O'zbekistan Respublikasi aloqa axboratlashtirish va telekommunikatsiya texnologiyalari davlat qo'mitasi
Toshkent axborat texnologiyalar universiteti
Nukus filiali Kompyuter injiniringi fakulteti Kompyuter injiniringi yo'nalishi 2 A kurs talabasi
Baltabaev Jaxangirning
C++ dasturlash tili fanidan
tayyorlagan

Kurs ishi

Mavzu: *C++ tilida tenglamalar yechishga dasturlar tuzish .*

Topshirgan: *Baltabaev J*
Qabul qilgan: *Yadgarov.Sh*

No'kis 2015-yil

Mavzu: C++ tilida tenglamalar yechishga dasturlar tuzish

REJA:

KIRISH:

1.C++ dasturlash tili haqida

ASOSIY BO'LIM:

1.C++ dasturlash tilida matematik amallar berilishi

2.C++ da arifmetik ifodalarining tuzilishi

3. Tenglamaning yechilishiga oid dasturlar

4.C++ dasturlash tilida massivlar ustida amallar

5.Tenglamalar sistemasini Gauss usulida yeshish

YAKUNLOVCHI BO'LIM:

1.Amaliy mashgulotlar

2.Xulosa

KIRISH

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

C tilini Dennis Ritchie B dan keltirib chiqardi va uni 1972 yili ilk bor Bell Laboratoriyasida, DEC PDP-11 kompyuterida qo'lladi. C o'zidan oldingi B va BCPL tillarining juda ko'p muhim tomonlarini o'z ichiga olish bilan bir qatorda o'zgaruvchilarni tiplashtirdi va bir qator boshqa yangiliklarni kiritdi. Boshlanishda C asosan UNIX sistemalarida keng tarqaldi. Hozirda operatsion sistemalarning asosiy qismi C/C++ da yozilmoqda. C mashina arhitekturasiga bog'langan tildir. Lekin yahshi rejalashtirish orqali dasturlarni turli kompyuter platformalarida ishlaydigan qilsa bo'ladi.

1983 yilda, C tili keng tarqalganligi sababli, uni standartlash harakati boshlandi. Buning uchun Amerika Milliy Standartlar Komiteti (ANSI) qoshida X3J11 tehnik komitet tuzildi. Va 1989 yilda ushbu standart qabul qilindi. Standartni dunyo bo'yicha keng tarqatish maqsadida 1990 yilda ANSI va Dunyo Standartlar Tashkiloti (ISO) hamkorlikda C ning ANSI/ISO 9899:1990 standartini qabul qilishdi. Shu sababli C da yozilgan dasturlar kam miqdordagi o'zgarishlar yoki umuman o'zgarishlarsiz juda ko'p kompyuter platformalarida ishlaydi.

C++ 1980 yillar boshida Bjarne Stroustrup tomonidan C ga asoslangan tarzda tuzildi. C++ juda ko'p qo'shimchalarni o'z ichiga olgan, lekin eng asosiysi u ob'ektlar bilan dasturlashga imkon beradi.

Dasturlarni tez va sifatli yozish hozirgi kunda katta ahamiyat kasb etmoda. Buni ta'minlash uchun ob'ektli dasturlash g'oyasi ilgari surildi. Huddi 70-chi yillar boshida strukturali dasturlash kabi, programmalarni hayotdagi jismlarni modellashtiruvchi ob'ektlat orqali tuzish dasturlash sohasida inqilob qildi.

C++ dan tashqari boshqa ko'p ob'ektli dasturlshga yo'naltirilgan tillar paydo bo'ldi. Shulardan eng ko'zga tashlanadigani Xerox ning Palo Altoda joylashgan ilmiy-qidiruv markazida (PARC) tuzilgan Smalltalk dasturlash tilidir. Smalltalk da hamma narsa ob'ektlarga asoslangan. C++ esa gibrid tildir. C++ funksiya va ob'ektlarning juda boy kutubhonasiga ega. Yani C++ da dasturlashni o'rganish ikki qismga bo'linadi. Birinchisi bu C++ ni o'zini o'rganish, ikkinchisi esa C++ ning standart kutubhonasidagi tayyor ob'ekt/funksiyalarni qo'llashni o'rganishdir.

C++ DA ARIFMETIK AMALLAR

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

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

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

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

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

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

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

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

MANTIQIY SOLISHTIRISH OPERATORLARI

C++ bir necha solishtirish operatorlariga ega.

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

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

Yuqoridagi solishtirish operatorlarini ishlatadigan bir dasturni ko'raylik.

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

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

Bu yerda bizga yangi bu C++ ning if (agar) struktura-sidir. if ifodasi ma'lum bir shartning to'g'ri (true) yoki noto'g'ri (false)bo'lishiga qarab, dasturning u yoki bu blokini bajarishga imkon beradi. Agar shart to'g'ri bo'lsa, if dan so'ng keluvchi amal bajariladi. Agar shart bajarilmasa, u holda if tanasidagi ifoda bajarilmay, if dan so'ng keluvchi ifodalar ijrosi davom ettiriladi. Bu strukturaning ko'rinishi quyidagichadir:

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

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

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

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

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

QIYMAT BERISH OPERATORLARI

Bu qismda keyingi bo'limlarda kerak bo'ladigan tushuncha-larni berib o'tamiz.C++ da hisoblashni va undan keyin javobni o'zgaruvchiga beruvchi bir necha operator mavjuddir. Misol uchun:

k = k * 4; ni

k *= 4;

deb yozsak bo'aladi.

Bunda *= operatorining chap argumenti o'ng argumentga qo'shiladi va javob chap argumentda saqlanadi. Biz har bir operatorni ushbu qisqartirilgan ko'rinishda yoza olamiz (+=, -=, /=, *= %=). Ikkala qism birga yoziladi. Qisqartirilgan operatorlar tezroq yoziladi, tezroq kompilyatsiya qilinadi va ba'zi bir hollarda tezroq ishlaydigan mashina kodi tuziladi.

