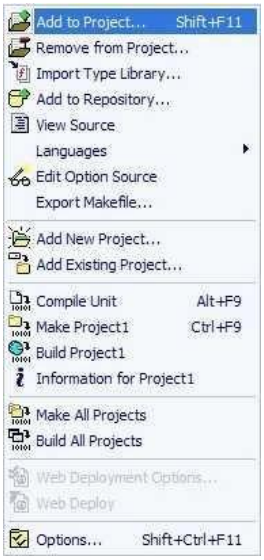


4.10-rasm. View (Ko‘rish) buyruqlar guruhi

Menyuning bu buyruqlari guruhidan ilova va komponentalarni boshqarishni asosiy muloqot oynalari chaqiriladi. Masalan, ilova menedjeri (**Project Manager**), komponentlar ro‘yxati (**Component List**) va oynalar ro‘yxati (**Window List**).

Project buyruqlari guruhi:

Project (Ilova) 4.11-rasmda keltirilgan.

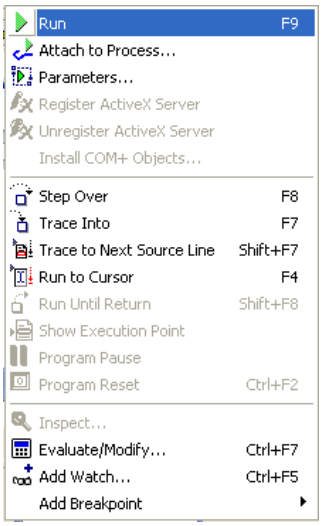


4.11-rasm. Project buyruq guruhlari

Menyuning bu buyruqlari guruhida ilovani boshqarish buyruqlari yig’ilgan. Bu buyruqlar yordamida fayllarni qo‘shish, o‘chirish, VCL komponentalar kutubxonasiga komponenta qo‘shish, kompilyatsiya qilish va shunga o‘xshash amallarni bajarish mumkin.

Run buyruqlari guruhi:

Run (Bajarish) 4.12-rasmda ifodalangan.

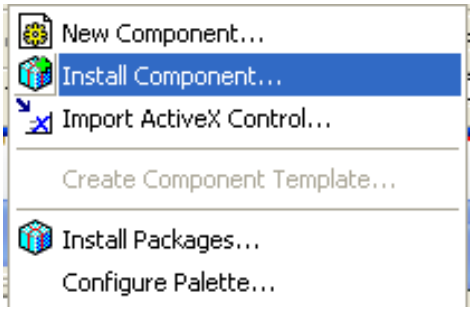


4.12-rasm. Run buyruqlar guruhi

Menyuning bu buyruqlari guruhi yordamida ilovani ishga tushirish va bekorqilish, ilovani butunlay va qadamba-qadam rejimda ishga tushirish, ko‘rish uchun qo‘shimcha o‘zgaruvchilar kiritish, ilova bajarilishini to‘xtatuvchi belgilar kiritish mumkin.

Component buyruqlari guruhi:

Component(Komponent) buyruqlar guruhi 4.13-rasmda berilgan.



4.13-rasm. Component (Komponent) buyruqlar guruhi

Menyuning bu buyruqlari guruhi yordamida yangi komponentalarni qo‘shish va konfiguratsiyalarini aniqlash ishlari bajariladi.

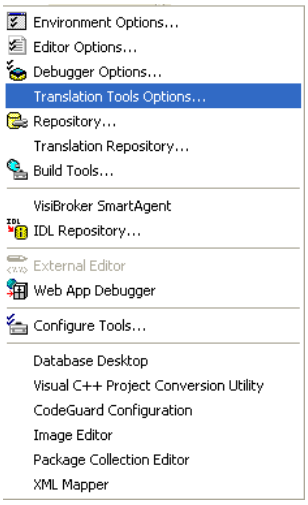
Database buyruqlari guruhi:

Database (Ma’lumotlar bazasi) buyruqlar guruhi (4.14-rasm) ma’lumotlar bazasi bilan ishlaydigan buyruqlarni o‘z ichiga olgan.

4.14-rasm. Database (Ma’lumotlar bazasi) buyruqlar guruhi

Tools buyruqlari guruhi:

Tools (Instrumentlar) buyruqlar guruhi 4.15-rasmda berilgan

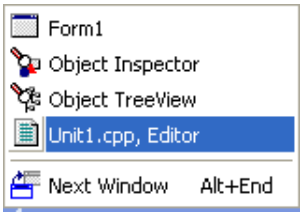


4.15-rasm. Tools buyruqlar guruhi

Menyuning bu buyruqlari guruhi yordamida ilova parametrlarini o‘rnatish va yordamchi dasturlar buyruqlarini chaqirish mumkin.

Windows buyruqlari guruhi:

Windows (oyna) buyruqlari guruhi

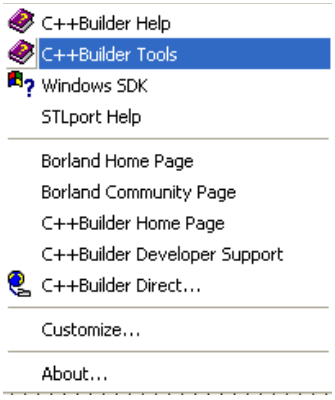


4.16-rasm. Windows(oyna) buyruqlari guruhi

Menyuning bu buyruqlari guruhi yordamida interfeysning oynalarini boshqarish mumkin.

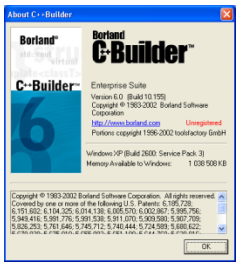
Help buyruqlari guruhi:

Help (Yordam) buyruqlari guruhi



4. 17-rasm. Help (Yordam) buyruqlari guruhi

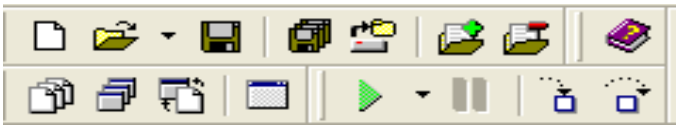
Menyuning bu buyruqlari guruhi yordamida muhit, til, komponentalar va kompyuter haqidagi ma’lumotlarni olishimiz mumkin. Masalan, quyidagicha:



4.18-rasm. About buyrug’i oynasi

Tezkor tugmalar:

Tezkor tugmalar asosiy menyu bilan yonma-yon joylashgan.



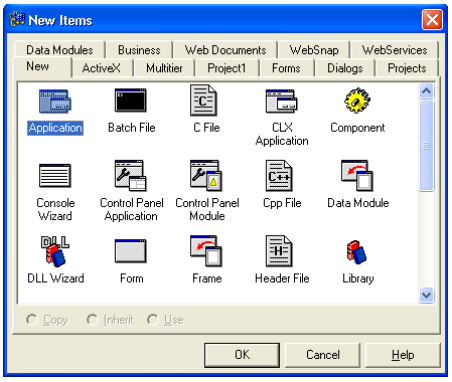
4.19-rasm.Tezkor tugmalar

Bu tugmalar ko‘p ishlatiladigan, tez-tez murojaat qilinadigan bo‘lganligi uchun alohida guruhlarga ajratilgan holda kichik piktogrammalar yordamida berilgan. Bu tugmalar asosiy menyuda ham mavjud, ammo vaqtdan yutish uchun ajratib ham qo‘yilgan.

Standard (Standart) tugmalar

Dasturning obyektlarini hosil qilish uchun Standart tugmalardan foydalaniladi. Bu obyektlarga dasturlar, yangi formalar, fayllar, biblioteka lar va hakozolar kirishi mumkin. Yangi obyektni hosil qilish uchun **New** tugmasini bosamiz. Bunda **New Items** (4.20-rasm) muloqot oynasi paydo bo‘ladi, bundan ilova tuzish uchun kerakli obyektni tanlaymiz. Bu oynada ko‘plab obyektlar ma’lum guruhlarga ajratilgan holda joylashgan.

Bu guruhning boshqa tugmalari faylni saqlash (Save) va ochish (Open) uchun xizmat qiladi. AddFileToProject tugmalari orqali ilovaga fayl qo‘shish va RemoveFileFromProject fayl o‘chirish imkonini beradi.



4.20-rasm. New Items muloqot oynasi

View (Ko‘rish) tugmasi

Bu guruhga quyidagi tugmalar jamlangan (New Form) yangi forma hosil qiladigan, (ViewUnit) modullarni qurish, (ViewForm) formani qurish, (**Toggle Form/Unit**) formadan modulga, moduldan formaga o‘tishni ta’minlaydigan tugmachalar.

Toggle Form Unit formadan kodni tahrirlashga, tahrirlashdan formaga o‘tkazuvchi tugma. Agar ilovada bir nechta forma qatnashsa, ularni qurish uchun **View Form (Shift+F12)** tugmadan foydalaniladi.

Yangi forma hosil qilish uchun **New Form** tugma bosiladi. Bunda yangi forma dizayneri paydo bo‘ladi. Odatda sarlavhasi **Form2** kabi nomlangan bo‘ladi. Yangi forma ilovada yangi oynalarni ko‘rsatish va ifodalash uchun zarur bo‘lishi mumkin.

Debug tugmasi

Bu guruhdagi tugmalar ilovani ishga tushirish uchun mo‘ljallangan. **Run (F9)** tugmasi ilovani bajarish va ishga tushirish uchun, **Pause** ilovani vaqtincha

to‘xtatib turish uchun, **TraceInto (F7)** va **Stepover (F8)** ilovani qadamba-qadam bajarish uchun mo‘ljallangan**.**

Custom tugmasi guruhi

Bu guruhda Help contents tugmasi mavjud bo‘lib, muhitning yordam tizimini chaqirish uchun ishlatiladi. Bu tugmalar guruhining joyi qat’iy tayinlanmagan, o‘zimiz xoxishimizga qarab asosiy panelning istalgan joyiga joylashtirishimiz mumkin. Buning uchun kursorni tugmalar guruhining chap chekkasiga olib boramiz va sichqonchaning chap tugmasini bosib istalgan joyga joylashtirishimiz mumkin. Agar sichqonchaning o‘ng tugmasini bossak 4.21-rasmdagi kabi menyu paydo bo‘ladi. Bunda asosiy menyu panelidagi barcha tugmalar guruhini qayta o‘rnatish mumkin.

4.21-rasm. Tugmalar guruhini o‘rnatish

Nazorat savollari

1. Borland C++ Builder dasturi muhiti strukturasi
2. Borland C++ Builder dasturi ob’ektlarining hususiyatlari
3. Borland C++ Builder 6 muhiti asosiy menyu buyruqlari
4. Project buyruqlari guruhi
5. Run buyruqlari guruhi
6. Borland C++ Builderning Form va Kod oynalari
7. Form oynasi Properties ob’ektining xuxusiyatlari
8. Form oynasi Events ob’ektining xuxusiyatlari

4.2. Borland C++ Builder komponentlarini o‘rganish

1. VCL komponentlar palitrasi

Borland C++ Builder 6 muhitini VCL vizual komponentalari bilan tanishamiz. Borland C++ Builder 6 muhitida ilova yaratish VCL vizual komponentalaridan keraklilarini formaga joylashtirishdan boshlanadi. Komponentalar ilovaning g’ishtchalari (elementlari, qismlari) deyishimiz mumkin. Builder so‘zining ma’nosi quruvchi degan ma’noni anglatadi. Barcha VCL komponentalari palitrasi, asosiy menyuning o‘ng tomonida joylashgan bo‘ladi. Komponentalar palitrasi ma’lum bir vazifalariga ko‘ra komponentalar to‘plamidan iborat komponentalar sahifasiga ajratilgan. Bu sahifalar ma’no va vazifasiga ko‘ra bir-biriga yaqin bo‘lgan komponentalarni bitta guruhga birlashtirgan. Sichqonchaning chap tugmasini sahifa komponentalari nomi ustida bossak, mazkur sahifa komponentalari guruhi ekranda paydo bo‘ladi. Masalan, Standard sahifasini bossak, ekranda qo‘yidagi komponentalar paydo bo‘ladi (4.22- rasm).

4.22-rasm. Komponentalar oynasi

C++ Builder 32 razryadli takomillashtirilgan Vizual Komponentalar Kutubxonasi VCL (Visual Component Library) bilan birgalikda yetkazib beriladi. Bu kutubxona eng murakkab ilovalarni qurish uchun mo‘ljallangan 100 dan ortiq takroran qo‘llanadigan komponentalardan iborat. Kutubxonaning asosiy komponentalari Palitralar komponentalarining Instrumental Panelida berilgan. Komponentalar belgilari dasturingiz shakliga (formaga) olib o‘tiladi.

Kutubxona Windows operatsion tizimlaridagi Foydalanuvchi Grafik Interfeysi standart interfeys obyektlarining to‘liq inkapsulyatsiyalanishini o‘z ichiga oladi. Ular orasida, ixtisoslashgan komponentalar bilan bir qatorda, relyatsion ma’lumotlar bazasini boshqarish uchun mo‘ljallangan komponentalar alohida o‘rin egallaydi. Ishonchli va samarali dasturlarni yaratishda C++ Builder OYD imkoniyatlaridan to‘liq foydalanadi. C++ Builder bu OMD ekan, OLE (OCX) boshqaruvchi elementlarni kiritish uncha qiyinchilik tug’dirmaydi.

C++ Builder bosh xususiyati avvalambor, uning dasturni vizual ishlash jarayonida nafaqat tayyor komponentalardan foydalanish, balki yangi komponentalarni yaratish qobiliyatida ham namoyon bo‘ladi. Yangi komponentalar, dastlabki komponentalar kabi, sodda bo‘lishi mumkin, bunda ularning funktsional imkoniyatlari ozgina kengaytirilgan yoki o‘zining mutlaqo

o‘ziga xos ko‘rinishi, xulq-atvori va kodining mazmuni bilan farqlanadigan bo‘ladi. Komponentalarning yaratilishi OYD ning vorislik mexanizmiga tayanadi, cheklanishlarga deyarli ega bo‘lmaydi hamda qo‘yidagi bosqichlardan o‘tadi:

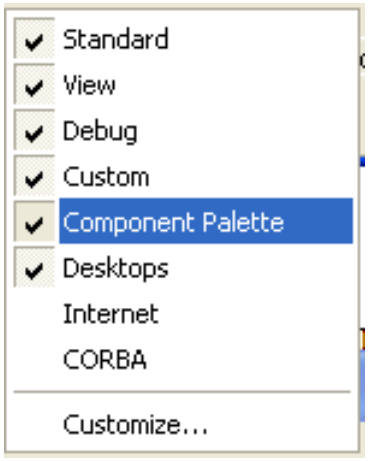
- ***mavjud komponenta turiga vorislik;***
- ***yangi xususiyatlar, metodlar va voqealarni aniqlash;***
- ***yaratilgan komponentani qayd etish.***

Qidirish oson bo‘lishi uchun, Palitra funktsional jihatdan o‘xshash komponentalarni birlashtiradigan qo‘shimcha ilovalar bilan bo‘lingan. Tanlab olingan komponentaning kontekst menyusini unga sichqonchaning o‘ng tugmasini bosib ochish mumkin.

Bular ichidan **OK** yozuvli **Button** tugmasini, **A** yozuvli **Label** tugmasini osonlik bilan topish mumkin.

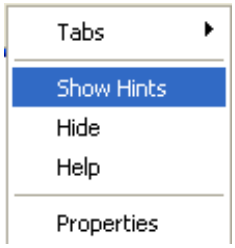
Agar sahifada komponentalar juda ko‘p bo‘lsa, harakatlanuvchi tugmalar orqali o‘ng va chapga panelni harakatlantirish orqali barcha komponentalarni ko‘rishimiz mumkin. Barcha komponentalar biror nom yoki maxsus ikonka orqali berilgan, agar sichqoncha kursorini ikonka ustiga oborsak shu komponenta nomi paydo bo‘ladi. E’tibor bergan bo‘lsangiz, ko‘pchilik komponentalar to‘rtburchak yoki dumaloq shaklga ega. Ixtiyoriy komponentani formaga ikki xil yul bilan joylashtirish mumkin. Komponenta ustida sichqonchani olib borib chap tugmasini ikki marta bossak mazkur komponenta forma markazida paydo bo‘ladi. Ikkinchi usul sichqoncha chap tugmasini komponenta ustida bitta bosib siljitish bilan formaga joylashtirish mumkin. Formaga joylashtirilgan komponentani formaning ixtiyoriy qismiga siljitishimiz mumkin. Komponetaning o‘lchamlarini sichqoncha yordamida o‘zgartirish mumkin. Formada turgan ixtiyoriy komponentani aktivlashtirish yoki passivlashtirish mumkin. Aktivlashtirish degani shu komponentaning ustida amal bajarish xususiyatini o‘zgartirish imkoniyatini beradi, buning uchun, mazkur komponentaning ustida sichqoncha tugmasini bosish kifoya. Aktivlik belgisi shu komponentani boshqalaridan ajralib belgilanib ko‘rinib turadi. Buni obyektlar inspektoridan ham ko‘rish mumkin. Sichqoncha yordamida tanlangan komponentani formadan o‘chirib tashlash mumkin.

Komponetalar palitrasini asosiy menyu panelidan olib tashlash mumkin. Buning uchun sichqonchaning o‘ng tugmasini bosish va hosil bo‘lgan menyu (4.23-rasm) dagi **Component Palette** so‘z oldidagi belgini sichqoncha tugmasini ikkita bosib, olib tashlash kifoya qiladi.



4.23-rasm. Kontekst menyu oynasi

Komponentlar palitrasining o‘rni Borland C++ Builder 6 interfeysida qat’iy tayinlanmagan. Uni o‘zimiz xoxlagancha joylashtirishimiz mumkin. Buning uchun kursorni palitraning chap chekkasiga olib borib sichqonchaning chap tugmasini bosib qo‘yib yubormagan holda, yangi joyga joylashtirish mumkin. Agar palitra asosiy menyudan tashqariga chiqib ketsa yangi oyna ochiladi. Bunda komponentalar palitrasiga qo‘shimcha buyruqlar kiritish uchun sichqonchaning ung tugmasi bosiladi. Bunda konteks menyu paydo bo‘ladi (4.24-rasm).



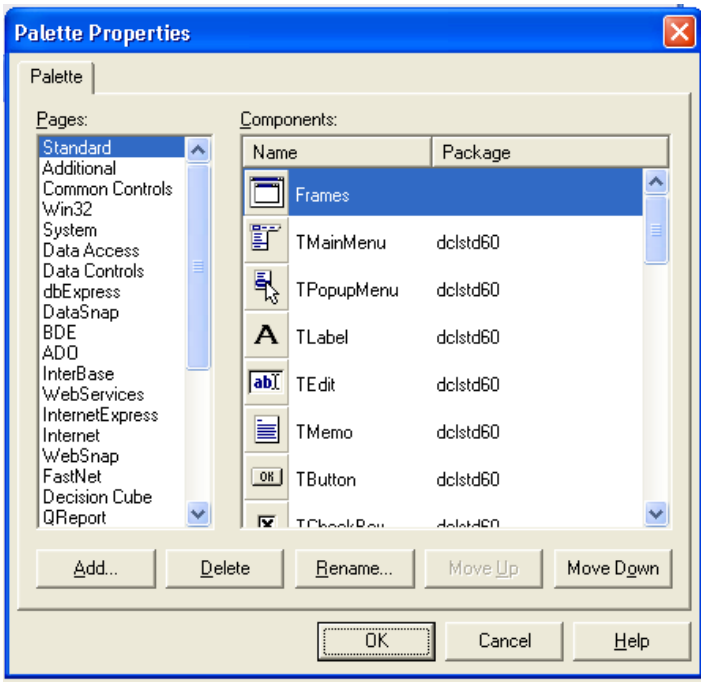
4.24-rasm. Buyruqlar palitrasi kontekst menyusi

Bu oynada ixtiyoriy sahifaga **Tabs vositasida** tez o‘tish imkoniyati mavjud (4.25-rasm), **ShowHints** tushuntish ma’lumotlari beriladi, **Hide** palitrani bekitish, **Help** ma’lumotlarni olish, yordamni chaqirish, **Properties** xususiyatlarni o‘rnatish.

