TOSHKENT AXBOROT TEXNOLOGIYALARI UNIVERSITETI

Axborot Texnologiyalari fakulteti

INFORMATIKA KAFEDRASI

Informatika va AT fanidan

Maruza matin

Mavzu: **C++ dasturlash tilida massivlar ustida amallar va Tenglamalar sistemasini GAUSS usulida yechish.**

Bajardi:

Tekshirdi:

TOSHKENT– 2013

REJA:

I. Kirrish.

   1. C++ haqida.

II. Asosiy qism:

   1. Qo’yilgan masala.
   2. Dastur haqida qisqacha nazariy ma’lumotlar.
   3. Dastur tanasi.
   4. Natija va uning tahlili.

III. Hulosa.

IV. Foydalanilgan adabiyotlar.

KIRISH

Kompyuter odamlar tomonidan uni, chuqur o'rganish vositasi emas, ko'proq o'zlarining oldilariga qo'yilgan, o'zlarining ishlariga tegishli bo'lgan muammolarini echish instrumenti bo'lib qoldi. Vaqt o'tishi bilan dasturchilar oldiga quyilgan masalalar o'zgarib boryapti. Bundan yigirma yil oldin dasturlar katta hajmdagi ma'lumotlarni qayta ishlash uchun tuzilar edi. Bunda dasturni yozuvchi ham, uning foydalanuvchisi ham kompyuter sohasidagi bilimlar bo'yicha professional bo'lishi talab etilardi.

Dastur so'zi ham komandalarning alohida blokini (berilgan kodini) aniqlovchi so'z, ham yaxlit holdagi bajariluvchi dasturiy mahsulotni belgilovchi so'z sifatida ishlatiladi. Dasturlashga talabni o'zgarishi nafaqat tillarning o'zgarishiga balki uni yozish texnologiyasini ham o'zgarishiga olib keldi. Foydalanuvchilarning ushbu yangi avlodini dasturlar bilan ishlashlarini osonlashtirilishi bilan bu dasturlarning o'zini murakkabligi darajasi oshadi. Zamonaviy dasturlar - foydalanuvchi bilan do'stona munosabatni yuqori darajada tashkil qiladigan ko'p sondagi oynalar, menyu, muloqot oynalari va vizual grafikaviy muhitlardan tarkib topgan interfeysga ega bo'lishi lozim.

Vaqt o'tishi bilan yillarda amaliy dasturchilarga juda ko'p integratsion dastur tuzish muhitlari taklif etilayapti. Bu muhitlar u yoki bu imkoniyatlari bilan bir–biridan farq qiladi. Aksariyat dasturlashtirish muhitlarining fundamental asosi C++ tiliga borib taqaladi.

Interpretator dasturni o'qish jarayonida uning komandalarini ketma - ket mashina tiliga o'tkazadi. Kompilyator esa yaxlit programma kodini biror bir oraliq forma - ob'ekt fayliga o'tkazadi. Bu bosqich kompilyatsiya bosqichi deyiladi. Birinchi elektron hisoblash mashinalari paydo bo'lishi bilan dasturlash tillari evolyutsiyasi boshlanadi. Dastlabki kompyuterlar ikkinchi jahon urushi vaqtida artilleriya snaryadlarining harakat traektoriyasini hisob-kitob qilish maqsadida qurilgan edi. Oldin dasturchilar eng sodda mashina tilini o'zida ifodalovchi kompyuter komandalari bilan ishlaganlar. Bu komandalar nol va birlardan tashkil topgan uzun qatorlardan iborat bo'lar edi. Keyinchalik, insonlar uchun tushunarli bo'lgan mashina komandalarini o'zida saqlovchi (masalan, ADD va MOV komandalari) assembler tili yaratildi. Shu vaqtlarda BASIC va COBOL singari yuqori sathli tillar ham paydo bo'ldiki, bu tillar tufayli so'z va gaplarning mantiqiy konstruktsiyasidan foydalanib dasturlash imkoniyati yaratildi. Bu komandalarni mashina tiliga interpretatorlar va kompilyatorlar ko'chirar edi. Bundan so'ng kompilyator ob'ektli faylni bajariluvchi faylga aylantiradigan kompanovka dasturini chaqiradi.

Interpretatorlar bilan ishlash osonroq, chunki dastur komandalari qanday ketma - ketlikda yozilgan bo'lsa shu tarzda bajariladi. Bu esa dastur bajarilishini nazorat qilishni osonlashtiradi. Kompilyator esa kompilyatsiya va kompanovka kabi qo'shimcha bosqichlardan iborat bo'lganligi uchun ulardan hosil bo'ladigan bajariluvchi faylni tahlil qilish va o'zgartirish imkoniyati mavjud emas. Faqatgina kompilyatsiya qilingan fayl tezroq

bajariladi, chunki bundagi komandalar kompilyatsiya jarayonida mashina tiliga o'tkazilgan bo'ladi.

C++ kabi kompilyatsiya qiluvchi dasturlash tillarini yana bir afzalligi hosil bo'lgan dastur kompyuterda kompilyatorsiz ham bajarilaveradi. Interpretatsiya qiluvchi tillarda esa tayyor dasturni ishlatish uchun albatta mos interpretator dasturi talab qilinadi.