1 ga OSHIRISH VA KAMAYTIRISH OPERATORLARI (INCREMENT and DECREMENT) C++ da bir argument oluvchi inkrenet (++) va dekrement (--) operatorlari mavjuddir. Bular ikki ko'rinishda ishlatilinadi, biri o'zgaruvchidan oldin

(++f - preinkrement, --d - predekrement), boshqasi o'zgaruvchidan keyin (s++ - postinkrement, s-- - postdekrement) ishlatilgan holi. Bularning bir-biridan farqini aytin o'taylik. Postinkrementda o'zgaruvchining qiymati ushbu o'zgaruvchi qatnashgan ifodada shlatilinadi va undan keyin qiymati birga oshiriladi. Preinkrementda esa o'zgaruvchining qiymati birga oshiriladi, va bu yangi qiymat ifodada qo'llaniladi. Predekrement va postdekrement ham aynan shunday ishlaydi lekin qiymat birga kamaytiriladi. Bu operatorlar faqatgina o'zgaruvchining qiymatini birga oshirish/kamaytirish uchun ham ishlatilinishi mumkin, yani boshqa ifoda ichida qo'llanilmasdan. Bu holda pre va post formalarining farqi yo'q.

Masalan:

++r;

r++;

Yuqoridagilarning funksional jihattan hech qanday farqi yo'q, chunki bu ikki operator faqat r ning qiymatini oshirish uchun qo'llanilmoqda. Bu operatorlarni oddiy holda yozsak:

r = r + 1;

d = d - 1;

Lekin bizning inkrement/dekrement operatorlarimiz oddiygina qilib o'zgaruvchiga bir qo'shish/ayirishdan ko'ra tezroq ishlaydi. Yuqoridagi operatorlarni qo'llagan holda bir dastur yozaylik.

```
//Postinkremet, preinkrement va qisqartirilgan teglashtirish operatrlari
# include <iostream.h>
```

```
int main()
{
int k = 5, l = 3, m = 8;
cout << k++ << endl;    //ekranga 5 yozildi, k = 6 bo'ldi.
l += 4;                 // l = 7 bo'ldi.
cout << --m << endl;    // m = 7 bo'ldi va ekranga 7 chiqdi.
m = k + (++l);          // m = 6 + 8 = 14;
return (0);
}
```

Dasturdagi o'zgaruvchilar e'lon qilindi va boshqangich qiymatlarni olishdi. **cout** << k++ << endl; ifodasida ekranga oldin k ning boshlangich qiymati chiqarildi, keyin esa uning qiymati 1 da oshirildi. l += 4; da l ning qiymatiga 4 soni qo'shildi va yangi qiymat l da saqlandi. cout << --m << endl; ifodasida m ning qiymati oldin predekrement qilindi,va undan so'ng ekranga chiqarildi. m = k + (++l); da oldin l ning qiymati birga ishirildi va l ning yangi qiymati k ga qo'shildi. m esa bu yangi qiymatni oldi. Oshirish va kamaytirish operatorlari va ularning argumentlari orasida bo'shliq qoldirilmasligi kerak. Bu operatorlar sodda ko'rinishdagi o'zgaruvchi-larga nisbatan qo'llanilishi mumkin halos. Masalan:

```
++(f * 5);
```

ko'rinish noto'g'ridir.

MANTIQIY OPERATORLAR

Bosqaruv strukturalarida shart qismi bor dedik. Shu paytgacha ishlatgan shartlarimiz ancha sodda edi. Agar bir necha shartni tekshirmoqchi bo'lganimizda ayri-ayri shart qismlarini yozardik. Lekin C++ da bir necha sodda shartni birlashtirib, bitta murakkab shart ifodasini tuzishga yordam beradigan mantiqiy operatorlar mavjuddir. Bilar mantiqiy VA - && (AND), mantiqiy YOKI - || (OR) va mantiqiy INKOR - ! (NOT). Bular bilan misol keltiraylik. Faraz qilaylik, bir amalni bajarishdan oldin, ikkala shartimiz (ikkitadan ko'p ham bo'lishi mumkin) true (haqiqat) bo'lsin.

```
if (i < 10 && l >= 20){...}
```

Bu yerda {} qavslardagi ifodalar bloki faqat i 10 dan kichkina va l 20 dan katta yoki teng bo'lgandagina ijro ko'radi.

AND ning (&&) jadvali:

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

Bu yerda true ni yeriga 1, false ni qiymati o'rniga 0 ni qo'llashimiz mumkin.

Boshqa misol:

```
while (g<10 || f<4){
...
}
```

Bizda ikki o'zgaruvchi bor (g va f). Birnchisi 10 dan kichkina yoki ikkinchisi 4 dan kichkina bo'lganda while ning tanasi takrorlanaveradi. Yani shart bajarilishi uchun eng kamida bitta true bo'lishi kerak, AND da (&&) esa hamma oddiy shartklar true bo'lishi kerak.

OR ning (||) jadvali:

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

&& va || operatorlari ikkita argument olishadi. Bulardan farqli o'laroq, ! (mantiqiy inkor) operatori bitta argumet oladi, va bu argumentidan oldin qo'yiladi. Inkor

operatori ifodaning mantiqiy qiymatini teskarisiga o'zgartiradi. Yani false ni true deb beradi, true ni esa false deydi.