Bu oynada ixtiyoriy sahifani qaytadan ko‘rib chiqish mumkin. **Rename** buyrug’i orqali sahifani qayta nomlash mumkin. **Add** buyrug’i orqali yangi sahifa qo‘shish mumkin. **Delete** ixtiyoriy sahifani o‘chirish mumkin. Bu sahifalarning berilish ketma-ketligini o‘zgartirish imkoniga egamiz. Masalan, **System** sahifasi **Standard** sahifasidan keyin berilgan edi, **System** satrini aktivlashtiramiz va sichqonchaning chap tugmasini bosamiz.

MoveUp buyrug’i orqali **Standard dan** tepaga joylashtiramiz va **OK** ni bosamiz. Endi **System** sahifasi Standard, sahifasidan oldin joylashgan bo‘ladi.

Xuddi shuningdek komplnentalarni **MoveUp** va MoveDown buyruqlari, joylashish ketma-ketligini o‘zgartirish mumkin.



4.25-rasm. Komponentalar palitrasini o‘rnatish oynasi

Yaratilayotgan ilova, ya’ni yangi loyixa ustida ishlash dialog oynasini, ya’ni boshlang’ich formani qurishdan boshlanadi.

Boshlang’ich forma **Form1** formasining hususiyatlarini o‘zgartirish hamda unga extiyojga qarab kerakli komponentalarni (kiritish va chiqarish maydonlar, buyruqli tugmalar) ni o‘rnatish orqali yaratiladi.

Formaning hususiyatlari (4.1-jadval) uning tashqi ko‘rinishi, o‘lchamlari, sarlavha matni hoshiyasining ko‘rinishini belgilaydi. Forma va uning komponentalari hususiyatlari va qiymatlarini o‘zgartirish uchun **Object Inspector** oynasidan foydalaniladi. Bu oynaning yuqori qismida ob’ektning nomi hamda hususiyatlarining joriy vaqtdagi qiymatlari ko‘rsatiladi. **Properties** (xususiyati) qistirmasining chap kolonkasida ob’ektning hususiyatlari, o‘ng tomonda esa ularning qiymatlari keltiriladi.

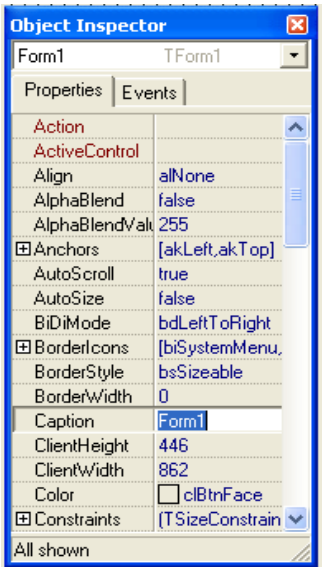
4.1-jadval. Formaning (***mform*** ob’ektining) hususiyatlari

Hususiyat	**Mazmuni**
Name	Formaning nomi. Dasturda formaning nomi formani boshqarish va forma komponentalariga murojaat qilish uchun foydalaniladi.
Caption	Sarlavha matni
Width	Formaning kengligi

Height	Formaning balandligi
Top	Formaning yuqori chegarasidan ekranning yuqori chegarasigacha bo‘lgan masofa
Left	Formaning chap chegarasidan ekranning chap chegarasigacha bo‘lgan masofa
BorderStyle	CHegaraning ko‘rinishi. CHegara oddiy (bsSizeable), ingichka (bs Single) bo‘lishi yoki umuman bo‘lmasligi (bsNone) mumkin. Agar oynaning chegarasi oddiy bo‘lsa, uni foydalanuvchi sichqonchadan foydalanib, o‘zgartirish mumkin. Ingichka chegarali oyna o‘lchamlarini o‘zgartirib bo‘lmaydi. Agar chegara bo‘lmasa, u holda ekranga sarlavhasiz oyna chiqarilishi mumkin. Bunday oynaning holati va o‘lchamlarini dasturning ishi mobaynida o‘zgartirish mumkin emas.
BorderIcons	Oynani boshqarish tugmalari. Xususiyatinig qiymati dasturning ishi davomida foydalanuvchilar qaysi tugmalardan foydalanish mumkinligini ko‘rsatadi. Hususiyatning qiymati biSystemMenu, biMinimaze, biMaximaze va biHelp hususiyatlarining qiymatlarini aniqlash orqali beriladi. biSystemMenu xususiyati ixchamlash va sistema tugmalariga, biMinimaze— ixchamlash tugmasiga, biMaximaze — kengaytirish tugmasiga, biHelp — ma’lumotnomalarni chiqarish tugmasi bilan ishlashga ruxsat beradi.
Icon	Dialog oynasi sarlavhasidagi sistema menyusini chaqirishni anglatuvchi nishon.
Color	Fon rangi. Rangni uning nomini ko‘rsatib yoki operatsion sistemaning ranglari gammasiga bog’lab qo‘yish orqali belgilash mumkin. Ikkinchi holda ranglar joriy ranglar sxemasi bo‘yicha aniqlanadi va operatsion sistemaning ranglar sxemasi o‘zgarganda o‘zgaradi.
Font	SHrift. Forma sirtida "ko‘rsatilmaganda ham" foydalaniladigan joriy shrift. Formaning xususiyati o‘zgartirilganda forma sirtida joylashgan komponentalarning Font xususiyati avtomatik tarzda o‘zgaradi, ya’ni komponentalar formadan Font xususiyatini meros qilib oladi.

Forma yaratishda birinchi navbatda caption (sarlavha) xususiyatining qiymatini o‘zgartirish lozim. Bizning misolimizda "Form1" matning "yugurish tezligi" bilan almashtirish kerak. Buning uchun **Object Inspector** oynasida sichqoncha tugmasini **Caption** satrida chertamiz. Natijada hususiyatning joriy qiymati ajratib ko‘rsatiladi va shu satrda kursor paydo bo‘ladi. SHundan keyin "YUgurish tezligi" matnini kiritish mumkin. (4.26-rasm).

Xuddi shu usul bilan formaning kengligi va balandligini aniqlovchi **Height** va **width** hususiyatlarni o‘zgartirish mumkin. Formaning o‘lchamlari, uning holati hamda boshqa boshqaruv elementlarining o‘lchamlari va ularning forma sirtidagi holatlari piksellarda (ekrandagi nuqtalar) beriladi. **Height** va **width** hususiyatlarini mos ravishda 250 va 330 qilib belgilang.



4.26-rasm. Hususiyat qiymatini qiymatni kiritish orqali o‘zgartirish

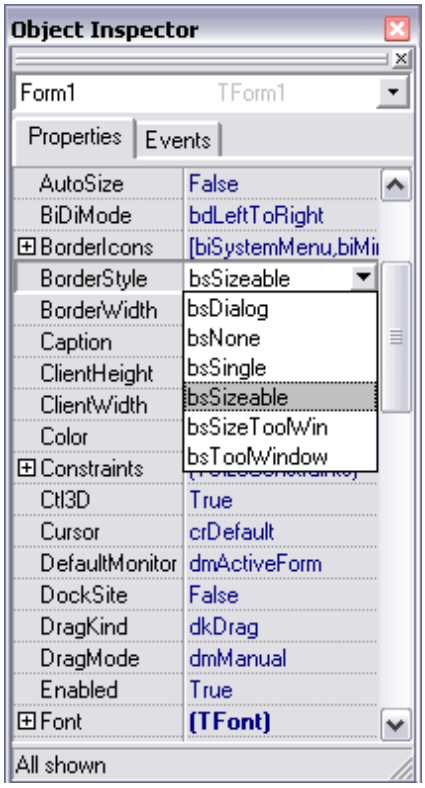
Forma - *bu oddiy oynadir. SHuning uchun uning o‘lchamlarini boshqa oynalar kabi sichqoncha yordamida o‘zgartirish mumkin. Bunda* ***Height*** *va* ***Width*** *hususiyatlarining qiymatlari avtomatik tarzda o‘zgaradi.*

Dialog oynasining ekrandagi holati formani tashkil qilishdagi holatiga mos keladi. Bu holatni **Tor** (ekranning yuqori chegarasidan chekinish) va **Left** (ekranning chap chegarasidan chekinish) hususiyatlarining qiymatlari belgilab beradi. Bu qiymatlarni ham sichqoncha yordamida o‘zgartirish mumkin.

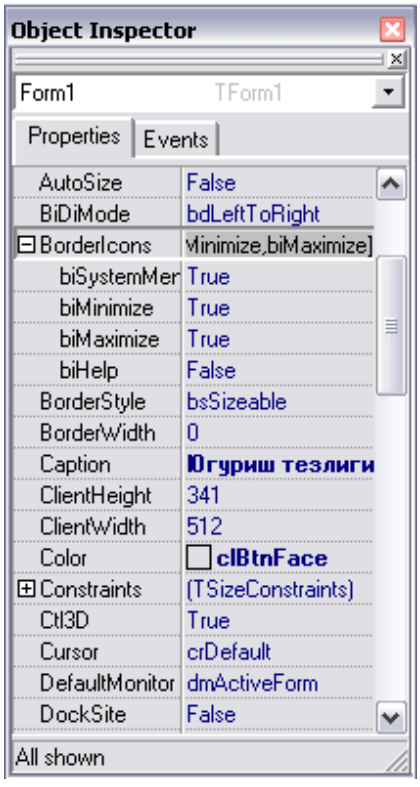
Ayrim hususiyatlarni tanlashda, masalan, **Borderstyle**, hususiyatning joriy qiymatini belgilashda, o‘ng tomonda ochiladigan ro‘yxat taklif qilinadi. Qiymatni ana shu ro‘yxatdan tanlash mumkin.

Ayrim hususiyatlar murakkab hisoblanadi, chunki ularning qiymatlari boshqa hususiyatlarning qiymatlaridan kelib chiqib aniqlanadi. Bunday

hususiyatlarning oldida "□" nishoni turadi. U chertilsa, aniqlovchi hususiyatlar ro‘yxati taklif qilinadi. Masalan, BorderIcons xususiyati oynalarni boshqarishning qaysi tugmalari bilan dastur ishi davomida ishlash mumkinligini belgilaydi. Agar biMaximize xususiyatiga False qiymati berilsa, dasturning ishlashi jarayonida **Razvernut** tugmasi oyna sarlavhasida ko‘rinmaydi.



Hususiyat qiymatlarini ro‘yxatdan tanlash



Murakkab hususiyatlarning ochilgan ro‘yxati

Ayrim hususiyatlarning yonida uch nuqtali tugma joylashgan. Bu hususiyat qiymatini aniqlashda yangi dialog oynasidan foydalanish mumkinligini anglatadi. Masalan, **Font** murakkab xususiyatining qiymatini belgilashda shrift tanlashning standart oynasidan foydalanish mumkin.

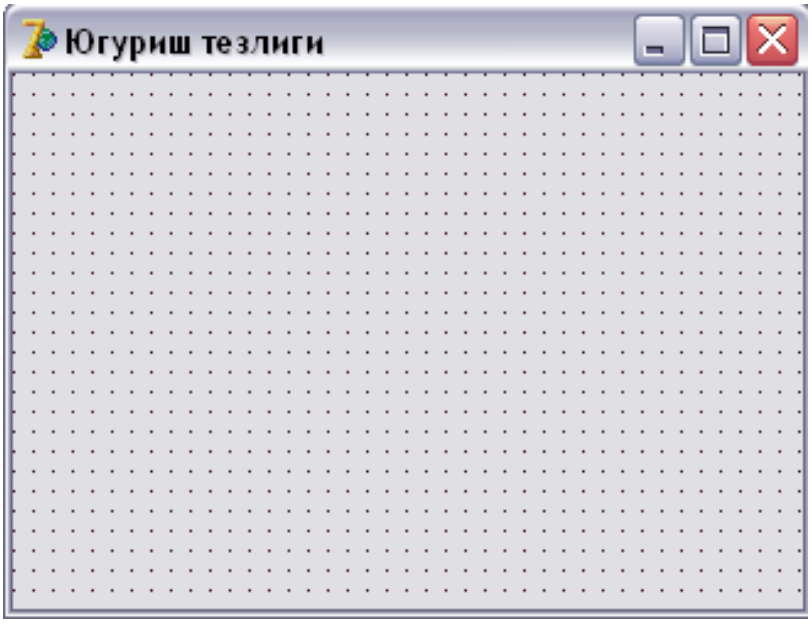
4.2-jadvalda yaratilayotgan formaning o‘zgartiriladigan hususiyatlari keltirilgan. O‘zgarmaydigan hususiyatlar bu jadvalga kiritilmagan.

4.2-jadval. Boshlang’ich formaning hususiyatlari

Hususiyat	Qiymat
Caption	Yugurish tezligi
Height	250
Width	330
BorderStyle	bsSingle

BorderIcons .biMinimize	False
BorderIcons .biMaximize	False
Font. Size	10

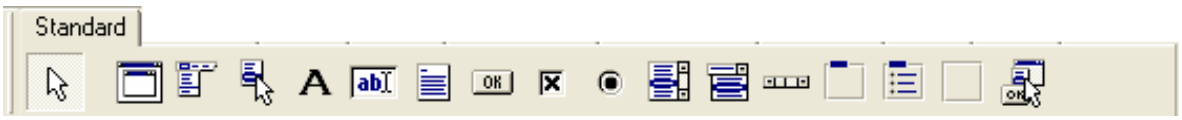
Keltirilgan jadvalda ayrim hususiyatlarning yonida nuqta (.) belgisi turibdi. Bu hususiyatning aniqlanadigan qiymatini belgilashni bildiradi. Boshlang'ich formaning jadvaldagi hususiyat lari o'rnatilganidan so'ng, forma rasmdagi ko'rinishni oladi (4.27-rasm).



4.27-rasm. Ilova uchun Forma

Yugurish tezligining dasturi foydalanuvchidan boshlang'ich ma'lumotlar – masofa hamda shu masofani yugurib bosib o'tish vaqtini olishi lozim. Bunday hollarda odatda boshlang'ich ma'lumotlar kiritish maydonlariga klaviatura yordamida kiritiladi. SHuning uchun formaga kiritish maydoni **Edit** komponetalarini joylashtirish lozim.

Eng ko'p foydalaniladigan komponentalar **Standard** qurollar panelida joylashtirilgan (4.28-rasm). Formaga komponentani qo'shish uchun komponentalar palitrasidan shu komponenta piktogrammasi ustiga sichqonchani keltirib, chertiladi. So'ngra kursorni komponentaning chap yuqori burchagi formada joylashishi kerak bo'lgan nuqtaga o'rnatiladi va chap tugmani yana bir marta chertiladi. Natijada formada standart o'lchamli komponenta paydo bo'ladi.



4.28-rasm. Standard qurollar panelining komponentalari ro‘yxati

Komponentaning o‘lchamlarini ularni formaga qo‘shish jarayonida belgilanishi mumkin. Buning uchun komponenta sichqoncha yordamida tanlanganidan so‘ng, kursorni formaning komponenta chap yuqori burchagi turishi kerak bo‘lgan nuqtasiga keltirib, chap tugmasi bosiladi va uni qo‘yib yubormagan holda kursorni komponentaning quyi o‘ng burchagi turishi kerak bo‘lgan nuqtaga olib kelinadi. SHundan so‘ng sichqonchaning chap tugmasini qo‘yib yuborish mumkin. Formada ko‘rsatilgan o‘lchamdagi komponenta paydo bo‘ladi.

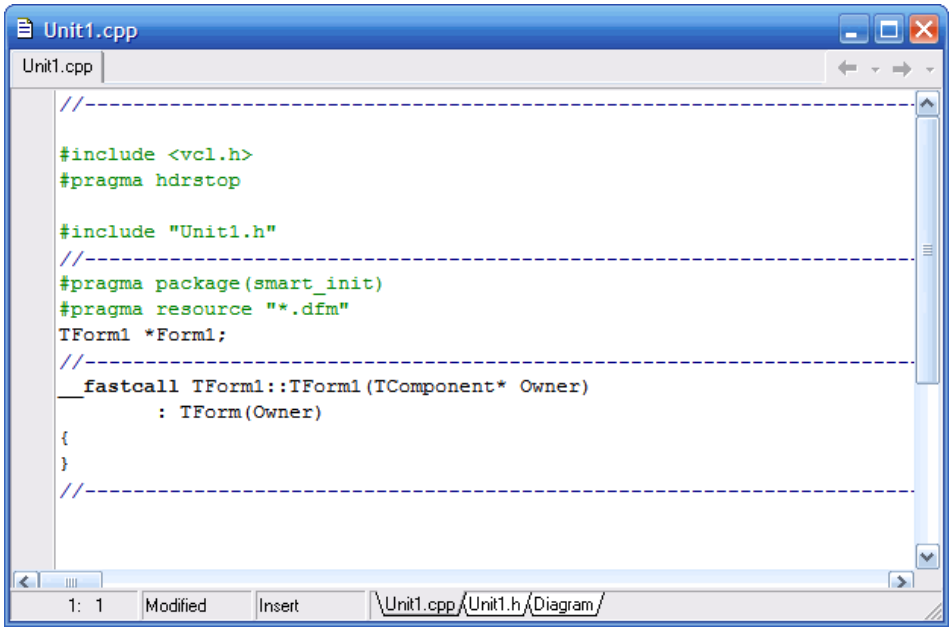
Delphi formaga qo‘shilayotgan xar bir komponentaga nom beradi. Bu nom komponenta nomi va uning tartib nomeridan iborat bo‘ladi. Masalan, formaga ikkita Edit komponentasi qo‘shilgan bo‘lsa, ularning nomlari mos ravishda Edit1 va Edit2 bo‘ladi. Dasturchi Name xususiyati qiymatini o‘zgartirib, bu nomlarni boshqasiga almashtirishi mumkin. Odatda sodda dasturlarda komponentalarning nomlari almashtirilmaydi.

Rasmda formaning ikkita Edit komponentalari, ya’ni kiritish maydonlarini qo‘shilganidan keyingi holati keltirilgan. Komponentalarning biri ajratilgan. Ajratilgan komponentaning hususiyatlari **Object Inspector** oynasida tasvirlangan. Boshqa komponenta hususiyatlarini ko‘rish uchun sichqonchaning chap tugmasini shu komponenta ustida chertish lozim. SHuningdek, komponenta nomini **Object TreeView** oynasidan yoki **Object Inspector** oynasining yuqori qismidagi ob’ektlarning ochiladigan ro‘yxatidan ham tanlash mumkin.

Inspektor oynasining yuqori qismida ob’ektni tanlash maydoni mavjud bo‘lib, unda tanlangan ob’ekt nomi ko‘rsatiladi. Yangi ob’ektni tanlash uchun strelkali tugma ustida sichqonchaning chap tugmasini bosish orqali ochiladigan (ochiladigan) ro‘yxatni ochishingiz kerak. Keyin ro‘yxatdan kerakli ob’ektni tanlang. Keyin tanlangan ob’ektning xususiyatlarini tahrirlashingiz mumkin. Ob’ektni tanlashning ikkinchi usuli - ob’ektning o‘zida sichqonchaning chap tugmachasini bosish. Ob’ektni tanlashning uchinchi usuli - **Object TreeView**-dagi ob’ekt nomiga sichqonchaning chap tugmachasini bosish kerak.