Misol uchun:

```
if ( !(counter == finish) )
  cout << student_bahosi << endl;
```

Agar counter o'zgaruvchimiz finish ga teng bo'lsa, true bo'ladi,bu true qiymat esa ! yordamida false ga aylanadi. false qiymatni olgan if esa ifodasini bajarmaydi. Demak ifoda bajarilishi uchun bizga counter finish ga teng bo'lmagan holati kerak. Bu yerda ! ga tegishli ifoda () qavslar ichida bo'lishi kerak. Chunki mantiqiy operator-lar tenglilik operatorlaridan kuchliroqdir. Ko'p hollarda ! operatori o'rniga mos keladigan mantiqiy tenglilik yoki solishtirish operatorlarini ishlatsa bo'ladi, masalan

yuqoridagi misol quyidagi ko'rinishda bo'ladi:

```
if (counter != finish)
  cout << student_bahosi << endl;
```

NOT ning jadvali:

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

Ifodalar

Dastur biror bir aniqlangan ketma - ketlikda bajariluvchi komandalar to'plamidan iborat..

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

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

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

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

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

Bo'sh joy (probel) belgisi.

Bo'sh joy belgilariga nafaqat probel, balki yangi satrga o'tish va tabulyasiya belgilari ham kiradi. Yuqorida keltirilgan ifodani quyidagicha ham yozish mumkin:

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

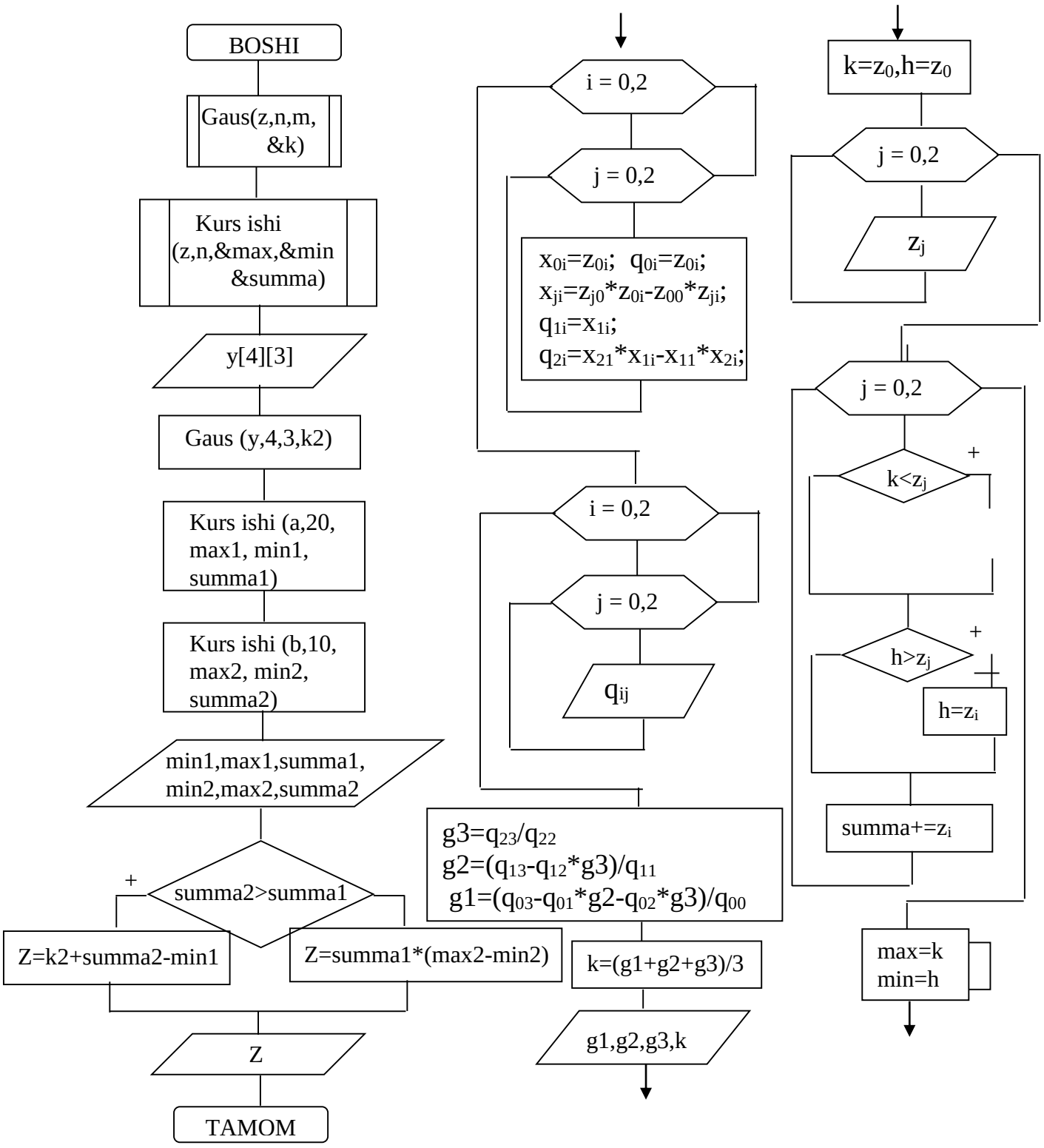
Bu variantda keltirilgan ifoda ko'rimsiz va tushunarsiz bo'lsa ham to'g'ridir. Bo'sh joy belgilari dasturning o'qilishliligini ta'minlaydi

TENGLAMALAR SISTEMASINI GAUS USULIDA YESHISH

Masalaga ko’ra tenglamalar sistemasini Gauss usulida yechish uchun noma’lumlar oldidagi koeffitsientlardan hosil bo’lgan massiv diagonali quyi qismini 0 larga aylanitirishimiz kerak. Buni men qism dastur sifatida yozdim. Bunda esa yuqorida qayd qilingan operatorlar bilan birga FOR takrorlanish operatorini ham qo’lladim. Bu operator ma’lum o’zgaruvchi ma’lum qiymatga yetguncha tarkorlanishlarni bajaradigan operator hisoblanadi. Bunga misolni siz asosiy dasturda ko’rishingiz mumkin.

Dastur tanasi.

Avvalo har qanday dasturni tuzishdan oldin qilinishi kerak bo’lgan ishlar algoritmi tuzib olinadi.



Yuqoridagi blok sxema dasturni ishga tushirish uchun zarur buyruqlar tartibidir. Lekin quyidagi programmada esa buyruqlar blok sexmadagidan ko’p. chunki ko’p buyruqlar porgramma naijasini ko’rishdagi dizayn, natijani chiqarishda qulaylik yaratish uchun yozilgan.

```
#include <iostream.h>

#include <conio.h>

#include <stdlib.h>

#include <time.h>

typedef float miss[20];

typedef float abc[10][10];      int i,j;

void gaus(abc z,int n,int m,float &k)             //1 rpocedura

{ abc x,q;    cout<<"\n\nberilgan tenglamalar sistemasi:"<<endl;

 for (i=0;i<n;i++) {x[0][i]=z[0][i];  q[0][i]=z[0][i];

  for (j=1;j<m;j++) x[j][i]=z[j][0]*z[0][i]-z[0][0]*z[j][i];

  q[1][i]=x[1][i]; q[2][i]=x[2][1]*x[1][i]-x[1][1]*x[2][i];}

 for (j=0;j<m;j++)

  { if(j==0) cout<<" ╔"; // sistema belgisini bildirish uchun mahsus simvollar

   else if (j==1) cout<<"╣ "; else cout<<" ╚";

  for (i=0;i<n;i++) {if((z[j][i]>=0) && (i!=0)&& (i!=3))
cout<<"+";cout<<z[j][i];if(i!=3) cout<<"*C"<<i+1;if(i==2)
cout<<"=";}cout<<endl;}

 cout<<"1-hisoblashdan keyin:"<<endl;

 for (j=0;j<m;j++)

  {if(j==0) cout<<" ╔"; // sistema belgisini bildirish uchun mahsus simvollar

   else if (j==1) cout<<"╣ "; else cout<<" ╚";for (i=0;i<n;i++) {if((x[j][i]>=0)
&& (i!=0)&& (i!=3)) cout<<"+";cout<<x[j][i];if (i!=3)
cout<<"*C"<<i+1;if(i==2) cout<<"=";} cout<<endl;}
```

```
    cout<<"2-hisoblashdan keyin:"<<endl;

    for (j=0;j<m;j++)

     { if(j==0) cout<<" ╔"; // sistema belgisini bildirish uchun mahsus simvollar

      else if (j==1) cout<<"╣ "; else cout<<" ╚";for (i=0;i<n;i++) {if((q[j][i]>=0)
&& (i!=0)&& (i!=3)) cout<<"+";cout<<q[j][i];if(i!=3)
cout<<"*C"<<i+1;if(i==2) cout<<"=";} cout<<endl;}

    float g3=0,g2=0,g1=0;  g3=q[2][3]/q[2][2]; g2=(q[1][3]-q[1][2]*g3)/q[1][1];

    g1=(q[0][3]-q[0][1]*g2-q[0][2]*g3)/q[0][0];

    cout<<"\nnatija:\n";

    cout<<"C1="<<g1<<"\nC2="<<g2<<"\nC3="<<g3<<endl;  // tenglamalar
sistemasi k=(g1+g2+g3)/3;cout<<"ularning o'rta arigmetigi="<<k;}
//yechimlari

   void kursishi(miss z,int m,float &max,float &min,float &summa)      //2
procedura

   {srand(time(0));int k=z[0],h=z[0];      summa=0;

    for (i=0;i<m;i++) z[i]=rand()/111-100;

    for (i=0;i<m;i++)

     {if (k<z[i]) k=z[i];

      if (h>z[i]) h=z[i];

      summa+=z[i];} max=k; min=h;}

   main()

   {textcolor (4);clrscr();textbackground(3);clrscr();

    miss a,b; float max1,max2,min1,min2,summa1=0,summa2=0,k2;

    cout<<"\n\t301-13  2a KI  studenti  Baltabaev Jahangirning   KURS ISHI\n";

    abc y={{7.09,1.17,-2.23,4.75},{0.43,1.4,-0.62,-1.05},{3.21,-1.25,2.13,-
5.06}};

    gaus (y,4,3,k2);       getch(); clrscr();
```

```
cout<<"\n\ t301-13  2a KI  studenti  Baltabaev Jahangirning  KURS ISHI\n";
kursishi(a,20,max1,min1,summa1);
 cout<<"\n\ta matritsaning elementlari yig'indisi="<<summa1<<endl;
 cout<<"\tminimum elementi="<<min1<<endl;
 cout<<"\tmaksimum elementi="<<max1<<endl<<endl;
kursishi(b,10,max2,min2,summa2);
 cout<<"\n\n\n\tb matritsaning elementlari yig'indisi="<<summa2<<endl;
 cout<<"\tminimum elementi="<<min2<<endl;
 cout<<"\tmaksimum elementi="<<max2<<endl;
long Z=0;
if (summa1<summa2) Z=k2+summa2-min1; else Z=summa1*(max2-min2);
 cout<<"\n\n\n\tso'ralgan Z qiymat="<<Z;                    //asosiy natija
getch();}
```

Natija va uning tahlili.

berilgan tenglamalar sistemasi:

$$\begin{cases} 7.09*C1+1.17*C2-2.23*C3=4.75 \\ 0.43*C1+1.4*C2-0.62*C3=-1.05 \\ 3.21*C1-1.25*C2+2.13*C3=-5.06 \end{cases}$$