Kod muharriri:

Dastur matnini hosil qilish uchun kod muharriridan foydalaniladi, bu oynaning tashqi ko‘rinishi 4.29-rasmda keltirilgan.



4.29-rasm. Kod muharriri oynasi

- Unit1.cpp ilovangizning bajarilayotgan ishga tushirish kodini saqlaydi. Aynan shu yerda siz foydalanuvchining komponentalar obyektlariga ta’siri paytidagi dastur reaktsiyasiga javob beradigan voqealarning qayta ishlatgichlarini yozib qo‘yasiz.
- Unit1.h barcha obyektlar va ularning konstruktorlarining e’lonlariga ega. Voqealarni qayta ishlash funktsiyalari e’lonlaridagi kalit-so‘zga e’tibor bering (C++ Builder bu funktsiyalarni avtomatik tarzda generatsiya qiladi). fastcall tufayli parametrlar stek orqali emas, balki markaziy protsessor registrlari orqali uzatiladi. Voqealarni qayta ishlatgichlarning chaqirishlari tez-tez ro‘y berib turadi, shuning uchun stek xotirasidan parametrlarni tanlab olishga sarflanadigan vaqtning tejalishi ancha sezilarli natijalarni beradi. C++ Builder kompilyatsiya qiladigan va to‘playdigan ilovalarning yuqori darajada tez harakatlanishining sabablaridan biri ham shu yerda yashiringan.
- Project.cpp ilovada mujassamlangan barcha obyektlarga xizmat ko‘rsatadi. Har qanday yangi shakl, dasturiy modul yoki ma’lumotlar moduli avtomatik tarzda loyihaviy faylga kiritiladi. Siz bosh menyu komandasi - View | Project Sourse yordamida yoki Loyiha Administratorining kontekstli menyusidan shu nomdagi bo‘limni tanlab olib, Kod Muharriri darchasida loyihaviy fayl dastlabki matnining mazmunini ko‘rib chiqishingiz mumkin. Hech qachon loyihaviy faylni qo‘lda tahrir qilmang!

Balki siz, birinchi ilova ishlanmasini tugatib, dastlabki fayllarni keyingi seans uchun saqlab qolishni xoxlarsiz. Buning uchun qo‘yidagi xatti-harakatlardan birini bajarish kerak:

- ***File | Save All*** *komandasi ilovaning hamma dastlabki fayllarini saqlaydi.*
- ***File | Save*** *komandasi dasturiy modulning ikkala komandasini saqlaydi,* ***File | Save A****s komandasi esa ularga yangi nom berishga ruxsat etadi.*
- ***File | Save Project AS*** *komandasi, fayllarning joriy nomlaridan foydalanib, loyihaviy fayl tarkibiy qismlarining hammasidagi o‘zgarishlarni saqlaydi.*

Nazorat savollari

1. Borland C++ Builder komponentlarini tavsifi
2. *Borland C++ Builderning Standard menyusi komponentalari tavsifi*
3. Borland C++ Builderning Object Inspector oynasi tavsifi
4. Edit, Label va Memo komponentalari xususiyatlari
5. Button, CheckBox va RadioButton komponentalari xususiyatlari
6. MainMenu, PopupMenu, ListBox va ComboBox komponentalari xususiyatlari

4.3. Borland C++ Builder Standard komponentlar palitrasi

C++ Builder dasturlash tili. Xozirgi vaqtga kelib komp’yuter olamida ko‘plab dasturlash tillari mavjud. Paskal, C++, Delphi va boshqa dasturlash tillaridir. C++ dasturlash tili universal tildir. U UNIX sistemasi bilan bog’langan bo‘lib, bu sistemada ishlatiladigan bir qancha dasturlar C++ tilida yozilgan. C++ Denis Ritchi tomonidan 1972 yili UNIX tipidagi operasion sistemalarini yaratish uchun loyihalashtirilgan. Borland C++ dasturlash tili Windows uchun mo‘ljallangan bo‘lib, uning birinchi versiyasi Windows operatsion sistema qobig’ida ishlagan. Borland C++ dasturlash tili - bu dasturlarni qayta ishlash muxiti bo‘lib, Windows operatsion sistemasida ishlaydi. Unda ob’ektli dasturlash tillari bo‘lgan Object mujassamlashgan. Borland C++ dassturlash tili turli xolat protseduralarini qayta ishlash va dasturlarni qayta ishlashda vaqtdan yutish va boshqalarni o‘z ichiga oladi.

Dastur yaratish muhiti

Dastur yaratish umumlashgan muhiti Redaktor form - Shakllar muharriri, Inspektor ob’ektov - Ob’ektlar inspektori, Palitra komponentov - Komponentlar palitrasi, Administrator proekta - Proekt administratori va to‘la umumlashgan Redaktor koda - Kodlar muharriri hamda kodlar va resurslar ustidan to‘liq

nazoratni ta’minlaydigan , dastur ilovalarini tezkor yaratadigan Otladchik - instrumentov - Sozlash- instrumentlari kabilarni birlashtiradi.

Komponentlar

Komponentalarni shaklga o‘rnatish uchun komponentlar palitrasidagi kerakli piktogramma tanlanadi, so‘ngra shaklning komponenta joylanishi kerak bo‘lgan joyi tanlanadi. Shundan so‘ng komponentalar xossalarini ob’ektlar inspektori yordamida tahrirlash mumkin. Properties bandida komponentalar xossalarining ro‘yxati (chapda) va bu xossalarning qiymatlar ro‘yxati (o‘nggda) joylashgan.

Komponentalar ko‘rinadigan (vizual) va ko‘rinmaydigan (vizual bo‘lmagan) larga bo‘linadi. Vizual komponentalar bajarilish paytida proektlash paytidagidek paydo bo‘ladi. Bunga knopkalar va tahrirlanuvchi maydonlar misol bo‘la oladi. Vizual bo‘lmagan komponentalar proektlan vaqtida shakldagi piktogramma ko‘rinishida paydo bo‘ladi. Ular bajarilish paytida hech qachon ko‘rinmaydi, ammo ma’lum funksionallikga ega bo‘ladi (masalan, berilganlarga murojatni ta’minlaydi, Windowsning standart muloqatlarini chaqiradi).

Xossalar

Xossalar komponentalaming tashqi ko‘rinishi va tabiatini aniqlovchi atributlar hisoblanadi. Xossalar ustunidagi ko‘p xossalar komponentalari oldindan o‘rnatilgan (po umolchaniyu) qiymatlarga ega bo‘ladi (masalan, knopkplar balandligi). Komponentalar xossalari xossalar varag’i (Properties) da aks ettiriladi. Ob’ektlar inspektori komponentalaming nashr etilgan (published) xossalarini aks ettiriladi. published-xossalardan tashqari komponentalar umumiy (public), faqat ilovalarning bajarilish paytidagina murojat qilish mumkin bo‘lgan nashr qilingan xossalarga ega bo‘ladi. Xossalar ro‘yxati ob’ektlar inspektori xossalar varag’ida joylahadi. Xossalarni proektlash paytida aniqlash mumkin yoki ilovalarning bajarilish paytida ko‘rinishini o‘zgartirish uchun kod yozish mumkin. Komponenta xossalarini proektlash paytida aniqlash uchun shakldagi komponenta tanlanadi, ob’ektlar inspektori xossalari varag’i ochiladi, aniqlanadigan xossa tanlanadi va zurur bo‘lsa xossalar muharriri yordamida o‘zgartiriladi (bu kiritish uchun oddiy maydon yoki son, osilib tushuvchi ro‘yxat, ochiluvchi ro‘yxat, muloqat paneli va boshqalar bo‘lishi mumkin).Biror komponentaning xossalarini dasturning bajarilish paytida o‘zgartirish uchun ***«Imya Komponenta» -> «Nazvanie svoystva»*** tavsifiga o‘zaruvchidek murojat qilish kerak, ya’ni qiymatlarni o‘zimiz hohlagandek o‘qishimiz yoki almashtirishimiz mumkin.

Xodisalar

Ob’ektlar inspektorining xodisalar varag’i (Events) komponentalar tomonidan taniladigan xodisalar ro‘yxatini ko‘rsatadi. Har bir komnonenta

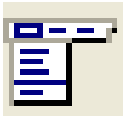
o‘zining shaxsiy xodisalarni qayta ishlovchi naborga ega bo‘ladi. C++ Builder da xodisalarni qayta ishlovchi funksiyalarni yozish va xodisalarni bu funksiya bilan bog’lashga to‘g’ri keladi. Biror bir xodisaga qayta ishlovchi yozib, siz dasturga bu xodisa ro‘y berganda yozilgan funksiyaning bajarilishini topshirasiz.

Xodisani qayta ishlovchini qo‘shish uchun shaklda xodisani qayta ishlovchi komponenta tanlanadi. So‘ngra xodisalar varag’ida ob’ektlar inspektori ochilib (Event bandi) xodisaning qatoridagi qiymatlar ustunida sichqonning chap tugmasi ikki marta bosiladi. Bu bilan C++ Builder ni xodisalarni qayta ishlash prototipini generatsiya qilishga va uni kodlar muharririda ko‘rinishiga majbur qiladi. Bu holda bo‘sh funksiya nomi generatsiya qilinadi va muharrir kod kiritilishi zarur bo‘lgan joyda ochiladi. Kursor buyruqlar qavslari ichiga joylashadi { ... }. So‘ngra xodisa sodir bo‘lganda bajarilishi kerak bo‘lgan kod kiritiladi. Xodisalarni qayta ishlovchi funksiya nomidan keyin ko‘rsatiladigan parametrlarga ega bo‘lishi mumkin.

4.30-rasm. Standard komponentalar

Komponentalar palitrasining Standard qo‘shimcha ilovalari komponentalari sizning dasturingizga Windows standart interfeysli elementlarning 14 tasining ulanishini amalga oshiradi.

TMainMenu



Bosh menyu komandalari Panelini va ularga mos keladigan tushib qoladigan menyularni yaratadi. Barcha menyu komandalarining identifikatorlari menyuning har qanday konkret komandasiga kirish huquqiga ega bo‘lgan Items xususiyati bilan aniqlanadi. AutoMerge xususiyati Merge va UnMerge metodlari bilan birgalikda turli shakldagi menyularning birlashish jarayonini boshqaradi.

TPopupMenu

Shakl yoki bironta boshqa komponenta uchun maxsus menyu yaratadi. E’tiborga oling, aynan shu maqsad uchun har qanday boshqa komponenta

PopUpMenu xususiyatiga ega bo‘lib, bu xususiyatda siz uning bilan bog’liq menyuga iqtibos qilishingiz mumkin.

Agar siz sichqonchaning o‘ng tugmasini shaklga yoki berilgan komponenta mansub bo‘lgan biron boshqa elementga bosish bilan maxsus menyu ekranda paydo bo‘lishini xoxlasangiz, AutoPopUp xususiyatining true qiymatini o‘rnating. Voqea qayta ishlatgichi - OnPopUp yordamida bevosita maxsus menyuning paydo bo‘lishi oldidan bajariladigan protsedurani aniqlashi mumkin.

TLabel

Shaklda tahrir qilib bo‘lmaydigan statik matnning to‘rtburchak sohasini aks ettiradi. Odatda matn boshqa komponenta nomidan iborat bo‘ladi.

Nom matni Caption xususiyatining qiymatidir. Alignment xususiyati matnni tekislash usulini aniqlaydi. Shrift o‘lchami avtomatik tarzda sohaning maksimal to‘ldirilishiga mos kelishi uchun, AutoSize xususiyatining true qiymatini o‘rnating. Kalta soha ichida matnning hammasini ko‘rish imkoniga ega bo‘lish uchun, WordWrap xususiyatining true qiymatini bering. Transparent xususiyatining true qiymatini o‘rnatsangiz, boshqa komponentaning bir qismini to‘g’ri uning ustida joylashtirilgan nom orasidan ko‘rinib turadigan qilishingiz mumkin.

TEdit

Axborot yakka satrining tahrir qilinayotgan kiritishidagi to‘rtburchak sohani shaklda aks ettiradi. Tahrir sohasining ichidagi boshlang’ich narsalarni Text xususiyatining qiymati bo‘lgan satr aniqlaydi.

TEdit komponentasi TCustomEdit sinfining to‘g’ridan-to‘g’ri hosilasi bo‘lib, uning barcha xususiyatlari, metodlari va voqealariga vorislik qiladi.

TMemo

Axborot ko‘plab satrining tahrir qilinayotgan kiritishidagi to‘rtburchak sohani shaklda aks ettiradi. Tahrir sohasining ichidagi boshlang’ich narsalarni Lines xususiyatining qiymati bo‘lgan satrlar massivi aniqlaydi. Ushbu xususiyat

qiymati ustunida tugmachani bossangiz, ro‘yxat elementlari muharririning darchasi ochiladi.

TMemo komponentasi TCustomMemo sinfining to‘g’ridan-to‘g’ri hosilasi bo‘lib, uning barcha xususiyatlari, metodlari va voqealariga vorislik qiladi.

TButton

Yozuvli to‘rtburchak tugmani yaratadi. Tugmacha bosilganda, dasturda biror-bir hatti- harakat nomlanadi (initsiallashtiriladi).

Tugmachalar ko‘proq **Dialogli** darchalarda qo‘llanadi. Default xususiyatining true qiymati tomonidan tanlab olingan yashirin tugmacha, Dialog darchasida har gal Enter klavishasi bosilganda OnClick voqea qayta ishlatgichini ishga tushiradi. Cancel xususiyatining true qiymati tanlab olgan bekorqilish tugmachasi, Dialog darchasida har gal Escape klavishasi bosilganda, OnClick voqea qayta ishlatgichini ishga tushiradi.

TVutton komponentasi TButtonControl sinfining hosilasi hisoblanadi.

TCheckBox

Ikkita holatga hamda tavsifiy matnga ega bo‘lgan kvadrat check-boxni yaratadi (bunda tavsifiy matn check-boxning vazifasini spetsifikatsiya qiladi).

Box holatini bildiruvchi «Check» biron-bir variantning tanlanishiga mos keladi (box ustidan tortilgan chiziq bilan belgilanadi). «UnCheck» holati esa tanlov olib tashlanishiga mos keladi - bunda Checked komponentasining xususiyati mos ravishda o‘zgaradi hamda OnClick voqeasi yuzaga keladi. Tavsifiy matn Caption xususiyatida saqlanadi. AllowGraed xususiyatining true qiymatini o‘rnatib, boxni to‘qroq rangli (masalan, kulrang) qilish mumkin. State xususiyati joriy holatni va box rangini aks ettiradi.

TCheckBox komponentasi TButtonControl sinfining hosilasidir.

TRadioButton

Ikkita holatga hamda tavsifiy matnga ega bo‘lgan yumaloq tugmachani yaratadi (bunda tavsifiy matn yumaloq tugmachaning vazifasini spetsifikatsiya qiladi).

Radio-tugmalar bir-birini istisno qiladigan tanlov variantlarining to‘plamidan iborat: ya’ni ushbu vaqt daqiqasida faqat bitta tugma tanlab olinishi mumkin (ichki qora doiracha bilan belgilanadi). Avval tanlangan tugmadan esa tanlov avtomatik tarzda olinadi. Radio-tugma bosilganda, Checked komponentasining xususiyati ham mos ravishda o‘zgaradi va OnClick voqeasi yuzaga keladi.

Odatda radio-tugmalar avvaldan shaklda o‘rnatilgan konteyner ichiga joylashtiriladi. Agar bitta tugma tanlangan bo‘lsa, ushbu guruhga mansub barcha boshqa tugmalarning tanlovlari avtomatik tarzda olib tashlanadi. Masalan, shakldagi ikkita radio-tugma, agar ular boshqa-boshqa konteynerlarda joylashgan bo‘lsagina bir paytning o‘zida tanlab olinishi mumkin. Agar radio- tugmalarning guruhlanishi ochiq-oydin berilmagan bo‘lsa, bu holda ularning hammasi, yashirin holda, konteyner darchalari (TForm, TGroupBox yoki TPanel) dan birida guruhlanadi.

TRadioButton komponentasi TButtonControl sinfining hosilasidir.

TListBox

Tanlash, qo‘shish yoki o‘chirish uchun mo‘ljallangan matn variantlari ro‘yxatining to‘rtburchak sohasini aks ettiradi.

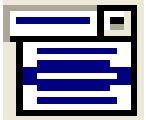
Agar ro‘yxatdagi barcha elementlar ajratilgan sohaga sig’masa, ro‘yxatni aylantirish lineykasi yordamida ko‘rib chiqish mumkin. Ro‘yxat elementlari Items xususiyatining ichida, dastur bajarilish vaqtida tanlab olinadigan element raqami esa ItemIndex xususiyatining ichida joylashgan bo‘ladi. Ro‘yxat elementlari matn muharririning darchasi Items xususiyati qiymatining grafasida tugmacha bilan ochiladi. Ro‘yxat elementlarini Items obyektining Add, Append, Delete va Insert metodlari yordamida dinamik tarzda qo‘shish, o‘chirish, orasiga joylash va o‘rnini almashtirish mumkin. Masalan:

```
ListBox1->Items->Add («Ro‘yxatning oxirgi elementi»);
```

Sorted xususiyatining true qiymati ro‘yxat elementlarini alifbo tartibida ajratib joylashtiradi.

TListBox komponentasi TCustomListBox sinfining hosilasi bo‘lib, uning barcha xususiyat, metod va voqealariga vorislik qiladi.

TComboBox



Tahrir sohasi hamda matn variantlarining tushib qoladigan ro‘yxati kombinatsiyasini tanlash uchun ishlatiladi.

Text xususiyatining qiymati bevosita tahrir sohasiga kiritib qo‘yiladi. Foydalanuvchi tanlab olishi mumkin bo‘lgan ro‘yxat elementlari Items xususiyatining ichida bo‘ladi. Dasturning bajarilish paytida tanlab olinishi mumkin bo‘lgan element raqami ItemIndex xususiyatining ichida bo‘ladi. Tanlab olingan matnning o‘zi esa SelText xususiyatining ichida bo‘ladi. SelStart va SelLength xususiyatlari matnning qaysi qismini tanlab olishni belgilab berish yoki matnning qaysi qismi tanlab olinganini bilish imkonini beradi.

Items obyektining Add, Append, Delete va Insert metodlari yordamida ro‘yxat elementlarini dinamik tarzda qo‘shish, o‘chirish orasiga qo‘yish va o‘rnini almashtirish mumkin. Masalan:

ComboBox->Items->Insert(0, «Ro‘yxatdagi birinchi element»);

Sorted xususiyatining true elementi ro‘yxat elementlarini alifbo tartibida navlarga ajratilishini ta’minlaydi. TComboBox komponentasining turini Style xususiyatidan tanlab olish mumkin.

TComboBox komponentasi TCustomComboBox sinfining hosilasi bo‘lib uning barcha xususiyatlari, metodlari va voqealariga vorislik qiladi.

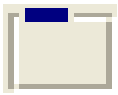
TScrollBar

Darcha, shakl yoki boshqa komponenta ichidagilarini ko‘rib chiqish uchun ishlatiladi. Masalan, biron-bir parametr qiymatini berilgan interval ichida harakatlanishi uchun, yugurgichli aylantirish lineykasini yaratadi.

Aylantirilayotgan obyekt xulq-atvorini OnScroll voqealar qayta ishlatgichi aniqlaydi. Foydalanuvchi Lineykaning o‘zida sichqonchani bosganda (yugurgichning har ikkala tomonida), yugurgich qanchaga surilishi kerakligini LargeChange xususiyatining qiymati aniqlab beradi. Foydalanuvchi sichqonchani strelkali tugmachalar (Lineyka oxiridagi) ustida bosganda yoki pozitsiyalash tugmachalarini bosganda, yugurgich qanchaga surilishi kerakligini SmallShange xususiyatining qiymati aniqlab beradi.

Min va Max xususiyatlarining qiymatlari yugurgichning yo‘l qo‘yilishi mumkin bo‘lgan joy almashinuvlari intervallarini belgilaydi. Sizning dasturingiz yugurgichni Position xususiyatining qiymati aniqlab beradigan kerakli pozitsiyaga joylashtirishi mumkin. SetPcirums metodi bir paytning o‘zida Min, Max va Position ga tegishli barcha xususiyatlar qiymatlarini aniqlab beradi.

TGroupBox



To‘g’ri burchakli ramka ko‘rinishidagi konteyner bo‘lib, u qandaydir bir interfeys elementlarining mantiqan bog’langan guruhini shaklda vizual birlashtiradi. Bu komponenta Windows ning bir nomdagi obyektning inkapsulyatsiyalanishidan iborat.

TRadioGroup

To‘g’ri burchakli ramka ko‘rinishidagi konteyner bo‘lib, u bir-birini mantiqan istisno qiladigan radio-tugmalar guruhini shaklda vizual birlashtiradi.

Radio-tugmalar bitta konteynerga joylashtirilganda «guruhlanadi». Bu guruhdan faqat bitta tugmacha tanlab olinishi mumkin. RadioGroup komponentasiga tugmalarni qo‘shish uchun, Items xususiyatining tahriri bajarilishi kerak. Items xususiyatining navbatdagi satriga nom berilsa, shu tugma guruhlovchi ramkada paydo bo‘ladi. Ushbu daqiqada qaysi tugma tanlab olinishi kerakligini ItemIndex xususiyatining qiymati aniqlab beradi. Columns xususiyatining tegishli qiymatini joylashtirib, siz radiotugmalarni bir necha ustunga guruhlashingiz mumkin.

TPanel

Boshqa komponentlarni o‘z ichiga olishi mumkin bo‘lgan bo‘sh Panelni yaratadi. Siz TPanel dan o‘z shaklingizda Instrumentlar Paneli yoki holatlar satrlarini yaratish uchun foydalanishingiz mumkin.

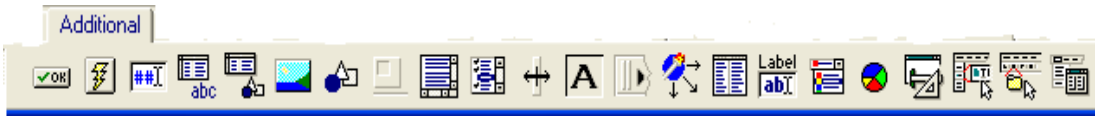
TPanel komponentasi TCustomPanel sinfining hosilasi bo‘lib, uning barcha xususiyatlar, metodlari va voqealari to‘liq vorislik qiladi.

Nazorat savollari

1. Borland C++ Builderning dastur yaratish muhiti
2. Borland C++ Builderning komponentlari tavsifi
3. Borland C++ Builderning komponentlari xossalari
4. File, Edit va View komponentlari tavsifi
5. Component, Database va Tools komponentlari tavsifi

4.4. Borland C++ Builder Additional komponentlar palitrasi

Komponentalar palitrasining **Additional** qo‘shimcha ilovalar komponentalari sizning dasturingizga Borland korporatsiyasi maxsus C++ Builder muhiti uchun ishlab chiqqan 9 ta boshqarish elementini kiritadi. Bit obrazining tasviri tushirilgan tugmachani yaratadi. Bunday tugmachalar ko‘proq maxsus dialogli darchalarda qo‘llanadi. Grafik tugmachalar bit obrazlari, ularning ko‘rinishi va tugmachada joylashishini spetsifikatsiyalash uchun xususiyatlarga ega bo‘ladi. Siz C++ Builder qurilmasiga kirgan alohida tasvirlar katalogidagi grafik tugmalarning tayyor stillaridan foydalanishingiz ham, yoki bo‘lmasa, tasvirlarni tahrir qilish tizimlarining biri tomonidan yaratilgan suratlardan foydalanishingiz ham mumkin. Tugmaning turli holatlariga (masalan «bosilgan», «qo‘yib yuborilgan», «ta’qiqlangan» va h.k.) turli bit obrazlari mos kelishi mumkin. Tasvirlar Fayllari Muharririning bmp kengaytmali darchasi Glyph xususiyatining qiymatlari tugmasi bilan ochiladi. BMP kengaytmali bit obrazlari fayllari tasvirlarining muharriri King xususiyati sizga yozuvlar va tegishli grafika (OK, Cancel, Hello va boshqalar) bilan ta’minlangan standartlashtirilgan tugmalarni yaratish imkonini beradi. TSpeedButton Odatda ma’lum menyu komandalarini tez chaqirish yoki rejimlarni o‘rnatish Paneli (TPanel) da joylashtiriladigan grafik tugmani yaratadi. Tezkor tugmaning turli holatlariga (masalan «bosilgan», «qo‘yib yuborilgan», «ta’qiqlangan» va h.k.) turli bit obrazlari mos kelishi mumkin. Bir-birining o‘rnini bosadigan tasvirlar va yozuv matnini tanlash uchun xususiyatlar mavjud. Kengayishli tasvirlar fayllari muharririning darchasi Gliph xususiyati qiymtlarining grafasidagi tugma bilan ochiladi. Tez tugmalarning boshqa xususiyatlari ularning biron-bir guruhdagi ishini tashkil etadi.



4.31-rasm. Additional komponentalari

Komponentalar palitrasining Additional qo‘shimcha ilovalar komponentalari sizning dasturingizga Borland korporatsiyasi maxsus C++ Builder muhiti uchun ishlab chiqqan 9 ta boshqarish elementini kiritadi.

TBitBtn

Bit obrazining tasviri tushirilgan tugmachani yaratadi. Bunday tugmachalar ko‘proq maxsus dialogli darchalarda qo‘llanadi.

Grafik tugmachalar bit obrazlari, ularning ko‘rinishi va tugmachada joylashishini spetsifikatsiyalash uchun xususiyatlarga ega bo‘ladi. Siz C++ Builder qurilmasiga kirgan alohida tasvirlar katalogidagi grafik tugmalarning tayyor stillaridan foydalanishingiz ham, yoki bo‘lmasa, tasvirlarni tahrir qilish tizimlarining biri tomonidan yaratilgan suratlardan foydalanishingiz ham mumkin.

Tugmaning turli holatlariga (masalan «bosilgan», «qo‘yib yuborilgan», «ta’qiqlangan» va h.k.) turli bit obrazlari mos kelishi mumkin.

Tasvirlar Fayllari Muharririning bmp kengaytmali darchasi (4.32-rasm) Glyph xususiyatining qiymatlari tugmasi bilan ochiladi.

4.32-rasm. BMP kengaytmali bit obrazlari fayllari tasvirlarining muharriri

King xususiyati sizga yozuvlar va tegishli grafika (OK, Cancel, Hello va boshqalar) bilan ta’minlangan standartlashtirilgan tugmalarni yaratish imkonini beradi.

TSpeedButton

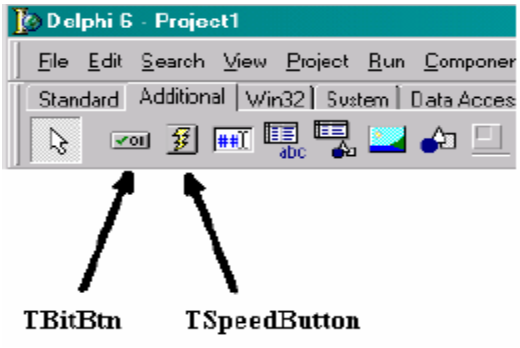
Odatda ma’lum menyu komandalarini tez chaqirish yoki rejimlarni o‘rnatish Paneli (TPanel) da joylashtiriladigan grafik tugmani yaratadi.

Tezkor tugmaning turli holatlariga (masalan «bosilgan», «qo‘yib yuborilgan», ta’qiqlangan» va h.k.) turli bit obrazlari mos kelishi mumkin. Bir-birining o‘rnini bosadigan tasvirlar va yozuv matnini tanlash uchun xususiyatlar mavjud. Kengayishli tasvirlar fayllari muharririning darchasi Gliph xususiyati qiymtlarining grafasidagi tugma bilan ochiladi. Tez tugmalarning boshqa xususiyatlari ularning biron-bir guruhdagi ishini tashkil etadi.

TSpeedButton va TBitBtn tugmalari

Bu tugmalar ***TButton*** vazifalarini bajaradi. Yagona farqi matndan tashqari racmlarni ham aks ettiradi. ***TSpeedButton*** tugmasi fokus olmaydi. Bu shuni bildiradiki, agar matn qatorida satr terib, bu tugma bosilsa, shu hodisa qayta ishlangandan so‘ng fokus yana matn qatoriga qaytib keladi. TAB tugmasi bilan bu tugmani ajratib bo‘lmaydi.

Tugmaga rasm o‘rnatish uchun ikki marta *Glyph* xossasi qatorida chertish lozim. Natijada rasm paydo bo‘lgan yuklash oynasida *Load* tugmasini bosish lozim. Ko‘p rasmlar ***Program Files\Common Files\Borland Shared\Images \Buttons*** katalogida joylashgandir.



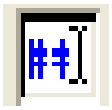
TBitBtn va ***TSpeedButton*** tugmalari deyarli bir xil xossalarga egadir. Ular uchun umumiy *Layout*, xossasi rasm va matn o‘zaro joylashuvini o‘zgartirishga imkon beradi. Quyidagi rasmda har xil qiymatlar mos variantlari ko‘rsatilgan.

TBitBtn tugmasining yana bir xossasi *Kind* bo‘lib oldindan tayyorlangan standart tugmalarni tanlash imkonini beradi. Quyidagi rasmda standart tugmalar va ularga mos qiymatlarni ko‘rish mumkin.

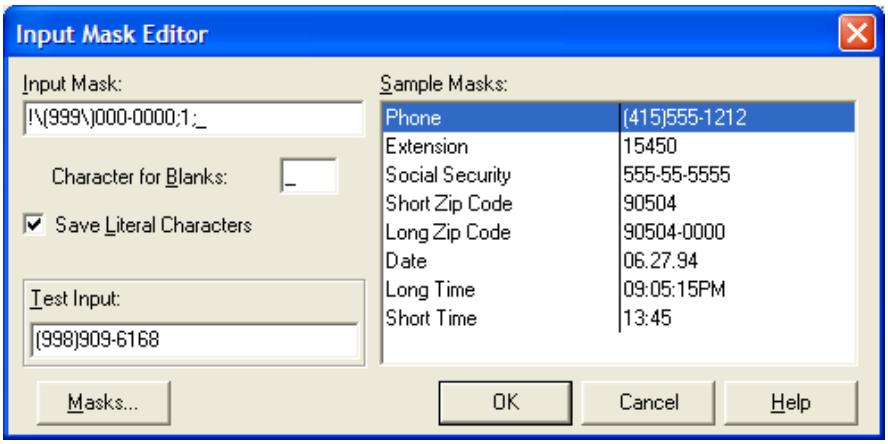
Yana bir xossa ModalResult – dialog oynasi uchun tugma qaytaradigan natijani tanlashga imkon beradi.

TSpeedButton tugmasining *GroupIndex* xossasi tugmalarni guruhlashga imkon beradi. Buning uchun bir guruhga tegishli tugmalarning *GroupIndex* xossasi bir xil qiymatga masalan 1 ga teng bo‘lishi kerak. Guruhlangan tugmalarning biri bosilsa, qolganlaridan ajralib qoladi. Buning uchun *Down* xossasi qiymati *true* ga teng bo‘lishi kerak.

TmaskEdit



O‘ziga xos formatdagi ma’lumotlarning tahrir qilinadigan nazoratdagi to‘rtburchak sohasini yaratadi. Kiritilayotgan matnning to‘g’riligi ruxsat etilgan formatlarni kodlovchi niqob vositasida tekshiriladi. Bu formatlarga matn kiritilgan va foydalanuvchiga taqdim etilgan bo‘lishi mumkin (sana, vaqt, telefon raqami va h.k.). EditMask xususiyati joriy niqob kodini saqlaydi. Niqoblar muharriri darchasi (4.33-rasm) ushbu xususiyat qiymatlari grafasida tugma bilan ochiladi.



4.33-rasm. Telefon raqamlarini kiritish uchun niqobni yaratish

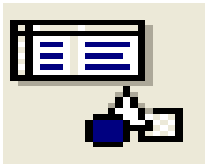
TMaskEdit komponentasi TCustomMaskEdit sinfining to‘g’ridan-to‘g’ri hosilasidir. U satrlar yoki ustunlar bo‘yicha belgili ketma-ketliklarni aks ettirish uchun mo‘ljallangan muntazam (regulyar) to‘rni yaratadi.

TStringGrid

Ushbu komponentaga tegishli barcha xususiyatlarning nomlari va vazifalari bo‘lib, siz ulardan dasturni loyihalash bosqichida to‘la foydalanishingiz mumkin. Ular keyingi paragrafda tavsifi berilgan TDrawGrid komponentasi xususiyatlariga to‘liq to‘g’ri keladi.

Simvolli ketma-ketliklar bilan bog’liq barcha obyektlar kerakli obyektga murojaat qilish imkonini beradigan Objects xususiyatida mujassam bo‘lgan. Dastur bajarilish paytida simvolli ketma-ketliklar va setka ustunining ular bilan bog’liq obyektlari Cols xususiyati bilan adreslanadi. Rows xususiyati setka satrlari bilan xuddi shunday ish to‘tish imkonini beradi. Setkaning barcha simvolli ketma-ketliklari setkaning kerakli uyasini adreslaydigan (manzillaydigan) Cells xususiyatida mujassamdir.

TDrawGrid



To‘zilma holiga keltirilgan grafik ma’lumotlarni satrlar yoki ustunlar bo‘yicha aks ettirish uchun muntazam setka yaratadi. RowCount va ColCount xususiyatlari 125ertical bo‘yicha va gorizontal bo‘yicha setka uyalarining sonini belgilaydi.

Options xususiyatining qiymatlari setkaning turi (masalan, ustunlar orasida ajratuvchi chiziqlarga ega bo‘lgan setka turi) va uning xulq-atvorini (masalan, ustundan ustunga Tab klavishasi bo‘ylab o‘tish) o‘zgartirish imkonini beradi. Setkadagi ajratish chiziqlarining eng GridLineWidth xususiyatli tomonidan belgilanadi, aylantirish chiziqchalari esa ScrollBars xususiyati tomonidan qo‘shiladi. FixedCold va FixedRows xususiyatlari ustunlar va satrlarning aylantirilishini ta’qiqlab qo‘yish imkonini beradi, Fixed Color xususiyati esa barcha usutn va satrlarga ma’lum rang beradi.

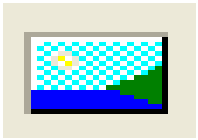
DefaultDrawing xususiyatining true qiymati setka uyalarining ichidagilarini avtomatik tarzda chizib ko‘rsatadi, bunda uning foni, asosi va rangi yashirin tanlanadi. Default Drawing xususiyatining false qiymatini o‘rnatish uchun, setka uyalarini «qo‘lda» to‘ldirish uchun mo‘ljallangan OnDrawCell voqeasi qayta ishlatgichining yozilishini talab qiladi. DefaultColWidths va DefaultRowHeights xususiyatlari yordamida yashirin tanlanayotgan barcha ustunlar va satrlarning enini o‘rnatish mumkin. ColWidth va RowHeight xususiyatlari konkret ustun enini va konkret satr bo‘yini spetsifikatsiyalaydi.

Dasturning ishlash paytida siz CellRest metodi yordamida biron-bir uyaning rasmini chizish uchun ma’lum sohani o‘z ixtiyoringizga olishingiz mumkin.

MouseToCell metodi ustun raqami va sichqoncha kursori o‘rnatilgan satr uyasining koordinatalarini qaytarib beradi. Setkaning tanlab olingan uyasi Selection xususiyatining qiymati bo‘lib qoladi.

Dastur bajarilish paytida qaysi satr setkaning ustki satri bo‘lishini aniqlash yoki TopRow xususiyati yordamida ko‘rsatilgan satrni ustki holatga qo‘yib qo‘yish mumkin. Qaysi ustun setkaning ko‘rinib turadigan usutni bo‘lishini aniqlash uchun, LeftCol xususiyatidan foydalaning. VisibleColCount va VisibleRowCount xususiyatlarining qiymatlari setkaning ko‘rinib turgan ustunlari va satrlarining umumiy sonini spetsifikatsiyalaydi.

TImage



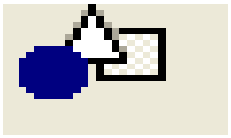
Shaklda grafik tasvir konteynerini yaratadi (bu bit obrazi, piktogarmma yoki metafayla bo‘lishi mumkin).

Tasvirlar fayllari muharririning darchasi Picture xususiyati qiymatlari grafasidagi tugma bilan ochiladi. Konteyner o‘z o‘lchamlarini tasvirni to‘liq sig’diradigan qilib o‘zgartirishi uchun, AutoSize xususiyatining true qiymatini o‘rnating. Kichikroq o‘lchamdagi dastlabki tasvir butun konteynerga cho‘zilib ketishi uchun, Stretch xususiyatining true qiymatini o‘rnating.

Tasvirlar fayllarining dinamik yuklanishi va saqlanishi uchun, Picture obyekt xususiyatining LoadFromFile va SaveToFile metodlaridan qo‘yidagi turlar yordamida foydalaning:

Image->Picture->LoadFromFile(«<fayl nomi>»); Image-> Picture ->SaveToFile(«<fayl nomi>»);

TShape



Aylana va ellips, kvadrat va to‘g’ri to‘rtburchak (burchaklarini yumaloqlash mumkin) kabi oddiy geometrik shakllarning rasmini chizadi.

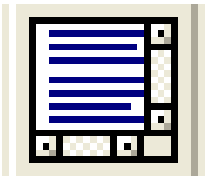
Tanlab olingan geometrik shaklning turini Shape xususiyati, rang va bo‘yash usulini Brush komponentasiga joylangan ikkita Color va Style xususiyatlari aniqlaydi. Shakllarning o‘lchamlarini ham tegishli xususiyatlar aniqlaydi.

TBevel

Xuddi uskuna bilan o‘yilgandek xajmli ko‘rinadigan chiziqlar, box lar yoki ramkalarni yaratadi.

Komponenta chizayotgan obyektni Shape xususiyati aniqlaydi, Style xususiyatining qiymati esa obyekt ko‘rinishini o‘zgartirib, uni bo‘rtiq yoki botiq holga keltiradi. Foydalanuvchi shakl o‘lchamlarini o‘zgartirganda ham obyektning nisbiy holatini o‘zgarmas qoldirish uchun, Align xususiyatining true qiymatini o‘rnating.

TScrollBox



Darchada o‘lchamlari o‘zgaruvchan box ni yaratadi, bu box shu topdayoq avtomatik tarzda zaruratga ko‘ra aylantirish lineykalari bilan ta’minlanadi.