1-hisoblashdan keyin:

$$\begin{cases} 7.09*C1+1.17*C2-2.23*C3=4.75 \\ 0*C1-9.4229*C2+3.4369*C3=9.487 \\ 0*C1+12.6182*C2-22.260002*C3=51.122902 \end{cases}$$

2-hisoblashdan keyin:

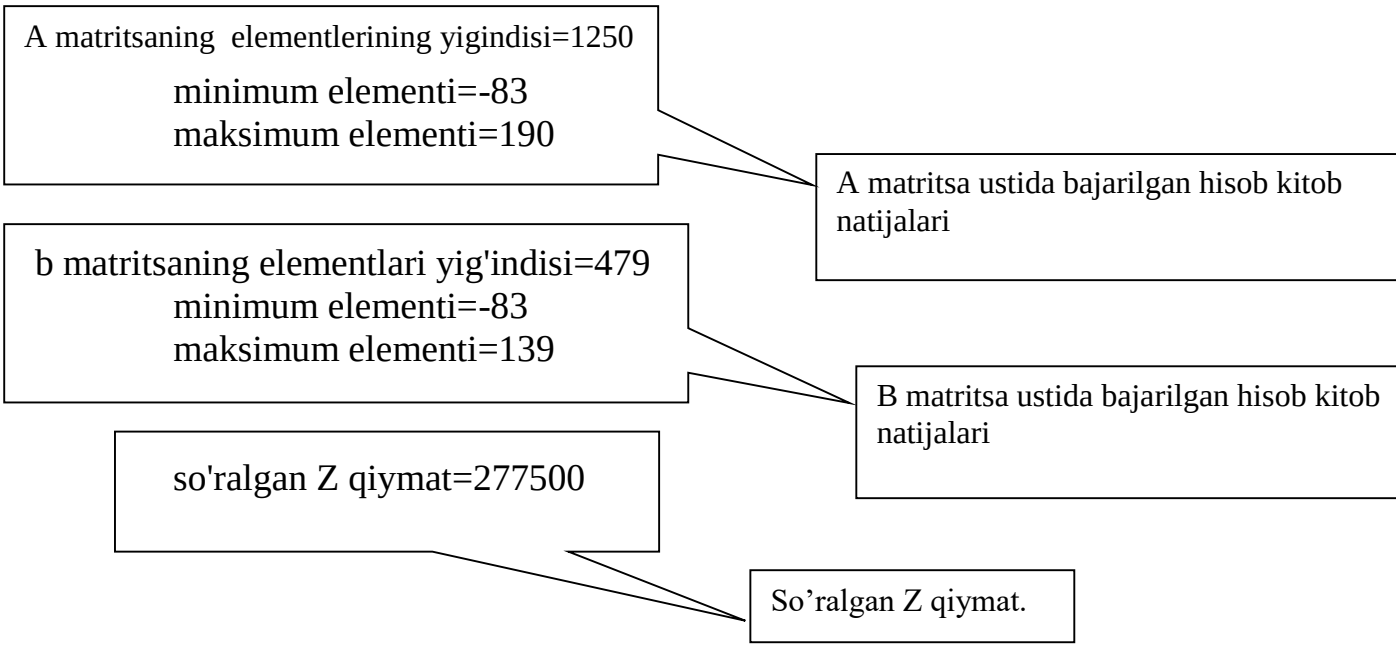
$$\begin{cases} 7.09*C1+1.17*C2-2.23*C3=4.75 \\ 0*C1-9.4229*C2+3.4369*C3=9.487 \\ 0*C1+0*C2-166.386292*C3=601.434875 \end{cases}$$

Tenglamalar sistemasining GAUSS usulidagi yechimi

natija:

C1=-0.083251
C2=-2.325221
C3=-3.61469
ularning o'rta arigmetigi=-2.007721

Tenglama yechimalri va ularning o’rta arifmetigi



Shunday qilib, oldimizga qo’yilgan masala o’z yechimini topdi. Lekin natijalar tahliliga keladigan bo’lsak, so’ralgan Z qiymat har doim har hil qiymatda chiqadi. Chunki bizga A va B matritsalar qiyamti ihtiyoriy ekani ma’lum qilingan. Uning bunday katta qiymat olganiga to’xtaladigan bo’lsak, x<y kengsizliking bajarilishi yoki bajarilmasligiga e’tibor bering, x→20 ta elementli massiv yig’indisi, y→10 ta elementli massiv yig’indisi. Bundan ko’rinib turibdiki x doim y dan katta bo’ladi va har doim Z son ‘Z=summa1*(max2-min2)’ tenglama orqali hisoblanadi. Bunda summa katta chiqishi tabiiy hol, max va min lar ayirmasi ham kamida 2 honali son chiqadi. Shunday ekan, Z ning katta qiymat qabul qilishi tabiiy hol. Chunki A va B massivlarga intelektual yondoshgan holda qiymat berilmasdan, uni kompyuter o’zi istagan holda oladi. U kichikroq, hisob kitobga va tushunishga mos qiymatlar olishi uchun ‘z[i]=rand()/111-100;’ ixtiyoriy qiymatni 111 ga bo’ldim va u manfiy ham qiymatlar olishi uchun 100 ni ayirdim. Natijada Z shunday qiymat qabul qildi.

AMALIY MASHG'ULOTLAR

1-misol $Y=x^3$ funktsyasining yeshimin toping.