Aylantirib ko‘rish boksi yordamida darchaning ayrim sohalarini aylantirib ko‘rishdan himoyalash mumkin. Masalan, Instrumentlar paneli va holat panelini himoyalash uchun, avval darchani aylantirish lineykasini berkitib qo‘ying, keyin esa aylantirish boksiini mijoz sohasida Instrumentlar paneli va holat paneli o‘rtasida joylashtiring. Boksni aylantirib ko‘rish lineykasi darchaga tegishli bo‘lib ko‘rinadi, biroq aylantirish faqat boks ichida amalga oshiriladi.

Aylantirib ko‘rish bokslaridan yana boshqachasiga ham foydalanish mumkin: ular biron-bir darchada ko‘plab aylantirib ko‘rilayotgan sohalar (turlar) yaratish imkonini beradi. Turlar ko‘pincha tijoriy matn protsessorlarida, buxgalteriya dasturlarida va loyihalarni rejalashtirish dasturlarida qo‘llanadi. Aylantirib ko‘rish boksi boshqa komponentlarga, masalan TButton va TCheckBox ga ham ega bo‘lishi mumkin.

Nazorat savollari

1. Borland C++ Builder Additional komponentlar palitrasi
2. TBitBtn, TSpeedButton, TSpeedButton tugmalari tavsifi
3. TmaskEdit, TStringGrid, TDrawGrid tugmalari tavsifi
4. TImage, TShape, TBevel va TScrollBox tugmalari tavsifi

4.5. Borland C++ Builder dasturlash muhitida jarayonlarni dasturlash

Dastur ishlashi jarayonida xotiradan kamroq joy egallashi dasturning tez ishlashiga olib keladi. Bu muammolar dasturdagi o‘zgaruvchilar sonini kamaytirish, yoki o‘zgaruvchilar saqlanadigan yacheyka hajmini kamaytirish orqali erishiladi. *Borland C++ Builder* 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 (4.3-jadval).

4.3-jadval. O‘zgaruvchilarning turlari va oraliq qiymatlari

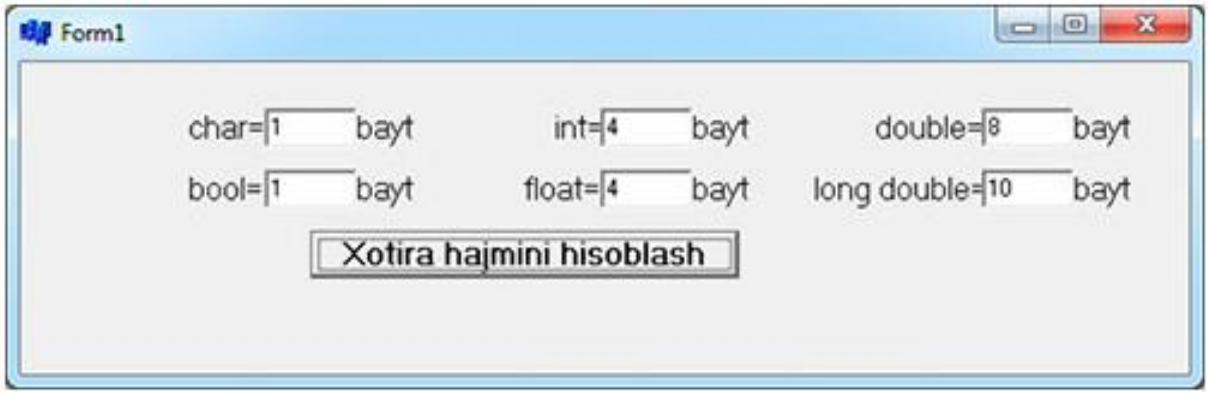
O‘zgaruvchi turi	Qiymatlar chegarasi
Butun sonlar	
unsigned short int	0..65535 2 bayt
short int	32768..32767 2 bayt
unsigned long int	0..42949667295 4 bayt
long int	2147483648..2147483647 4 bayt
int (16 razryadli)	32768..32767 2 bayt
int (32 razryadli)	2147483648..2147483647 4 bayt
unsigned int (16 razryadli)	0..65535 2 bayt
unsigned int (32 razryadli)	0..42949667295 4 bayt
Haqiqiy sonlar	
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
Mantiqiy ifoda	
bool	true yoki false 1 bayt
Belgilar	
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. *Borland C++ Builder* 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 egallagan xajmini aniqlash.

***1-misol**. Borland C++ Builderda turlar uchun xotira hajmini hisoblash dasturi.*

```
void __fastcall TForm1::Button1Click(TObject *Sender)
{
    Edit1->Text=sizeof(char);
    Edit2->Text=sizeof(bool);
    Edit4->Text=sizeof(int);
    Edit5->Text=sizeof(float);
    Edit6->Text=sizeof(double);
    Edit7->Text=sizeof(long double);
}
```

NATIJA:



Matemetik funksiyalardan dasturda foydalanish uchun **math.h** sarlavha faylini progarmmaga qo‘shish kerak: **#include <math.h>**

1. **abs(x)** - butun sonlar uchun /x/
2. **fabs(x)** - haqiqiy sonlar uchun
3. **labs(x)** - uzun butun son uchun
4. **Pow(x,y)**-x^y
5. **sqrt(X)**-ildiz(X)
6. **ceil(x)**-haqiqiy toifadagi x o‘zgaruvchisi qiymatini unga eng yaqin katta butun songa aylantiradi.
7. **floor(x)**-haqiqiy toifadagi x o‘zgaruvchisi qiymatini unga eng yaqin kichik butun songa aylantiradi.
8. **cos(x)**-x burchak kosinusini aniqlash. x radian o‘lchovida.
9. **sin(x)**-x burchak sinusini aniqlash. x radian o‘lchovida.
10. **exp(x)**-e^x

11. **log(x)**-x sonining natural logarifmini qaytaradi.
12. **log10(x)**-x sonining 10 asosli logarifmini qaytaradi.

2-misol. n va m natural sonlari berilgan. n sonini m soniga bo‘lib, qoldiqni aniqlovchi dastur tuzilsin.

Borland C++ Builder da dasturu:

```
void __fastcall TForm1::Button1Click(TObject *Sender)
{
      int a,n,m;
      n=StrToFloat(Edit1->Text);
      m=StrToFloat(Edit2->Text);
     a=n%m;
     Edit3->Text=FloatToStr(a);
}
```

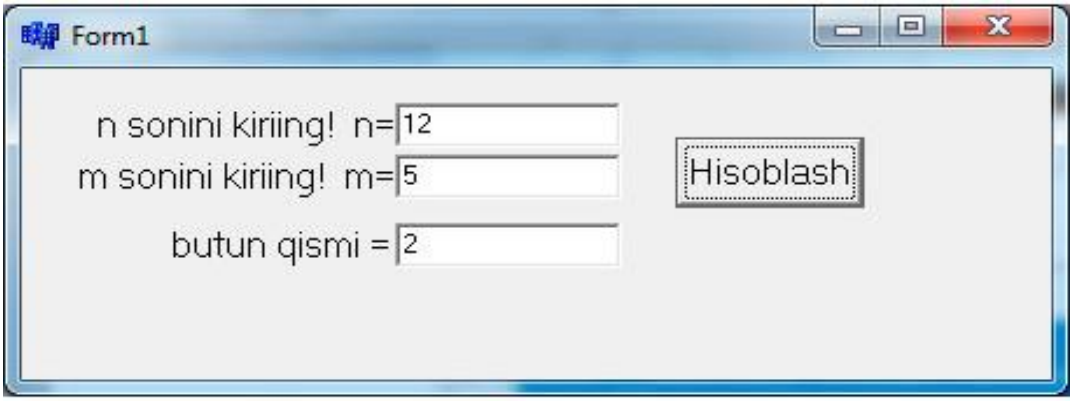
NATIJA:

3 – misol. n va m natural sonlari berilgan. n sonini m soniga bo‘lib, butun qismini aniqlovchi dastur tuzilsin.

Borland C++ Builder da dasturu:

```
void __fastcall TForm1::Button1Click(TObject *Sender)
{
      int a,n,m;
      n=StrToFloat(Edit1->Text);
      m=StrToFloat(Edit2->Text);
      a=n/m;
      Edit3->Text=FloatToStr(a);
}
```

NATIJA:

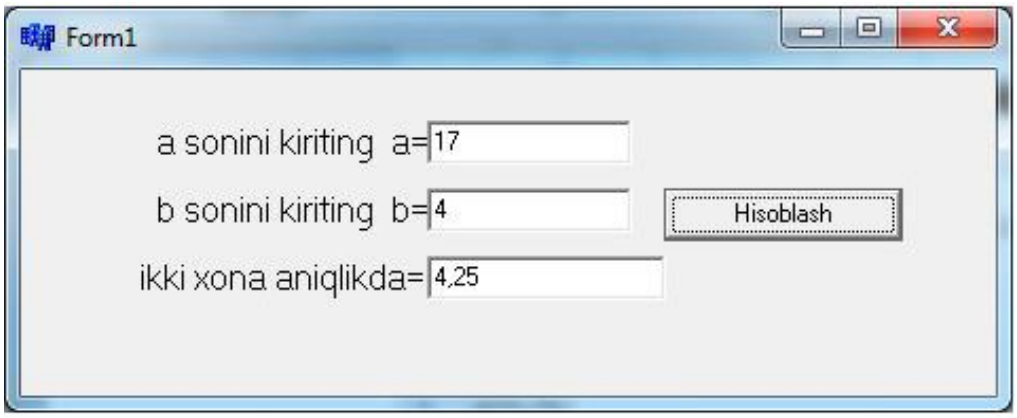


4 - misol. a sonini b soniga bo`lib 2 xona aniqlikda chiqarish.

Borland C++ Builder da dasturu:

```
void __fastcall TForm1::Button1Click(TObject *Sender)
{
    int a,n,m;
    n=StrToFloat(Edit1->Text);
    m=StrToFloat(Edit2->Text);
    a=n/m;
    Edit3->Text=FloatToStr(a);
}
```

NATIJA:

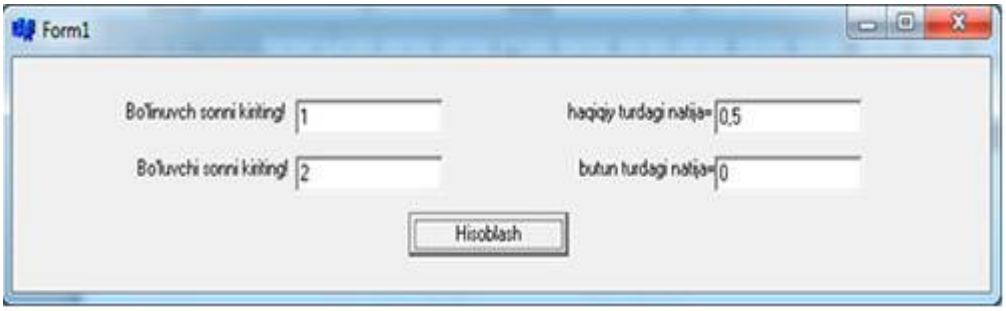


5 - misol. Butun sonni bo‘lish

Borland C++ Builder da dasturu:

```
void __fastcall TForm1::Button1Click(TObject *Sender)
{
     int a,b;
     a=StrToFloat(Edit1->Text);
     b=StrToFloat(Edit2->Text);
     Edit3->Text=FloatToStr((float(a))/(float(b)));
    Edit4->Text=FloatToStr(a/b);
}
```

NATIJA:



Borland C++ Builderda shartli operator

Shundan dasturlar mavjudki ularning shartiga qarab ikki xil natija qabul qilishi mumkin. Bu o`z navbatida dasturni tarmoqlanishga olib keladi. Tarmoqlarning qaysi qismi bajarilishi ayrim shartlarga qarab aniqlanadi. Shart operatori: Shart operatori boshqarishni qaysi tarmoqqa uzatishni ta'minlaydi.Shart operatorining ikki xil ko`rinishi mavjud. Operatorning umumiy ko`rinishi va qisqa ko`rinishi.

Shart operatorining umumiy ko`rinishi:

if *(<shart>)<operator1>;*
else *<operator2>****;***

if agar, else aks holda ma`nolarini anglatadi.

Shart operatorining qisqa ko`rinishi:

if (<shart>) <operator1>;

<shart> tekshirilishi lozim bo`lgan mantiqiy ifoda <operator 1> Agar shart rost (true) qiymatga ega bo`lsa bajarilishi lozim bo`lgan operator.

<operator 2> Agar shart yolg`on (false) qiymatga ega bo`lsa bajarilishi lozim bo`lgan operator.

Shart operatori tarkibida ixtiyoriy operatordan foydalanish mumkin. Shu o`rinda Shart operatoridan ham.

Misol: Berilgan a sonini juft yoki toqligini aniqlovchi dastur tuzilsin. Agar a sonini 2 ga bo‘lganda qoldiq 0 ga teng bo‘lsa, bu son juft, aks xolda toq.

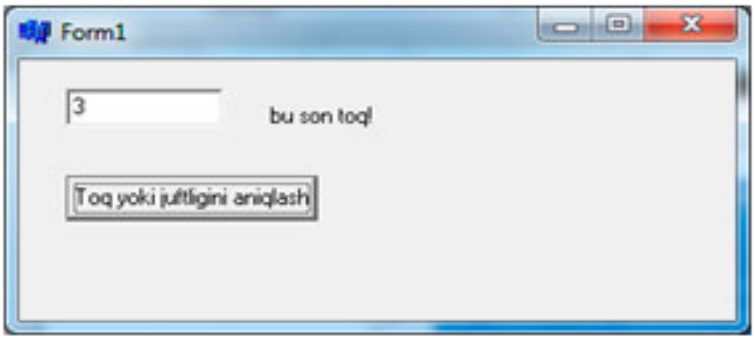
Borland C++ Builder tili operatorlarni blok ko‘ rinishida bo‘ lishiga imkon beradi. Blok ‘{‘ **va** ‘}’ belgi oralig’iga olingan operatorlar ketma-ketligi bo‘ lib, u kompilyator tomonidan yaxlit bir operator deb qabul qilinadi. Blok ichida yangi o‘zgaruvchilarni ham e’lon qilish mumkin. Bu o‘zgaruvchilar faqat blok ichida ko‘rinadi, undan tashqarida ko‘rinmaydi, ya’ni blokdan tashqarida bu o‘zgaruvchilarni ishlatib bo‘lmaydi. Blokdan keyin nuqtali vergul qo‘yilmaydi, lekin blok ichida har bir operator nuqtali vergul bilan yakunlanishi shart. Shart operatorida bir nechta operatordan foydalanish uchun bu operatorlarni blok ichiga yozish lozim bo‘ladi. Yuqoridagi masalani blok orqali ifodalash quyidagicha bo‘ladi.

6-misol. Berilgan a sonini juft yoki toqligini aniqlovchi dastur tuzilsin.

Borland C++ Builder da dasturu:

```
void __fastcall TForm1::Button1Click(TObject *Sender)
{
     int a;
     a=StrToFloat(Edit1->Text);
     if (a%2==0) Label1->Caption=("bu son juft!");
    else Label1->Caption=("bu son toq!");
}
```

NATIJA:



Dasturlashning yaxshi usuli: Shart operatorida doimiy ravishda bloklardan foydalanish yo‘l qo‘yilishi mumkin bo‘lgan xatoliklarni oldini oladi.

Ba’zi dasturchilar oldin ochuvchi va yopuvchi qavslarni {, } yozish, undan keyin blok ichidagi operatorlarni yozish lozimligini takidlashadi. **? :** 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>;

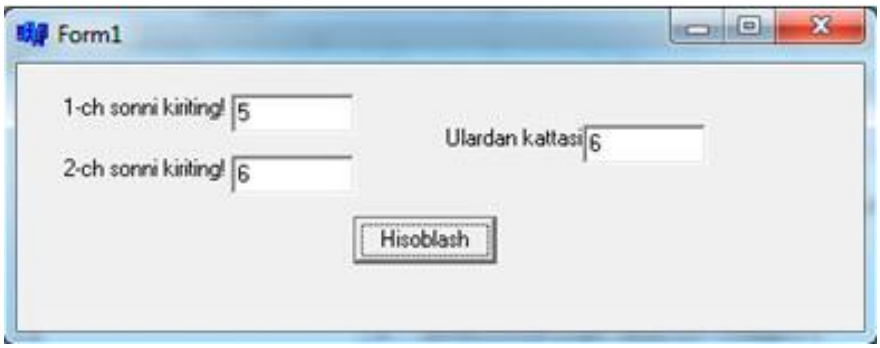
if shart operatoriga o‘xshash holda bu shart amali quyidagicha ishlaydi: agar **<shart ifoda> rost (true)** bo‘lsa **<ifoda1>** bajariladi, aks holda **<ifoda2>**. Odatda ifodalar qiymatlari birorta o‘zgaruvchiga o‘zlashtiriladi.

7-misol. 2 ta sondan kattasini topuvchi dastur tuzilsin.

Borland C++ Builder da dasturu:

```
void __fastcall TForm1::Button1Click(TObject *Sender)
{
    int a,b,c;
    a=StrToFloat(Edit1->Text);
    b=StrToFloat(Edit2->Text);
    c=(a>b)?a:b;
    Edit3->Text=FloatToStr(c);
}
```

NATIJA:



8-misol. Agar a>b shart bajarilsa max o‘zgaruvchisi a ni, aks xolda b ni o‘zlashtiradi. Tanlash operatorida bir nechta qiymatga bir hil operator ishlatishi quyidagicha bo‘ladi.

Borland C++ Builder da dasturu:

```
void __fastcall TForm1::Button1Click(TObject *Sender)
{
   int a;
   a=StrToFloat(Edit1->Text);
switch(a);
  {
      case1:
      case3:
      case5:
      case7:
     case9:
           Label2->Caption=("Kiritilgan son toq");
     case2:
     case4:
     case6:
     case8:
     case10:
          Label2->Caption=("Kiritilgan son juft");
  default : Label2->Caption=(1 dan kichik yoki 10 dan katta son kiritilgan!);
  }
}
```

NATIJA:

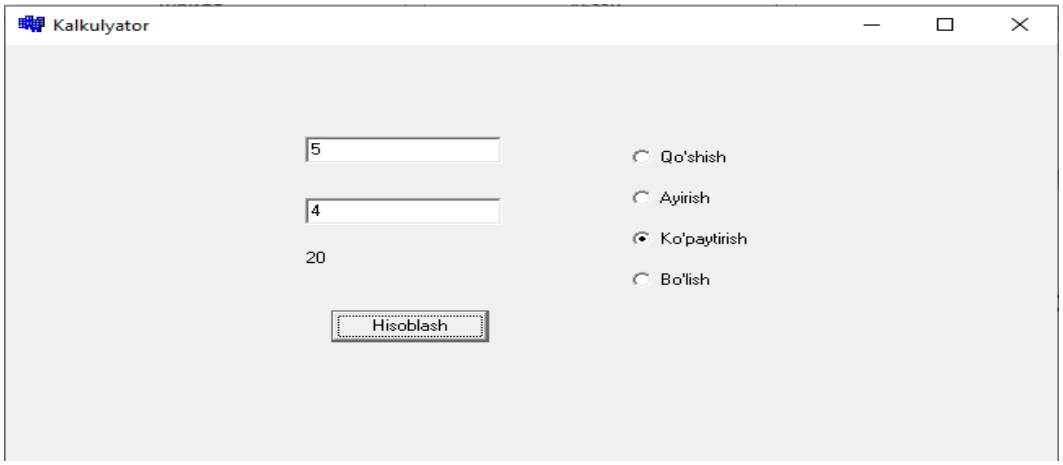
9-misol. Borland C++ Builderda “kalkulyator” yasash.

Borland C++ Builder da dasturu:

```
void __fastcall TForm1::Button1Click(TObject *Sender)
{
 if (RadioButton1->Checked==true)
```

```
 Label1->Caption=FloatToStr(StrToFloat(Edit1->Text)+StrToFloat(Edit2->Text));
 if (RadioButton2->Checked==true)
 Label1->Caption=FloatToStr(StrToFloat(Edit1->Text)-StrToFloat(Edit2->Text));
 if (RadioButton3->Checked==true)
 Label1->Caption=FloatToStr(StrToFloat(Edit1->Text)*StrToFloat(Edit2->Text));
 if (RadioButton4->Checked==true)
 Label1->Caption=FloatToStr(StrToFloat(Edit1->Text)/StrToFloat(Edit2->Text));
}
```

NATIJA:



Nazorat savollari

1. Borland C++ Builder dasturlash muhitida jarayonlarni dasturlash
2. Borland C++ Builderda o‘zgaruvchilarning turlari va oraliq qiymatlari
3. Borland C++ Builder sizeof funksiyasi tavsifi
4. Matemetik funksiyalardan dasturda foydalanish uchun math.h sarlavha fayli
5. Borland C++ Builderda matemetik funksiyalar
6. Borland C++ Builderda turlar uchun xotira hajmini hisoblash dasturi
7. Borland C++ Builderda tarmoqlanuvchi jarayonlarni dasturlash
8. Borland C++ Builderda takrorlanuvchi jarayonlarni dasturlash

TEST TOPSHIRIQLARI

1. Mantiqiy mulohazalar ustida nechta amal aniqlangan?
   A. 3
   B. 4
   C. 2
   D. amal aniqlanmagan.

2. Qandaydir shartni rostlikka tekshirish natijasiga ko‘ra programmada takrorlanishni amalga oshiradigan operator qaysi?
   A. break
   B. if
   C. for
   D. while

3.Satrlarni ulash uchun qaysi operatordan foydalanamiz?
   A. strcpy()
   B. strcmp()
   C. strcat() va strncat()
   D. strrev()

4.do – while operatorining sintaksisi to‘g’ri berilgan qatorni toping ?
   A. do { <sikl tanasi> ; } while(<shart>) ;
   B. do { <sikl tanasi> } while(<shart>) ;
   C. do { < operator> ; } while(<shart>) ;
   D. do { <sikl tanasi> ; while(<shart>) ; }

5.C++ da x ning absolut qiymati qanday yoziladi ?
   A. fab(x)
   B. abs(x)
   C. abs x
   D. fab x

6. Identifikator sifatida foydalanish mumkin bo‘lgan javobni ko‘rsating ?
   A. abs
   B. number
   C. cout
   D. include

7. C++ tilida quyidagi operatorlar ketma-ketligi bajarilishi natijasida ekranga nima chiqadi ? *int a=4,b=2; cout<<--a+b;*
   A. 6
   B. -2

C. 2
D. 5

8.C++ tilida belgi turidagi o‘zgaruvchilar qanday e’lon qilinadi ?
A. char
B. bool
C. int
D. float

9.C++ da funksiya qiymat qaytarishda qaysi kalit so‘zi ishlatiladi ?
A. return
B. cout
C. cin
D. getline

10. Qaysi qatorda satrga izoh to‘g’ri berilgan ?
A. //izoh
B. /*izoh
C. /izoh
D. /*izoh /

11.C++ da satrlar bilan ishlash uchun qaysi kutubxonadan foydalanamiz?
A. stdio.h
B. string.h
C. math.h
D. iostream.h

12.Ikki o‘lchamli massivning sintaksisi to‘g’ri berilgan qatorni toping?
A. <tur><nom>[<uzunlik>]
B. <tur><nom>{<uzunlik>}{<uzunlik>};
C. <tur><nom>[<uzunlik>][<uzunlik>];
D. <tur><nom [<uzunlik>];

13.Lokal o‘zgaruvchilar - …
A. Funksiya tashqarisida e’lon qilingan o‘zgaruvchilar
B. Dastur boshida e’lon qilingan o‘zgaruvchilar
C. Funksiya ichida e’lon qilingan o‘zgaruvchilar
D. Butun o‘zgaruvchilar

14.Bir yoki har xil turdagi belgilarni jamlanmasi - …
A. binar fayllar
B. struktura
C. massiv

D. dinamik massiv

15.C++ da mantiqiy inkor ,qo‘shish va ko‘paytirish amallari mos ravishda to‘g’ri berilgan qatorni toping?

A. ! , || , &&
B. && , ! , ||
C. ! , && , ||
D. || , && , !

16.C++ tilida ixtiyoriy turni o‘lchamini aniqlaydigan amal qaysi?

A. sizeof()
B. size()
C. strln()
D. bunday amal mavjud emas

17.for takrorlash operatoridan to‘g’ri foydalanilgan qatorni toping.

A. for(int i=7, i>10, i++)
B. for(i=2; i<15;)
C. for(int i=10; i<0; i--)
D. for(int i=1;i<=8,++i)

18.Takrorlash operatori tanasining ixtiyoriy joylariga qo‘yish orqali shu joylardan takrorlashdan chiqishni amalga oshiradigan operator

A. break
B. continue
C. go to
D. while

19.Satrni teskari tartiblash uchun qaysi funksiyadan foydalaniladi?

A. strcat()
B. strcpy()
C. strcmp()
D. strrev()

20.Malumotlarni kiritish va natijalarni chiqarish uchun qaysi shakl ishlatiladi?

A. 
B. 
C. 
D. 

21.printf va scanf funksiyalari qaysi kutubxonaga tegishli?

A. stdio.h
B. fstream.h
C. cmath
D. iostream

22.Postfix-inkrement to‘g’ri yozilgan qatorni toping.

A. ++a
B. a++
C. --a
D. a--

23. x^{x^x}·lnx ni C++ tiliga o‘giring.

A. pow (pow (x , x) , x) * log x
B. pow(pow(x,x))*log(x)
C. pow (x, pow (x , x)) * log(x)
D. pow (pow (x , x) , x)

24.Uzunligi ko‘rsatilgan massivni to‘liq initsializatsiyalashga doir misol qaysi qatorda to‘g’ri berilgan?

A. float a[3]={ 1.2 , 3 , 5.6}
B. int b[5]={2, 3}
C. int a[ ]={7, 3, 8, 2}
D. double b[4]

25. Kompilyator - bu:

A. Kiritilgan dasturning ayrim qismlaridan boshqa dastur tuzuvchi translyator
B. Kiritilgan dastur matnini to‘lig’icha mashina tiliga o‘giruvchi translyator
C. Foydalanuvchining dasturini mashina tiliga o‘girib beruvchi qurilma
D. Kiritilgan dasturni qismlari bo‘yicha mashina tiliga o‘giruvchi translyator

26. C++ da kiritish operatorini ko‘rsating

A.cin
B.cout
C.read
D. write

27. C++ tilidagi xizmatchi so`zlarning to`g`risini toping ?

A. for, begin, end
B. if, case, end
C. Case, with, to
D. for, if, break

28. y=$e^{-x \cdot cosx}$ ni C++ tiliga o‘giring.

A. y= exp((x)*cos(x))
B. y=exp((-x)*cos(x))
C. y=exp((-x) cos(x))
D. y= exp((-x*cos(x))

29. … - takrorlash operatori dastur tanasini bajarishni to‘xtatadi, lekin takrorlashdan chiqib ketmasdan sakrab o‘tadi.

A. for
B. continue
C. do … while
D. while

30. C++ tilining kalit so‘zlari berilmagan qatorni toping.

A. enum, void, case
B. float, return, char
C. new , try, sizeof
D. read ,write, statik

31. Sikl operatorining qaysi turlarida sikl tanasi bajarilmasligi mumkin?

A. while()
B. for()
C. do … while()
D. for() va while()

32. Qiymat qaytarmaydigan funksiyani toping ?

A.void
B.break
C.return
D.rekursiya

33.Quyidagi programma qismi nima natija chiqaradi?

```
int a=17;
cout<<a%10<<”w”<<a/10;
```

A. 7w1
B. 17w1
C. 6w8
D. Dastur ishlamaydi.

34.Project bo‘limi - …

A. Dasturni komplatsiya qilish buyruqlaridan iborat
B. Muhitda ishlash bo‘yicha kerakli ma’lumotlarni olish mumkin
C. Loyihalar bilan ishlash buyruqlaridan iborat
D. Dasturda mavjud xatoliklarni topishni yengillashtiruvchi buyruqlardan iborat

35.C++ tili alfavitiga kirmaydigan belgilar qatorini toping.

A. € · ÷ a
B. [] , A .:
C. .: _ 3 ||
D. / & 0

36. Dasturning tashqi ko‘rinishini shakllantirish uchun qaysi kutubxonadan foydalanamiz?

A. string.h
B. conio.h
C. time.h
D. stdlib.h

37.O‘z-o‘ziga murojaat qiluvchi funksiya bu - …

A. rekursiv funksiya
B. bazaviy funksiya
C. global funksiya
D. bosh funksiya

38.To‘g’ri yozilgan operatorni toping. *Agar a≤b va c<3 bo‘lsa , u holda a=c+b bo‘lsin.*

A. if (a<b && c<3) { a=c+b; }
B. if (a<=b && c<3) { a=a+b; }
C. if (a<=b && c<3) { a=c+b; }
D. if (a<=b && c<3) { a=c+b }

39.x ga eng yaqin kichik butun sonni chiqaruvchi funksiya qaysi?

A. ceil(x)
B. hypot(x)
C. sqrt(x)
D. floor(x)

40. #*include* nima vazifani bajaradi?

A. Bir manba fayl ichida boshqa bir faylga murojaatni amalga oshirish mexanizmi
B. Makro deriktivalarni belgilash
C. Dastur kodini o‘zida jamlovchi fayl
D. To‘g’ javob yo‘q

41.C dasturlash tilida funksiya …….. orqali faollashtiriladi?

A. Funksiyani tavsiflash
B. Funksiyaga murojaat
C. Funksiyani o‘zgarishi
D. Funksiyani aniqlash

42. 

Blok – sxemasi qanday ma’noga ega?·

A. Siklik jarayon
B. Hisoblashlar bloki
C. Birlashtirish
D. Shartli blok

43. Save as - …….

A. Faylni saqlash
B. Dasturlash muhitida ishni tamomlash
C. Muharrirlanayotgan faylni boshqa nom bilan saqlash
D. TJY

44.*bool* tipiga mansub o‘zgarmaslar berilgan qatorni toping?

A. 1.2 ; 3;
B. true ; false
C. ‘k’ , ‘i’ , ‘t’ , ‘o‘ , ‘b’
D. 1, 2, 3, 4,

45. *y=ceil (6.9)-floor(6.2)* natijani toping.

A. 1
B. 0
C. 2
D. -0.7

46. Tanlash operatori berilgan qatorni toping

A. switch()

B. goto()
C. if()
D. continue

47. $y = \log_3 x^2 + \sqrt{x^3}$ ni C tiliga o‘giring.
A. log(x*x)/log(3)+sqrt pow(x,3);
B. log(3 , x*x) +sqrt(pow(x,3));
C. log(3 , x*x) +sqrt pow(x,3);
D. log(x*x)/log(3)+sqrt(pow(x,3));

48. postfiks operatori qachon bajariladi?
A. Qiymat o‘zlashtirilguncha
B. Qiymat o‘zlashtirilgandan keyin
C. Farqi yo‘q
D. TJY

49. Ishorasiz ma’lumotlarni tavsiflashning ma’lumotlar turi - ….
A. unsigned
B. int
C. float
D. double

50.y=ln 5x +$x^5 \cdot e^{2x+1}$ ni C tiliga o‘giring.
A. y=ln(5*x)+pow(x,5)*exp(2*x+1)
B. y=log(5x)+pow(x,5) exp(2x+1)
C. y=log(5*x)+pow(x,5)*exp(2*x+1)
D. y=log (5*x)+pow(x,5)*exp(2x+1)

51.C++ da qiymat berish operatori qaysi qatorda to‘gri berilgan?
A. /
B. +
C. –
D. =

52.C++ da qiymatlarni birga oshirish qaysi operatorning vazifasi?
A. Inkrement
B. Dekrement
C. Identifikator
D. Cout

53.Tarmoqlanuvchi amalni bajarish uchun qaysi operatordan foydalaniladi?

A. Taqqoslash operatori
B. Tayinlash operatori
C. Mantiqiy operator
D. Shartli operator

54. C ++ da matematik funksiyalar bilan ishlash uchun qaysi kutubxonadan foydalaniladi?

A. string.h
B. math.h
C. stdio.h
D. iostream.h

55. Tsikl nima?

A. Forscha <aylana>
B. Arabcha <to‘rtburchak>
C. Inglizcha <uchburchak>
D. Yunoncha <doira>

56.C++ da tsikl operatorlarining nechta turi mavjud?

A. 3ta
B. 4ta
C. 5 ta
D. 2ta

57.C++ tilida k ning ildizi qanday yoziladi?

A. abs(k)
B. sqrt k
C. sqrt(k)
D. fab(k)

58. C++ da haqiqiy son turidagi o‘zgaruvchilar qanday e’lon qilinadi?

A. double
B. float
C. int
D. bool

59. Ko‘rsatkichlar necha xil turga bo‘linadi?

A. 3
B. 7
C. 5
D. Hech qanday turga bo‘linmaydi

60.Massiv elementlarining indekslari nechadan boshlanadi?

A. 0
B. 1
C. 2
D. 3

61.Massivlar nechaga bo‘linadi?

A. 3
B. 4
C. 5
D. 2

62. fstrem kutubxonasida nechta obyekt mavjud?

A. 3
B. 4
C. 5
D. 2

63.Ko‘rsatkichlarga boshlang’ich maxsus NULL qiymat berilsa bunday ko‘rsatkich …… deyiladi?

A. Obyektga yo‘naltirilgan ko‘rsatkich
B. Bosh ko‘rsatkich
C. Void ko‘rsatkich
D. Funksiya

64.Ikki o‘lchovli massiv to‘g’ri berilgan qatorni toping?

A. a[2]{2}
B. a{3}{4}
C. a{3}
D. a[2][2]={{1,2},{3,4}}

65.C++ tilining grafik imkoniyatlarini ochib beruvchi kutubxonani belgilang.

A. time.h
B. string.h
C. conio.h
D. graphics.h

66.Dasturning tashqi ko‘rinishini shakllantirish uchun qaysi kutubxonadan foydalaniladi?

A. conio.h
B. graphics.h
C. time.h

D. string.h

67.C++ tilining asosiy funksiyasi qaysi?
A. return
B. void
C. include
D. main()

68.C++ tilida arctg(x) qanday yoziladi?
A. atan(x)
B. arctan(X)
C. tan(x)
D. atan x

69.Run menyusining vazifalari nimalardan iborat?
A. Loyihada formani qo‘shishga va o‘chirishga imkon beradi
B. Loyihani normal va sozlovchi rejimlarda bajarishga imkon beradi
C. Yangi kompyuterlarni yaratish va o‘rnatishga imkon beradi
D. Malumotlar bazasi bilan ishlash uchun foydalanishga imkon beradi

70. Obyekt bu – ….
A. Malumotlarni qayta ishlash usuli
B. Malumotlarni yozish
C. Foydalanuvchi interfeysi
D. Ma’lumotlar va ular bilan ishlash usullari majmuasi

71.Obyektga yo‘naltirilgan dasturlashning asoslari nimalar?
A. Obyektlar va sinflar
B. Obyektlar
C. Sinflar
D. Funksiyalar

72.Algoritmning xossalarini toping.
A. Chiziqli ,tarmoqlanuvchi ,takrorlanuvchi
B. Aniqlanganlik,tushunarlilik,diskritlik,ommaviylik natijaviylik
C. Aniqlanganlik , tushunarlilik,diskritlik
D. Tarmoqlanuvchi ,takrorlanuvchi

73.Algoritm bu - ……
A. Buyruqlar va amallar ketma-ketligi
B. Amallar ketma-ketligi
C. Chekli amallar ketma-ketligi
D. Buyruqlar ketma-ketligi

74.Algoritmning berilish usullari qanday?

A. Matn orqali
B. Formulalar orqali
C. Jadval ko‘rinishida
D. Matn ,formula,dastur, jadval,algoritmik til, blok - sxema

75.Algoritm so‘zi qaysi olim nomi bilan bog’liq ?

A. Abu Rayhon Beruniy
B. Al-Xorazmiy
C. Mirzo Ulug’bek
D. Ibn Sino

76.*for(int i=1;i<=10;i+=3) i--;* takrorlash operatori necha marta ishlaydi?

A. 5
B. 10
C. Cheksiz
D. Umuman ishlamaydi

77.C++ tilida necha o‘lchovgacha massivlar ishlatish mumkin?

A. 31
B. 7
C. 32
D. 15

78.Massiv elementlariga tasodifiy qiymatlar berish uchun qaysi direktiva ishlatiladi?

A. stdlib.h
B. iostream.h
C. math.h
D. string.h

79. y=$\frac{1}{x^4+ln(x)}$ ni C tilida yozing.

A. y=1/(pow(x,4+log(x));
B. y=1/(pow(x,4)+log(x));
C. y=1/(pow(x,4))+log(x);
D. y=1*(pow(x,4)+log(x))

80.do …. while so‘zining ma’nosi nima?

A. do <shart> while <toki>
B. while <toki>
C. Do <bajar> while <toki>
D. do<uchun while <toki>

81.y=a · $e^{lgx} - \sqrt{a \cdot x + b}$ C tiliga o‘giring.

A. y=a*exp(log10(x))-sqrt(a*x+b);
B. y=a ·exp(log10(x))-sqrt(a*x+b);
C. y=a*exp(log10(x))-sqrt(a*x+b;
D. y=a*exp(log (x))-sqrt(a *x+b);

82. C ++ tilida miqdorlar necha xil bo‘ladi?

A. 3
B. 4
C. 5
D. 2

83.  Blok qanday ko‘rsatmani bildiradi?

A. Oddiy harakat
B. Boshlanishni
C. Qiymat kiritishni
D. Shart tekshiradi

84.Belgili o‘zgarmas qaysi javobda to‘g’ri tavsiflangan?

A. “salom”
B. Tenglama
C. 9.78
D. 9e6

85.*int a=2,b=4,c; c=(a+b)*a%b;* natijani aniqlang.

A. 6
B. 8
C. 10
D. 12

86. ifstream qanday vazifani bajaradi?

A. Faylni o‘qish
B. Faylni yaratish
C. Faykni yaratish va o‘qish
D. Faylni yozish

87. fout.open - ……..