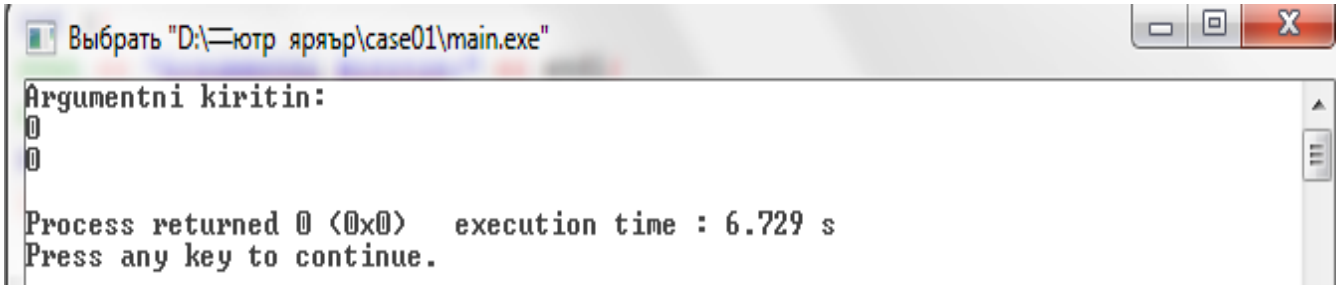
C++ matematik amallaridan foydalanip quyidagi $Y=x^3$ funktsyasining Yeshilishi va funktsya grafigini ko'rip shiqamiz.

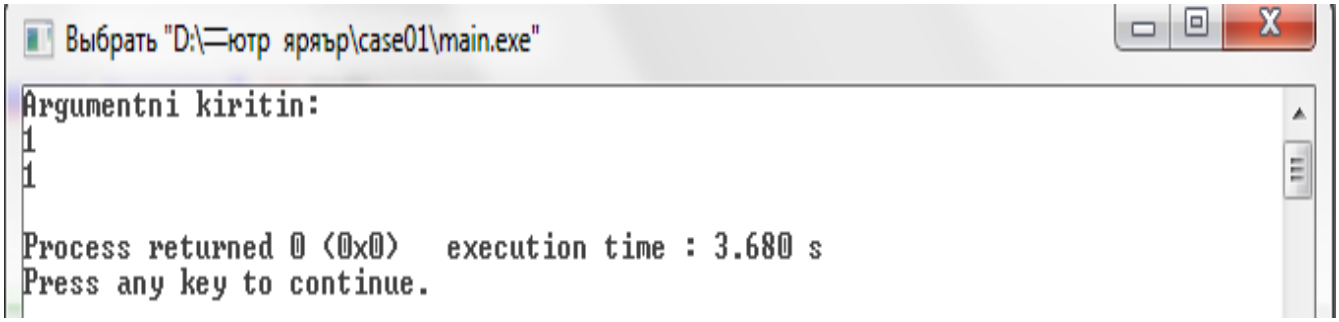
Bu funktsyani yeshishda C++ tilining tanlash operatoridan foydalanamiz.

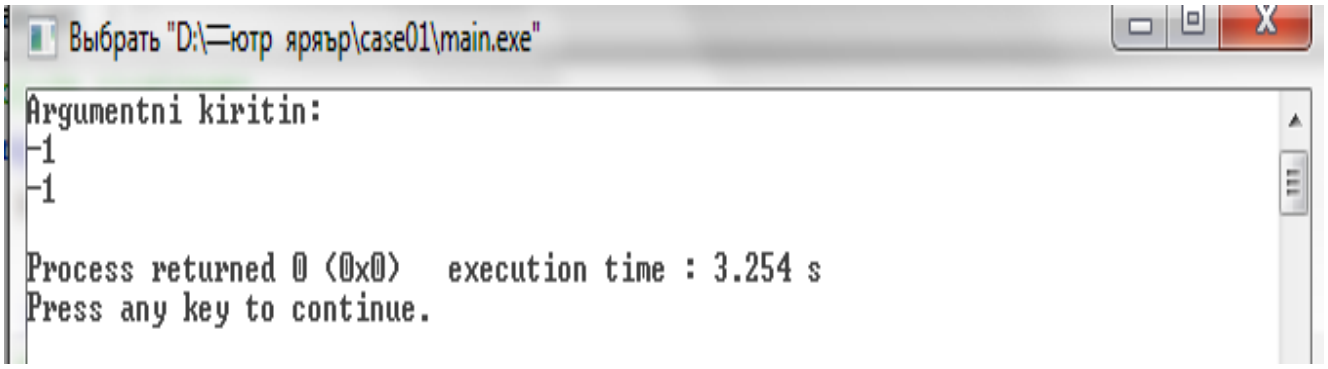
Funktsya kodi:

```
#include <iostream>
using namespace std;
int main()
{
   int x;
   cout << "Argumentni kiritin:" << endl;
   cin >> x;
   switch(x)
   {
      case 1: cout << "1" << endl;
         break;
      case 2: cout << "8" << endl;
         break;
      case 3: cout << "27" << endl;
         break;
      case 0: cout << "0" << endl;
         break;
      case -1: cout << "-1" << endl;
         break;
      case -2: cout << "-8" << endl;
         break;
      case -3: cout << "-27" << endl;
         break;
   }
   return 0;
}
```

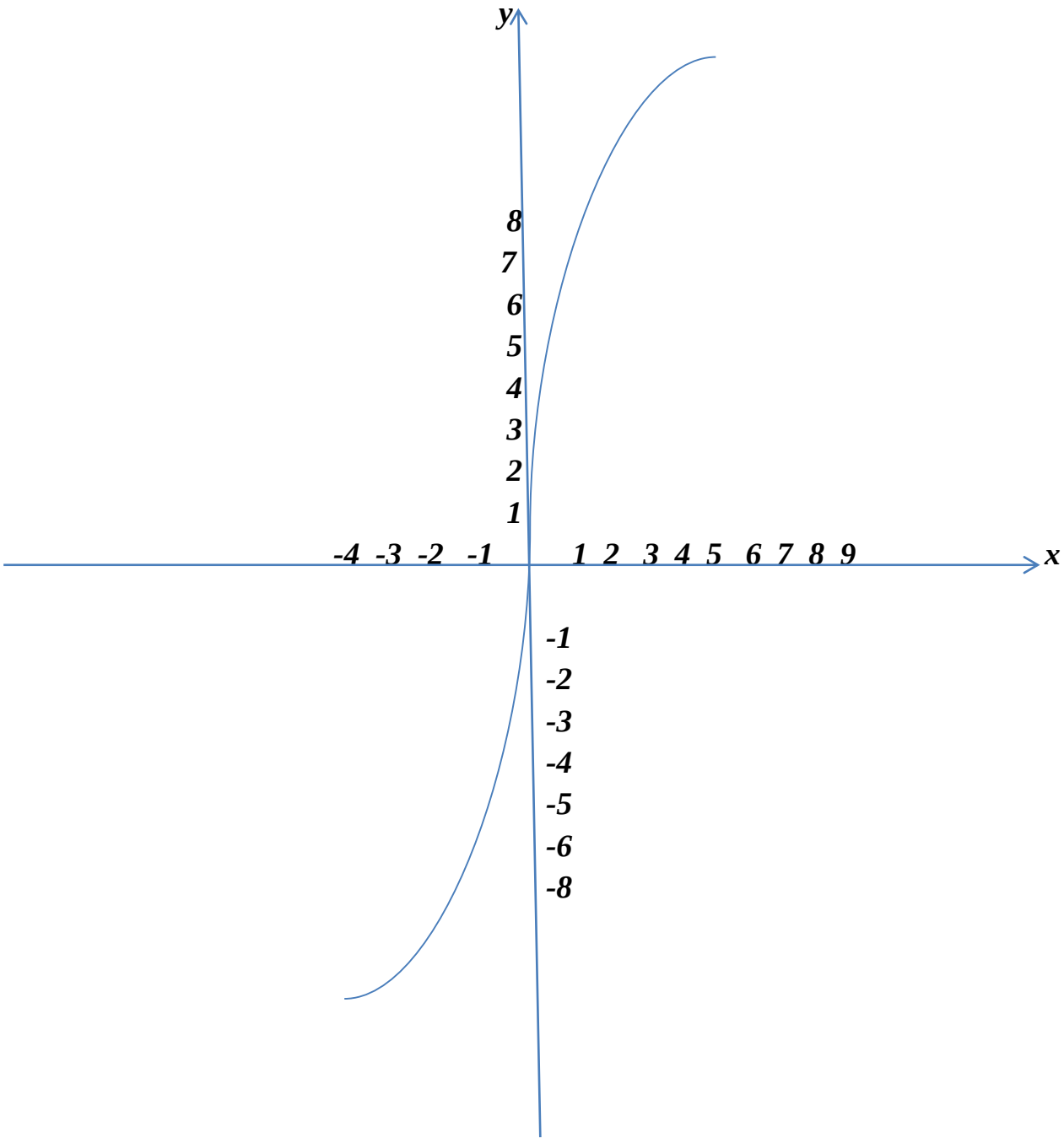
Olingan natijalar quyidagisha:







Funktsya grafigi quyidagicha:



***2*-misol**

$Y=ax^2+bx+c$ kvadrat tenglamasining korenlar sonini toppish va kelip shiqish sabablarini aniqlovshi dastur tuzish.

Bunda biz C++ dasturlash tilining tarmoqlanuvshi algoritmidan poydalanamiz.

Dastur kodi:

```
#include<iostream>
#include<cmath>
using namespace std;
int main()
{ float a,b,c,x,x1,x2,D;
cout<<”a=”; cin>>a;
cout<<”b=”; cin>>b;
cout<<”c=”; cin>>c;
D=b*b-4*a*c;
If (D>0)
{x1=(-b-sqrt(D))/2*a;
  x2=(-b+sqrt(D))/2*a;
cout<<”x1=”<<x1<<endl;
cout<<”x2=”<<x2<<endl;
}
if(D==0)
{ x=-b/2*a;
cout<<”x=”<<x<<endl;}
if(D<0)
{cout<<”yeshimga ega  emas”<<endl;}
return 0;
}
```

Olingan natijalar quyidagisha:

```
-------------- Build: Debug in Baltabaev Jaxangir 1 (compiler: GNU GCC Compiler)---------------

mingw32-g++.exe -Wall -fexceptions  -g     -c "C:\Users\shams\Desktop\Baltabaev Jaxangir 1\main.cpp" -o obj\Debug\main.o
mingw32-g++.exe  -o "bin\Debug\Baltabaev Jaxangir 1.exe" obj\Debug\main.o
Output size is 945.67 KB
Process terminated with status 0 (0 minutes, 1 seconds)
0 errors, 0 warnings (0 minutes, 1 seconds)
```

1.Diskriminant D>0 holat

2.D=0 holat

3.D<0 holat

3-misol.