A. Faylni yopish
B. Faylni o‘qish
C. Obyektni fayl bilan bog’lash
D. Faylni yozish

88. fout.close - ……

A. Faylni yozish
B. Faylni o‘qish
C. Faylni yopish
D. Faylni ochish

89.Oshkormas holatda beriladigan massiv ……. massivdir
A. Dinamik
B. Statik
C. Ikki o‘lchovli
D. Ko‘p o‘lchovli

90.C++ tilida yozing. y=$sin^2x^2 + lnx^3 - 3\sqrt{x}$
A. y=pow(sin(pow(x,2)),2)+log(x*x*x)-3*sqrt(x);
B. y=pow(sin(pow(x,2)),2)+log(x*x*x)+3*sqrt(x);
C. y=pow(sin(pow(x,2) ,2))+log(x*x*x)-3*sqrt(x);
D. y=pow(sin(pow(x,2)),2)+log(x*x)-3*sqrt(x);

91.y=$tg^2 3x + \sqrt{x + 0.5|x|} + a\ lnx^3$ C++ tilida yozing?
A. y=tan(3*x)*tan(3*x)+sqrt(x+1/2*abs(x))+a*log(pow(x,3))
B. y=tan(3*x)*tan(3*x)+sqrt(x+1/2*abs(x))-a*log(pow(x,3))
C. y=tan(3*x)*tan(3*x)+sqrt(x+1/2*abs x)+a*log(pow(x,3))
D. y=tan(3*x) tan(3*x)+sqrt(x+1/2*abs(x))+a*log(pow(x,3))

92. switch so‘zining manosini toping?
A. Tasodifiy
B. Ifoda
C. Uchun
D. Belgi

93.Shartli operatorning umumlashmasidan iborat operator qaysi?
A. Taqqoslash operatori
B. Sikl operatori
C. TJY
D. Tanlash operatori

94. ……. - # define makro qiymati ?
A. Makroslar
B. Define
C. Mikroslar
D. Derektivalar

95.y=y*(a+3) qiymat berish buyrug’ini qisqartirilgan ko‘rinishi berilgan qatorni toping?

A. y=y*3
B. y*a+y*3
C. y=y(a+3)
D. y*=a+3

96.CodeBlocks da menyular satri nechta bo‘limdan iborat?
A. 11
B. 10
C. 12
D. 9

97. CodeBlocksda sarlavha satrida qanday asosiy tugma mavjud?
A. Dastur nomi va uskunalar paneli
B. Dastur nomi va oynani boshqarish
C. Uskunalar paneli
D. Oynani boshqarish

98.y=$arccos^3(6x+3)+\sqrt[5]{5^{x+a}}*10^{-7}$ C++ tilida yozing?
A. (pow(acos*(6*x+3),3))+pow(5,(x+a)/5)*1/pow(10,7)
B. (pow(arccos*(6*x+3),3))+pow(5,(x+a)/5)*1/pow(10,7)
C. (pow(acos*(6*x+3),3))-pow(5,(x+a)/5)*1/pow(10,7)
D. (pow(acos*(6*x+3),3))+pow(5,(x+a)/5)*1/pow(10,-7)

99.z=$\sqrt{ax+b}\cdot 36{,}6x+\log_8 x+10^{-8}\cdot sec(x)$ C++ tiliga o‘giring?
A. z=pow((a*b+b),1/2)*36.6*x+(log10(x)/log10(8))+1/pow(10,-8)*1/cos(x)
B. z=pow((a*b+b),1/2)*36,6*x+(log10(x)/log10(8))+1/pow(10,-8)*1/cos(x)
C. z=pow((a*b+b),1/2)*36,6*x+(log10(x)/log10(8))+1/pow(10.-8)*1/cos(x)
D. z=pow((ab+b),1/2)*36,6*x+(log10(x)/log10(8))+1/pow(10,-8)*1/cos(x)

100. *int a=43,b=3;cout<<a /b;* natijani aniqlang?
A. 14
B. 14.3333
C. 14,3333
D. 14,(3)

101.Kompyuter tushunadigan yagona til?
A. kompilyator
B. mashina kodi
C. dastur tili alifbosi
D. dasturlash tillari

102. \a Escape belgisini vazifasi?

A. keying satrga o‘tish
B. tovush signalini chiqaradi
C. tabulyatsiya chiqarish
D. hech qanday vazifa bajarmaydi

103.O‘nlik sanoq sistemasidan o‘n oltilik sanoq sistemasiga o‘tish uchun qaysi manipulyatordan foydalaniladi?

A. hex
B. dec
C. ost
D. exp

104.Int ma’lumot turi xotiradan qancha joy egallaydi?

A. 3
B. 5
C. 1
D. 4

105.Ma’lumot turi oldidan signed xizmatchi so‘zini ishlatishning ahamiyati.

A. tur qiymati musbat,ham manfiyligini bildiradi
B. faqat musbat
C. faqat manfiy
D. ishora ahamiyati yo‘q

106.Int turining qiymat qilish diapazoni ?

A. -2147483648…+2147483648
B. -32768…+32768
C. -64561845…+64561845
D. -128…+128

107.Identifikator tog’ri berilgan qatorni toping.

A. 7bb,olma
B. #kuch,nom
C. b_4,a77
D. 3ol,$bb

108. Qiymatni o‘zlashtirish belgisini ko‘rsating.

A. /
B. +
C. =

D. %

109. cin va cout operatori qaysi kutubxonada joylashgan?

A. <iostream>
B. <cmath>
C. <cet>
D. <fstream>

110. x sonni kub ildizini qaytaradigan matematik funksiyani ko‘rsating.
A. sqrt
B. hypot
C. pow
D. cbrt

111.goto da nishon bu-….
A. Davomida ; qo‘yilgan identifikator
B. O‘zgaruvchi
C. Davomida : qo‘yilgan identifikator
D. O‘zgarmas

112.Funksiya e’loni ……. tavsiflaydi?
A. Funksiya aniqlanishini
B. Funksiyani chaqirish
C. Funksiya prototipi
D. TJY

113. Agar programmada o‘zgaruvchilar birorta tashqi qurilma bilan bog’lash uchun ishlatish zarur bo‘ladigan bo‘lsa , qaysi modifikatordan foydalanamiz?
A. void
B. main
C. sum
D. volatile

114.Qaysi qatorda massiv to‘g’ri e’lon qilingan?
A. A[2][3]
B. A[3]=>{2,4,3}
C. A[2}
D. A{4}==[1,2,3,4]

115. y=$e^{\frac{3+4x}{ln10}}$+ln (cosx) C tiliga o‘giring?
A. y=exp((3+4*x)/log(10))+log(cos(x));

B. y=exp((3+4x)/log(10))+log(cos(x));
C. y=exp((3+4*x)/log(10))+log(e,cos(x));
D. y=exp((3+4*x)/log(10)+log(cos(x)));

116. …….. – funksiya sarlavhasi va figurali qavsga olingan qandaydir amaliy mazmunga ega tanadan iborat bo‘ladi?

A. Funksiya chaqirilishi
B. Funksiya e’loni
C. TJY
D. Funksiya aniqlanishi

117. Strukturalar massivining har bir elementining maydonlariga murojaat nima orqali amalga oshiriladi?

A. .
B. ,
C. “ ”
D. !

118.Matn fayllarning komponentari nima deb nomlanadi?

A. Struktura
B. Satrlar
C. Massiv
D. Belgi

119.Fayllar bilan ishlashning bosqichlarini to‘g’ri sanalgan qatorni toping.

1. Fayl o‘zgaruvchisi diskdagi fayl bilan bog’lanadi.
2. Fayl nomi o‘zgartirish.
3. Fayl ustida yozish yoki o‘qish amallari bajariladi.
4. Fayl yopiladi.

A. 1,3,2,4
B. 1,3,4,2;
C. 1,2,3,4
D. 1,4,2,3

120.y=$\frac{1+\frac{a}{b}}{2-ln(x+e^{ab})}$ ni C ++ da yozing.

A. y=(1+a/b)/(2-log(x+exp(a*b));
B. y=(1+a/b)*(2-log(x+exp(a*b)));
C. y=(1+a/b)/(2-log(x+exp(ab)));
D. y=(1+a/b)/(2-log(x+exp(a*b)));

121. double tipidagi o‘zgaruvchi qabul qilishi mumkin bo‘lgan qiymat qaysi qatorda berilgan?

A. 3.456812 va 5e-8
B. 0 va true
C. 1 va true
D. True va false

122. C++ da || belgining ma’nosi nima?

A. Mantiqiy yoki
B. Mantiqiy va
C. Mantiqiy ko‘paytirish
D. TJY

123. *int a=43,b=8;cout<<a /b;* natijani aniqlang?

A. 5.375
B. 5
C. 6
D. 5.4

124.C++ da :: nima?

A. Funksiyani chaqirish
B. Ko‘rinish sohasiga ruhsat berish
C. Strukturani tanlash
D. Sinf elementini tanlash

125.*int a=23,b=12; cout<<a/b+a%b-8 ;* natijani aniqlang.

A. 4.9166666666
B. 5
C. Dastur ishlamaydi
D. 4

126. y=$\frac{-x+\sqrt{b^2-7c}}{2abc}$ ni C ++ tiliga o‘giring?

A. y=(-x+sqrt(b*b-7*c))/(2*a*b*c)
B. y=(-x+sqrt(b *b-7c))/(2a*b*c)
C. y=(-x+sqrt(b*b-7*c))/2*a*b*c
D. y=-x+sqrt(b*b-7*c)/(2*a*b*c)

127. switch operatorining umumiy ko‘rinishi berilgan qatorni toping.

A. switch(ifoda)
{
case <ifoda1> : <1-ifodaning ketma ketligi; break;
case <ifoda2> : <2-ifodaning ketma ketligi; break;

……..
case <ifodaN> : <N-ifodaning ketma ketligi; break;
default: N+1 operatorning ketma ketligi;
}

B. switch(ifoda)
{
case <ifoda1> : <1-ifodaning ketma ketligi; break;
case <ifoda2> : <2-ifodaning ketma ketligi; break;
……..
case <ifodaN> : <N-ifodaning ketma ketligi; break;
}

C. switch(ifoda)
{
case <ifoda1> : <1-ifodaning ketma ketligi;
case <ifoda2> : <2-ifodaning ketma ketligi;
……..
case <ifodaN> : <N-ifodaning ketma ketligi;
default: N+1 operatorning ketma ketligi;
}

D. switch(ifoda)
{
case <ifoda1> : <1-ifodaning ketma ketligi; break;
case <ifoda2> : <2-ifodaning ketma ketligi; break;
……..
case <ifodaN> : <N-ifodaning ketma ketligi; break;
default: N+1 operatorning ketma ketligi; break;
}

128.Takrorlash jarayonini boshqaradigan o‘zgaruvchining boshlang’ich qiymati nima deyiladi?

A. Shart
B. Initsializator
C. Sikl
D. O‘zgarish qadami

129. Xotirani taqsimlash uchun qaysi operatordan foydalaniladi?

A. printf
B. new
C. coino
D. sizeof

130. pTwo ko‘rsatkichi - ……

A. int tipli ko‘rsatkich bo‘lgan o‘zgaruvchini anglatadi
B. int tipli ko‘rsatkich bo‘lgan o‘zgarmasni anglatadi
C. int tipidagi o‘zgarmas manzilni o‘zida saqlaydi
D. int tipidagi o‘zgaruvchi manzilni o‘zida saqlaydi

131. main() funksiyasini necha xil usulda yozish mumkin?

A. 3
B. 4
C. 5
D. 2

132.Tiplarni yangi nomlar bilan atash uchun qaysi operatordan foydalaniladi?

A. typedef
B. typeof
C. new
D. typeor

133.To‘g’ri tasdiqni toping.

A. Birlashmalarda virtual metod, konstruktor va destruktor ,qiymat berish amallaridan foydalanib bo‘lmaydi.
B. Birlashmalar klasslar shajarasiga kiradi
C. Birlashmalar bitli maydonlarni o‘z ichiga oladi
D. TJY

134. ……. – faqat ma’lum bir tipdagi ma’lumotlarni saqlaydi.

A. Tiplashmagan fayllar
B. Matn fayllar
C. Tiplashgan fayllar
D. Binar fayllar

135. $x = \frac{\sqrt{\text{y } \log_5 y \cdot a^2}}{2a}$ C ++ tilida yozing.

A. x=sqrt(y*log(y*a*a)/log(5))/2*a
B. x=sqrt(y*log(y*a*a)/log(5)/2*a
C. x=sqrt(y*log(y*a*a,5))/2*a
D. x=sqrt(y*log(5,y*a*a))/2*a

136.Global o‘zgaruvchilarning amal qilish sohasi …….?

A. Amal qilish sohasi bo‘lmaydi
B. Butun dastur davomida bo‘ladi
C. Local o‘zgaruvchilari bilan bir xil.
D. Uning ko‘rinish sohasi bilan ustma-ust tushadi

137. Operand qiymatini birga oshirish nima deyiladi?

A. inkrement
B. dekremant
C. signed
D. posfiks

138. Biror xotira adresini o‘zida saqlaydigan o‘zgaruvchi yoki konstanta nima deyiladi?

A. identifikator
B. litera
C. operator
D. ko‘rsatkich

139.  - shakl C++tilida qanday ma’noni beradi?

A. boshlash, tamom
B. jarayon
C. o‘zgaruvchilar e’loni
D. hisoblash jarayoni

140. For operatorining pratativi to‘g’ri ko‘rsatilgan qatorni toping.

A. for(shart){ko‘rsatmalar ketma ketligi}
B. for{ko‘rsatmalar ketma ketligi}(shart)
C. for(<boshlang’ich qiymat><ifoda><ko‘rsatma>)
D. for(<boshlang’ich qiymat><ifoda>)

141. endl funksiyasining vazifasi?

A. yurgichni satr boshiga qaytarish
B. so‘roq belgisini chiqaradi
C. yangi qatorga o‘tish
D. hech qanday vazifa bajarmaydi

142. include so‘zining ma’nosi?

A. o‘z ichiga olmoq
B. oqim
C. cqartmoq
D. o‘qimoq

143. Ma’lumotni ekranga chiqarish operatori?

A. cin
B. cout
C. getline
D. endl

144. Matematik funksiyalar to‘g’ri ko‘satilgan qatorni toping.

A. pow, hypot, cbrt
B. sqrt,cout,pow
C. log(x),cin,exp(x)
D. endl,sqrt,pow

145. UNIX operatsion sistemasi ilk bor ommaga qachon taqdim qilingan?

A. 1974-yil
B. 1978-yil
C. 1965-yil
D. 1969-yil

146. C++ tili kim tomonidan yaratilgan?

A. Denis Rechi
B. Byorn Straustrup
C. Ken Tompson
D. Bu haqda aniq ma’lumot yo‘q

147. C++dasturlash tili qachon ishlab chiqarilgan?

A. 1985-yil
B. 1972-yil
C. 1981-yil
D. 1980-yil

148. Escape belgilari qaysi belgidan boshlanadi?

A. /
B. \
C. ” ”
D. @

149. Apostrof to‘g’ri ko‘rsatilgan qatorni toping.

A. “ ”
B. !
C. $
D. ‘ ‘

150. x^3 ning C++ tilida yozilish dasturini toping.

A. cbrt(x)
B. pow(x,3)
C. hypot(x)
D. sqrt(3)

151. C++ tili alifbosida nechta harf qatnashgan?

A. 25 ta
B. 52 ta
C. 30 ta
D. 48 ta

152. Dekrement amalining vazifasi.
A. qiymatni birga kamaytiradi
B. qiymatni birga oshiradi
C. qiymatni o‘zgaruvchiga o‘zlashtiradi
D. qiymatni 2 barobar kamaytiradi

153.Parametrli takrorlash operatori?
A. While
B. Do-while
C. If
D. For

154.Massiv-nima ?
A. har xil turdagi ma’lumot majmui
B. aniq o‘lchamli,bir turdagi,tartib raqamiga ega ma’lumotlar majmui
C. barchasi to‘g’ri
D. elementlar majmui

155. Massiv elementlari tartib raqami qanday sonlardan iborat?
A. manfiymas butun sonlar
B. haqiqiy sonlar
C. musbat butun sonlar
D. butun sonlar

156. Massiv e’loni to‘g’ri ko‘rsatilgan qatorni toping.
A. int olam[3]={1,2.1,3}
B. float yurt[5]={‘a’,2,3,4,5}
C. int olma[4]={1,2,4,6}
D. int olam[2]={12323}

157.Tiskl operatorlari.
A. for ,else, if;
B. else ,if
C. for ,while,do-while;
D. endl,for

158. Bir nechta satrli izoh kiritish uchun qaysi izoh satridan foydalanish maqul?

A. /* */
B. //
C. /*
D. *// *//

159. Identifikatordagi belgilar uzunligi chegarasi?

A. 31 ta
B. 255 ta
C. 24 ta
D. Cheksiz

160. logarifmik funksiyalar berilgan qatorni toping.

A. Log10(x), Log(x), Log2(x),
B. Log10(x), Log3(x), Log(x),
C. Log3(x), Log10(x), Log2(x),
D. Ln(x),log10(x),

161. <string > kutubxonasi qaysi kutubxonaga kiradi?

A. <iostream>
B. <cmath>
C. <fstream>
D. <string>

162.Satr uzunligini aniqlash uchun qaysi funksiyadan foydalaniladi?

A. empty()
B. append()
C. math()
D. length()

163. Berilgan masalani kompyuter yechishi uchun,algoritm asosida kompyuter tushunadigan tilga o‘girib yozish nima deyiladi?

A. Dastur
B. operator
C. kompilyator
D. ko‘rsatkich

164. Teng emas belgisi dasturda qanday yoziladi?

A. ≠
B. !=
C. =!
D. ==

165.Identifikatorlar noto‘g’ri berilgan qatorni toping.

A. #den, 3olma;
B. Yur, phone;
C. Turna ,nom;
D. Kema,kitob

166.Identifikatorning turlari qaysi qatorda berilgan?

A. zaxira, konstanta, o‘zgaruvchi;
B. o‘zgaruvchi,o‘zgarmas,funksiya;
C. funksiya,o‘zgarmas,zaxira
D. zaxira, standart,foydalanuvchi;

167.Dasturchi tomonidan aniqlanadigan identifikator.

A. Foydalauvchi
B. zaxira
C. standart
D. o‘zgaruvchi

168.Rezerved identifikator deganda qaysi identifikator tushuniladi?

A. foydalanuvchi
B. Zaxira
C. standart
D. o‘zgaruvchi

169.Dasturda berilgan ko‘rsatma ta’siri tugaganini qaysi belgi orqali bildiriladi?

A. ,
B. :
C. ;
D. ”

170.Dasturga kiritilgan son, belgi va matn nima deyiladi?

A. konstanta
B. o‘zgaruvchi
C. Litera
D. foydalanuvchi

171.Massiv elementlari tartib raqami nechidan boshlanadi?

A. 1
B. 0
C. Element o‘zidan
D. -1

172. To‘g’ri burchakli uchburchakning a va b tomonlari berilgan gepotenuzani qaysi usul bilan yechish qulaylik beradi?

A. sqrt(pow(a,2)+ pow(b,2));
B. pow((a*a+b*b),1/2
C. hama holat bir xil
D. hypot(a,b);

173. Quyidagi ma’lumotlarning qay biri to‘g’ri?

A. massiv o‘lchami= eng katta tartib raqamga
B. massiv o‘lchami,elementlar bir turga mansub
C. massiv o‘lchami,eng katta tartib raqamga
D. massiv o‘lchami= berilgan elementlar soniga

174. Dastur nechta turga bo‘linadi?

A. 2 turga
B. 3 turga
C. 4 turga
D. 5 turga

175.Grafikda jarayon vazifasini o‘taydigan shaklni ko‘rsating.

A. 
B. 
C. 
D. 

176.Takrorlash operatorlari ishida ba’zan ma’lum bir qiymatlar uchun biror bir qism bajarilmasligi zarur bo‘lib qoladi.Bunday hollardan qanday operator ishlatiladi?

A. printf
B. Continue
C. break
D. for

177. $\sqrt{x+\frac{a}{b}-e^{-t}}$ ni C++ tilida yozing.

A. sqrt(x+a/b-exp(-t))
B. pow((x+a/b-exp(-t)),2)
C. sqrt(x+a/b-exp(-t)
D. sqrt(x+a*b-exp(-t))

178.sqrt funksiyasi qaysi kutubxonalarga tegishli?

A. string.h, iostream.h

B. math.h, cmath
C. stdio.h, conio.h
D. stdlib.h, iostream

179. Bir yoki har xil turdagi belgilarni jamlanmasi - …
A. binar fayllar
B. struktura
C. massiv
D. dinamik massiv

180. C++ tilida ma’lumotlarni qog’ozga chiqarish qaysi shakl ichida bo‘ladi?
A. 
B. 
C. 
D. 

181. Dasturlash muhitida dasturlarni yozish va qayta ishlash nechta bosqichda amalga oshiriladi?
A. 4 ta
B. 6 ta
C. 10 ta
D. 5 ta

182.C++ da haqiqiy o‘zgaruvchilar necha xil bo‘ladi?
A. 2 xil
B. 4 xil
C. 3 xil
D. 5 xil

183. Quyidagilardan qaysi biri o‘zlashtirish operatori?
A. z:-0
B. a:=b
C. a=c+1
D. x=‘a’+1

184. Codebloks muhitini ishga tushirish ketma-ketligini to‘g’ri berilgan javobni toping.
A. Пуск->все программы->CodeBloks-> CodeBloks.
B. Пуск-> CodeBloks-> CodeBloks.
C. Пуск->все программы->CodeBloks
D. Пуск-> CodeBloks

185. C++ tilining grafik imkoniyatlaridan foydalanish uchun qaysi derektivadan foydalaniladi?

A. #include<string.h>
B. #include<stdeb.h>
C. #include<conio.h>
D. #include<graphics.h>

186. $z=(1+u)\frac{x-\frac{y}{u}}{a-\frac{1}{1-x^2}}$ ifodani C++ tilida yozing.

A. z=(1+u)*((x-y/u)/(a-1/(1-x*x)))
B. z=(1+u)((x-y/u)/(a-1/(1-x*x)))
C. z=(1+u)*((x-y/u)\(a-1/(1-x*x)))
D. z=1+u*((x-y/u)/(a-1/(1-x*x)))

187. C++ tilida char tipi necha bitni egallaydi?

A. 16 bit
B. 32 bit
C. 4 bit
D. 8 bit

188. C++ tilida char tipining qiymat chegarasi ko‘rsating?

A. 0…255
B. -128…127
C. -32768…32767
D. -147483648…147483647

189. C++ komilyatorlari eng kamida nechta indeks bilan ishlaydi?

A. 10 ta
B. 12 ta
C. 15 ta
D. 30 ta

190. C++ tilida murojaatlar qanday e’lon qilinadi?

A. <tur> * <nom>
B. <tur> <nom>
C. <tur> & <nom>
D. <tur> ! <nom>

191. C++ tilida satrdagi kichik harflarni katta harflarga o‘tkazishda qaysi funksiyadan foydalanamiz?

A. strupr
B. stricmp

C. strcmp
D. strlwr

192. C++ tilida birlashmalar qaysi so‘z yordamida kiritiladi?
A. bioskiy
B. union
C. struct
D. sizeof

193. C++ tilida strrchr funksiyasining vazifasi nima?
A. Berilgan belgini berilgan satr oxiridan boshlab izlaydi.
B. 2 ta satrni mos o‘rindagi belgilarini solishtiradi.
C. Str1 satrdagi str2 satrga kiruvchi birorta belgini izlaydi.
D. S belgining satr string satrida izlaydi.

194. C++ tilida faylni faqat o‘qish uchun qaysi oqimdan foydalaniladi?
A. ″r″
B. ″a″
C. ″w″
D. ″wt″

195. C++ da matnli fayllar bilan ishlash uchun qaysi funksiyadan foydalaniladi?
A. foper
B. argc
C. fget, fputs
D. felose

196. C++ tilida har qanday ‘;’ belgi bilan tugaydigan ifodaga nima deyiladi?
A. O‘zgaruvchi.
B. Satr
C. Itmal.
D. Til ko‘rsatmasi.

197. Direktivalar qanday belgi orasida keltiriladi?
A. < >
B. ‘‘
C. “ ”
D. << >>

198. *int a=2,b=11; cout<<(a%b)%a;* natijani aniqlang.
A. 0
B. 1
C. 2
D. Dastur ishlamaydi.

199. Blok- …va…belgi oralig’iga olingan operatorlar ketma-ketligi yaxlit bir operator deb qabul qilinadi.
Nuqatalar o‘rnini to‘ldiring.

A. “ va”
B. ‘{‘ va ‘}’
C. < va>
D. << va >>

200. O‘zgaruvchi mavjud bo‘lgan programma bo‘lagining bajarilishiga ketgan vaqt intervaliga nima deyiladi?

A. O‘sish vaqti
B. Yashash vaqti
C. Ko‘rinish sohasi
D. Amal qilish sohasi

GLOSSARIY

O‘zbekcha	Ruscha	Englizcha
Active – ma’lumotlar to‘plamini ochsh (*True*) va yopish (*Falce*)	**Active -** открытие (*True*) и закрытие (*Falce*) набора данных	**Active** - opening (True) and closing (False) dataset
AND – mantiqiy ko‘paytirish	**AND -** логическое умножение	**AND** - is logical multiplication
Add - Item parametr orqali aniqlanadigan elementni oxiriga qo‘shish	**Add -** добавить определяемый параметром Item элемент в конец	**Add** - add a parameter defined by the Item element to the end
Amaliy dasturiy ta’minot - muayyan foydalanuvchi ehtiyojlarini qondirish uchun mo‘ljallangan.	**Прикладное программное обеспечение** предназначено для решения конкретных задач пользователя.	The application software is designed to address specific user needs.
abs(*z*), **abs**(*x*)- *z* kompleks sonini modulini hisoblash yoki haqiqiy x sonini absolyut (mutlaq) qiymati	**abs**(*z*),**abs**(*x*)- вычисление абсолютное значение комплексное число z или вычисление абсолютное значение *x.*	**abs**(*z*), **abs**(*x*)- the calculation of z complex module or the absolute value of the real x number number
Bool – mantiqiy turdagi o‘zgaruvchi tavsifi	**Bool** - описание переменной логического типа	**Bool** - boolean variable declaration
Belgi (Piktogramma) - biror ob’ektni (fayl, dastur va hokazo) aynanlash uchun ekranda joylashgan kichik grafik tasvir.	**Значок (Пиктограмма)** – это небольшое графическое изображение, служащее для представления некоторого объекта (файл, программа, окно, устройство и т.д).	**Icon** - a small graphic image that serves to represent an object (a file, a program, a window unit, etc.).
C interfeysi. C tili interfeysi.	**C interface.** Интерфейс языка С.	**C interface.** C language interface
C kutubxonasi. C tili kutubxonasi.	**библиотека С.** Библиотека языка С.	**C library.** C language library.
C language. C tili.Operatsion tizimning katta qismi yozilgan umumiy maqsadli dasturlash tili.	**C language.** Язык С. Язык программирования общего назначения, на котором написана большая часть операционной системы.	**C language.** A general-purpose programming language in which most of the operating system is written.
C++ tili. C++ tili. Bjarne Stroustrup tomonidan C tili asosida ishlab chiqilgan dasturlash tili	**C++ language.** Язык С++. Язык программирования, разработанный Бьерном Страуструпом на основе языка С	**C++ language.** Programming language developed by Bjarne Stroustrup based on the C language
Canvas – chizish uchun vositalar o‘rnatiladi	**Canvas -** набор средств для рисования	**Canvas** - a set of tools for drawing
Caption - imzo	**Caption -** подпись	**Caption** - caption
Cin- Standart kiritish oqimi	**Cin-** Стандартный входной поток	**Cin-** Standard input stream
Clear - tozalash	**Clear -** очистить	**Clear** - clear

Close - yopish
Color – komponentlarni rang bilan to‘ldirish

const – o‘zgarmaslarni bo‘limda e’lon qiladi

CopyToClipboard - almashish buferiga nusha olish
CutToClipboard - almashish buferiga qirqib olish
Double- suzuvchi nuqtali ikki aniqlikdagi haqiqiy sonni ifodalaydi

Dec- barcha butun qiymatlarni 10 lik sanoq sistemasida chiqaradi

Dasturiy ta’minot - kompyuter tizimining ishlashi uchun dasturiy ta’minot va hujjatlarni majmui va undan foydalanish.

Dasturiy ta’minot to‘plami – masalalarni echishning ma’lum bir sinfini hal qilish uchun mo‘ljallangan dasturlar to‘plamidir.
Display – ko‘rsatish
div – bo‘linmaning butun qismi
Execute - bajarish

Fayl - bu tashqi xotira qurilmasida biror nom bilan saqlangan to‘plam.

Faylning asosi - fayl nomi va uning kengaytmasining birikmasi.

FileName – fayl nomi
Font – shrift xossasi

for<Параметр> := <Выражение1> **downto** <Выражение2> **do** <Оператор>; - bir qadamga kamayib borishni tashkil qiluvchi parametrli takrorlanish jarayoni

for<Параметр> := <Выражение1> **to** <Выражение2> **do** <Оператор>; - bir qadamga ortib borishni

Close - закрыть

Color - цвет заливки компонента

const - объявляет раздел описания констант

CopyToClipboard - копировать в буфер обмена
CutToClipboard - вырезать в буфер обмена
Double-представляет вещественное число двойной точности с плавающей точкой

Dec- выводит все целые значения в 10-значной системе счисления

Программное обеспечение - это совокупность программ и документации, обеспечивающих функционирование вычислительной системы, и их применение по назначению.

Пакет прикладных программ – это комплекс программ, предназначенный для решения задач определённого класса.

Display - показ
div - целочисленное деление
Execute - выполнить
Файл – это именованная совокупность данных, размещенных на внешнем запоминающем устройстве.

Составное имя файла – это совокупность имени файла и его расширения.

FileName - имя файла
Font - параметры шрифта

for<Параметр> := <Выражение1> **downto** <Выражение2> **do** <Оператор>; - организует *цикл с параметром* с убывающим шагом

for<Параметр> := <Выражение1> **to** <Выражение2> **do** <Оператор>; - организует *цикл с параметром* с возрастающим

Close - Closes
Color - the color component fill
const - declares constants describing section
Copy To Clipboard - copy to clipboard
CutToClipboard - cut to the clipboard
Double - represents a floating-point double-precision real number
Dec- outputs all integer values in the 10-digit number system
Software - a set of software and documentation for the operation of a computer system, and their intended use.
Software package - a set of programs designed to address a specific class of problems.
Display - display
div - integer division
Execute - perform
File - this is a named collection of data stored in the external storage device.
A composite file name - a combination of the file name and its extension.
FileName - file name
Font - Font settings
for <parameter>: = <Expression 1> **downto** <Expression 2> **do** <statement>; - Organizes the cycle parameter with decreasing step
for <parameter>: = <Expression 1> **to** <expression2> **do** <statement>; -

tashkil qiluvchi parametrli takrorlanish jarayoni	шагом	Organizes the cycle parameter with increasing step
Height - balandlik	**Height -** высота	**Height** - height
Icon - belgi, shaklga qo‘llanilgan	**Icon -** иконка, внедрённая в форму	**Icon** - icon that has been put into shape
Identifikator. Funksiya, o‘zgaruvchi yoki boshqa foydalanuvchi tomonidan belgilangan ob’yektga berilgan nom	**Идентификатор.** Имя, присвоенное функции, переменной или любому другому определенному пользователем объекту	**Identifier.** Name given to a function, variable, or any other user-defined object
if <condition> **then** <action>; **if** <condition> **then** <action> **else** <other action>; - Shartli o‘tish operatori.	**if** <условие> **then** <действие>; **if** <условие> **then** <действие> **else** <другое действие>; - Условный переход: выполняется некоторое условие, следует выполнить некоторое действие, а следует выполнить другое действие	**if** <condition> **then** <action>; **if** <condition> **then** <action> **else** <other action>; - Conditional jump: some condition, you should perform some action, and should perform another action
Interfeys - bu foydalanuvchi kompyuter bilan yoki dastur bilan foydalaniladigan vosita.	**Интерфейс** – это средства взаимодействия пользователя с компьютером или с программой.	**Interface** - this means the user interacts with the computer or with the program.
ItemIndex – ajratilgan ro‘yxatning raqami **Items –** ro‘yxat punktlari	**ItemIndex -** номер выделенного пункта списка **Items -** пункты списка	**ItemIndex** - number selected list item **Items** - list of items
Kompilyatsiya qilish. Yuqori darajadagi dasturlash tilida yozilgan dastur matnini oraliq til, assembler tili yoki mashina tilidagi matnga aylantirish.	**Компилировать.** Преобразовывать текст программы, написанной на языке программирования высокого уровня, в текст на промежуточном языке, ассемблере или машинном языке.	**Compile.** Convert the text of a program written in a high-level programming language to text in an intermediate language
Kompyuterlar turlari - analog, raqamli va gibrid.	**Типы ЭВМ** – аналоговые, дискретные и гибридные.	**Types of computers** - analog, digital, and hybrid.
Mantiqiy operator (!): ! noto‘g’ri = rost ! rost = noto‘g’ri	**Логический оператор (!):** ! ложь = истина ! истина = ложь	**Boolean operator (!):** ! false = true ! true = false
Massiv-umumiy nom bilan kirish mumkin bo‘lgan bir xil turdagi o‘zgaruvchilar to‘plami	**Массив-**коллекция переменных одного типа, обращение к которым осуществляется по общему имени	**Array-**collection of variables of the same type, accessed by a common name
Mikroprotsessor - barcha qurilmalar ishlashini nazorat qilish uchun va axborot arifmetik mantiq operatsiyalarini amalga oshirish uchun markaziy	**Микропроцессор** – это центральное устройство ПК, предназначенное для управления работой всех устройств и для выполнения арифметико-	**Microprocessor** - a central computer unit for controlling operation of all devices and for

kompyuter birligi.	логических операций над информацией.	performing arithmetic logic operations on information.
Ob’ektga yo‘naltirilgan dasturlash (OYD) - dasturlash metodologiyasi uchta xarakterli tamoyilga asoslangan: inkapsulyatsiya, polimorfizm va meros	**Объектно-ориентированное программирование (ООП) -** методология рограммирования, в основе которой лежат три характерных принципа: инкапсуляция, полиморфизм и наследование	**Object Oriented Programming (OOP) -** programming methodology based on three characteristic principles: encapsulation, polymorphism and inheritance
O‘zgaruvchan ishga tushirish. Bir vaqtning o‘zida o‘zgaruvchini e’lon qilish va unga qiymat berish	**Инициализация переменной.** Одновременное объявление переменной и присваивание ей значения	**Variable initialization.** Simultaneously declaring a variable and assigning a value to it
O‘zgaruvchi - bu ma’lum bir qiymatni belgilash mumkin bo‘lgan xotiradagi nomlangan joy	**Переменная**-именованная ячейка памяти, которой может быть присвоено определенное значение	**Variable**- is a named location in memory that can be assigned a specific value
Picture - rasm, komponentga qo‘llanilgan	**Picture -** рисунок, внедрённый в компонент	**Picture -** drawing, is inserted in the component
Position - komponentning jotiy qiymati	**Position -** текущее значение компонента	**Position -** the current value of the component
Print – printerga chiqarish	**Print -** печать на принтере	**Print -** print to the printer
Rekursiya - bu o‘zini chaqiradigan funktsiya jarayoni	**Рекурсия**- процесс вызова функцией самой себя	**Recursion**- is the process of a function calling itself
Repeat <Действия> **until** <Условия> - Sarti keyin tekshiriladigan operator	**Repeat** <Действия> **until**<Условия> - Оператор цикла с постусловием. **Действия** в теле цикле последовательно повторяясь, будут происходить до тех пор, пока не выполнится **Условие**	**repeat** <Options> **until** <Conditions> - operator cycle with postcondition. Actions in the body of the cycle sequence is repeated, will take place as long as the following conditions are satisfied
SaveToFile – faylga saqlash	**SaveToFile -** сохранение в файл	**SaveToFile -** saving a file
Style – component konturining rangi	**Style** - цвет контура компонента	**Style** - the color of the component circuit
Tizimiy ta’minot - kompyuter va kompyuter tarmoqlarini ishlashi uchun dasturlar va dasturiy ta’minot tizimlari majmui.	**Системное программное обеспечение** – это совокупность программ и программных комплексов для обеспечения	**System software** - a set of programs and software systems for the operation of the

	работы компьютера и вычислительных сетей.	computer and computer networks.
Var - o‘zgaruvchini e’lon qilish bo‘limi	**Var -** объявляет раздел описания переменных	Var - declares the variable declaration section
While <Условие> **do**<Действия> - sharti avval tekshiriladigan operator	**While** <Условие> **do**<Действия> - Оператор цикла с предусловием. **Действия** в теле цикла будут выполняться, **пока** (*while*) справедливо **условие**.	**while** <condition> **do** <action> - cycle operator with the precondition. Actions in the loop will run until (while) we have the condition.
!= - Teng emas operatori; mantiqiy inkor amali qiymati bilan birhil.	**!= -** оператор не равно; совпадает со значением логического отрицания.	**!=** -The inequality operator; compares values for inequality returning a bool.
#define- Makro deriktivalarni belgilash	**#define-** Определение директив макроса	**#define-** a directive that defines a macro.
#include - bir source fayl ichida boshqa bir faylag murojatni amalga oshirish mexanizmi	**#include -** механизм ссылки на другой файл в одном исходном файле.	**#include** - a mechanism for referencing another file within one source file
.c file. Dastur jodini o‘zida jamlovchi fayl	**.c file.** Файл, содержащий программу	**.c file.** File containing the program
.h file. Sarlavha fayli	**.h file.** Заголовочный файл	**.h file.** Header file
{…} -operatorlarning qavsi	**{…}**- операторные скобки	**{…}**- Curly bracket

E.

FOYDALANILGAN ADABIYOTLAR YO‘YXATI

1. Sh.F. Madraximov, S.M. Gaynazarov. C++ tilida programmalash asoslari. Toshkent, O‘zMU, 2009 y.

2. Никита Культин. Microsoft Visual C++ в задачах и примерах. БХВ-Петербург - Петербург. 2010.

3. Б. Страуструп. Язык программирования C++. Специальное издание.- М.:ООО «Бином-Пресс», 2006.-1104 с.

4. Павловская Т.А. C++. Программирование на языке высокого уровня – СПб. Питер. 2005- 461 с.

5. Павловская Т.С. Щупак Ю.С. C/C++. Структурное программирование. Практикум.-СПб.: Питер,2002-240 с.

6. Павловская Т.С. Щупак Ю.С. C++. Объектно- ориентированное

программ-мирование. Практикум.-СПб.: Питер,2005-265с

7. Герберт Ш. C++ Базовый курс. Москва Издательский дом Вильямс. 2010-621 с.

8. Стенли Липпман. Язык программирование C++. Базовой курс. Вильямс - М.: 2014.

9. Подбельский В.В. Язык C++.- М.; Финансы и статистика- 2003 562с.

10. Eshtemirov S. Nomozov F. C++ dasturlash tili. Uslubiy qo‘llanma. Samarqand 2016. -146 b.

11. Nazarov F. C++ tilida dasturlash asoslari. Uslubiy qo‘llanma. Samarqand 2017. -160 b.

12. J. Axmadaliyev, R. Holdarboyev. “C++ dasturlash tili”ni o‘rganish bo‘yicha uslubiy qo‘llanma. Andijon-2015 yil.

MUNDARIJA

ОГЛАВЛЕНИЕ

CONTENTS

Z.T. Negmatulloyev

ALGORITMIK TILLAR VA DASTURLASH

O‘QUV QO‘LLANMA

Guliston-2023