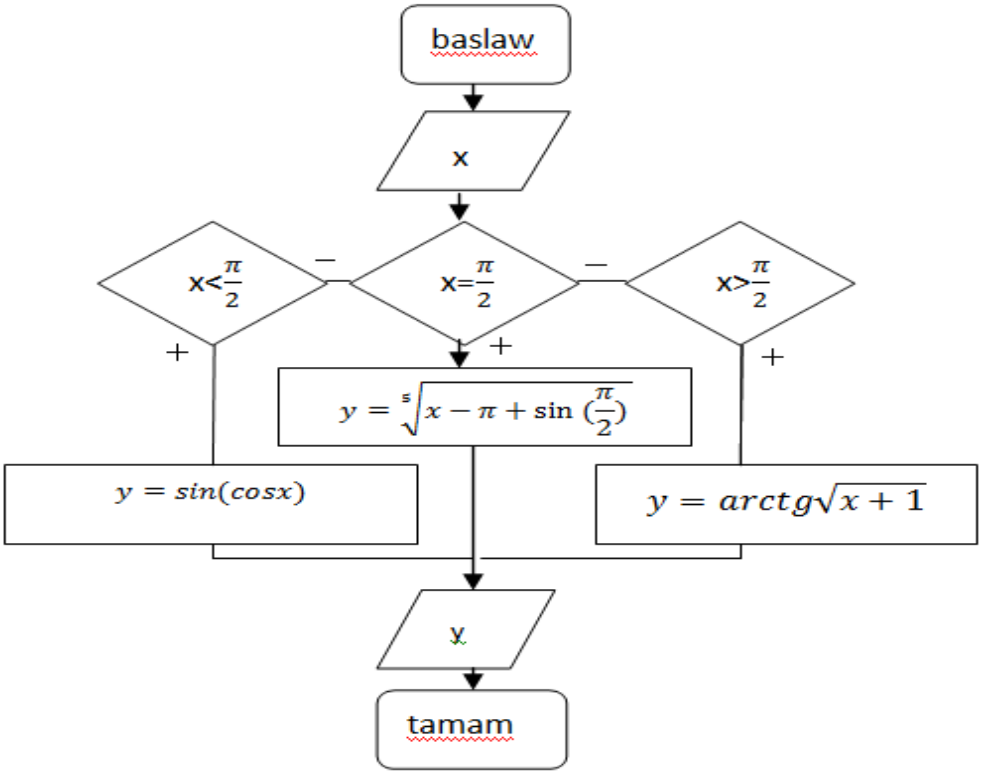
$$y=\begin{cases}\sqrt[5]{x-\pi+\sin\left(\frac{\pi}{2}\right)} & eger\ x=\frac{\pi}{2}\\ arctg\sqrt{x+1} & eger\ x>\frac{\pi}{2}\\ \sin(cosx) & eger\ x<\frac{\pi}{2}\end{cases}$$

Yuqoridagi tenglamalar sistemasining yeshimlarini keltirib otamiz:

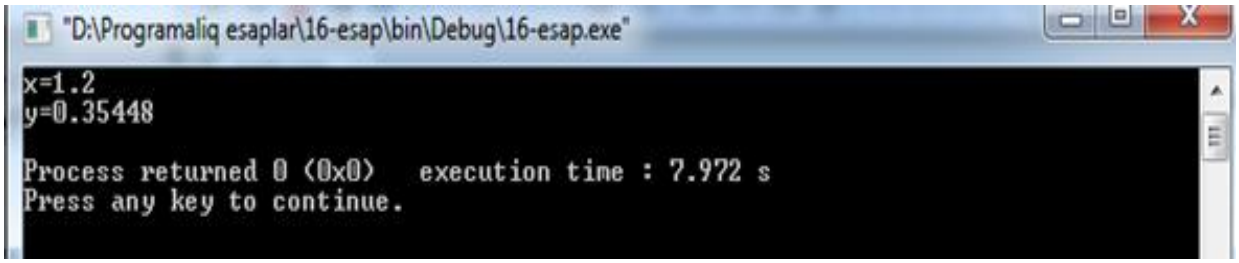
Dastur kodi:

```
#include <iostream>
#include <cmath>
using namespace std;
int main()
{
float x,y;
const float pi=3.14159;
cout<<"x="; cin>>x;
if(x==pi/2)
{
y=pow( x-pi+sin(pi/2), 1/5);
}
if (x>pi/2)
{
y=atan(sqrt(x+1));
}
if (x<pi/2)
{
y=sin(cos(x));
}
cout << "y=" <<y<<endl;
return 0;
}
```

Dastur tanasi:



Olingan natijalar quyidagisha:



XULOSA

Xulosa qilib aytganda, C++ dasturlash tili va unda o`zgarmaslar, o`zgaruvchilar toifalari, hamda ifodalar va operatorlar bilan ishlash xaqida umumiy ma’lumotlarga ega bo`ldim. C++ - va dastur tuzuvchi mehnatini engillashtiradi. C++ dasturlash tili zamonaviy vizual loyihalash texnologiyasi asosida ob`ektga yo`naltirilgan dasturlash nazariyasini hisobga olgan holda tuziladi. Windows muhitida ishlaydigan dastur tuzish uchun qulay bo`lgan vosita bo`lib, komp’yuterda dastur yaratish ishlarini avtomatlashtiradi, xatoliklarni kamaytiradi

Kurs ishida olingan natijalar shuni ko’rsatadiki, dastur to’g’ri ishlayapti va nochiziqli tenglamalar sistemasining yechimlari to’g’ri topilgan va ular bir xil. Aniqlikning oshirish bilan iteratsiyalar soni ham oshib boradi. Agar boshlang’ich yaqinlashish aniq yechimga yaqinroq olinsa yaqinlashish tezligi ortadi va, tabiiyki, iteratsiyalar soni ham kamayadi.

Men C++ dasturi strukturasi xaqida, belgilar bayoni, algoritm va dastur tushunchasi, ma’lumotlarni kiritish va chikarish operatorlari xamda dasturda ishlatiladigan toifalar, ifodalar va operatorlar hamda sinflar va funksiya bilan ishlash xaqida bilim va kunikmalarga ega bo`ldim.

Foydalanilgan adabiyotlar:

1. Бьярн Страуструп. Введение я язык C++

2. Крис Паппас, Уильям Мюррей. Программирование на C и C++

“Ирина”, BHV, Киев 2000г

3. Фридман Александр Львович, Язык программирование C++

4.Aripov M.M., Imomov T., Irmuhamedov Z.M. va boshqalar. Informatika.

5.Axborot texnologiyalari. Toshkent, 1-qism. 2002, 2-qism. 2003

Foydalanilgan elektron saytlar:

http://www.infocity.kiev.ua/

http://www.intuit.ru/
www.google.uz
www.ziyonet.uz
www.elivrary.tuit.uz