Ko'p yillar davomida dasturlarning asosiy imkoniyati uning qisqaligi va tez bajarilishi bilan belgilanib kelinar edi. Dasturni kichikroq qilishga intilish kompyuter xotirasini juda qimmatligi bilan bog'liq bo'lsa, uning tez bajarilishiga qiziqish protsessor vaqtining qimmatbaholigiga bog'liq edi. Lekin kompyuterlarning narxi tushishi bilan dastur imkoniyatini baholash mezoni o'zgardi. Hozirgi kunda dasturchining ish vaqti biznesda ishlatiladigan ko'pgina kompyuterlarning narxidan yuqori. Hozirda professional tarzda yozilgan va oson ekspluatatsiya qilinadigan dasturlarga talab oshib bormokda. Ekspluatatsiyaning oddiyligi, konkret masalani echish bilan bog'liq bo'lgan talabni ozroq o'zgarishiga, dasturni ortiqcha chiqimlarsiz oson moslashtirish bilan izohlanadi.

C++ HAQIDA

C++ universal dasturlash tili bo'lib, xar xil darajadagi masalalar uchun echim topish mumkin. C++ tilining yana bir axamiyali tomoni shundaki, u ANSI standarti talablariga to'liq javob beradi to'g'rirog'i ANSI standarti talablari asosida yaratilgan. C++ tilining asosiy tushunchalaridan biri bu klasslardir. Klass bu – foydalanuvchi tomonidan yaratilgan (ifodalangan) til. C++ tilida C tilining deyarli barcha imkoniyatlari saqlangan. Bunga asosiy sabab C++ tili yaratilayotgan bir paytda ko'plab foydalanuvchilar C tilida dasturlar yaratgandi. Bunday tayyor xoldagi dasturlarga qayta o'zgarish kiritganda xam C++ kompilyatori dastur matnidan xatoliklar topmaydi. Ya'ni dasturni xar ikkala tilda xam foydalanib tuzish mumkin.

C++ tilini yaratish davomida ko'plab tildagi yaxshi kerakli tomonlari (xususiyatlari) olingan. Jumladan Simula 67 tilidan klasslar konseptsiyasi, Algol-68 dan operatsiyalarni yuklash va o'zgaruvchilarni xoxlagan joyda ta'riflash.

C++ nomi 1983 yil yozida paydo bo'ldi. Dastlabki versiyalari esa 1980 yillardan boshlab "C s klassami" (C klasslar bilan) nomi bilan yuritildi.

C++ nomini Rik Massitti o'ylab topdi. Nomning o'zi ko'rsatib turibdiki, C++ ga C dan bosqichma bosqich o'tilgan. "++" – bu C da operatsiyalarning ortishi.

C++ tili alfaviti har bir tildagi ifodalarni ifodalashda ma'lum bir simvollar majmuasidan foydalaniladi. Masalan: ingliz tilida yozilgan kitobda 26 lotin alifbosi, 10 arab raqamlari va bazi bir tinish belgilari - kombinatsiyalaridan foydalanadi. Shunga o'xshash, C++ tilida lotin alfavitining 26 kichmk harflardan:

abcdefghijklmnopqrstuvwxyz

lotin alfavitining 26 bosh harflaridan:

ABCDEFGHIJKLMNOPQRSTUVWXYZ

va arab raqamlaridan:

0123456789

va quyidagi maxsus belgilardan:

= - * / = , . _ : ; ? ” ’ ~ | ! # % $ & ( ) [ ] { } ^ @ iborat.

Maxsus simvollar bir biridan probel (bo’sh o’rin) bilan ajralib turadi. Probel bilan ajratilmagan bazi simvollar ketma-ketlgi da bir simvol qiymatiga teng.

Masalan: ++ -- ++ && || << >> >= <= ++ -= / =+: :: /* */ //

Identifikatorlar. Dasturda foydalaniladigan o’zgaruvchilar, konstantalar va funktsiyalarning nomlari – identifikatorlar deb ataladi.

Identifikatorlarni harflar, raqamlar va maxsus simvollarning ixtiyoriy uzunlikdagi ketma-ketligdan qurish mumkin. Lekin C++ kompiliyatoriga birinchi boshlangan harfdan 31-chi simvolgacha joylashadi.. Demak, identifikatorlarni 31 simvoldan oshirmaganimiz maqul. Kompilyatorda joylashgan bosh va kichik xarflar bir-biridan farqli simvollar bo’ladi. Masalan NAME va Name identifikatorlari bir biridan farqli, ya'ni ularga yod qurilmasidan har xil katakcha ajratiladi. Lekin, paskal tilida bu identifikatorlar bir-biridan farq qilmaydi. Quyidagi berilgan nomlari yaroqsiz identifikatorlar ekanligini tekshirib ko’risa bo’ladi.

1st_year; #social_sec; Not_Done!;

Birinchi nom raqamdan boshlangani uchun yaroqsiz bo’ladi. Ikkinchi nom # simvoli bilan boshlangani uchun, uchunchi esa nom yaroqsiz simvol bilan yakunlangani uchun yaroqsiz identifikator bo’ladi.

Konstantalar. C++ tilida konstantalarni foydalanishning ikki turi. Konstanta– bu dastur bajarilish jarayonida qiymatini o’zgartirmaydigan identifikator bo’lib topiladi.

1. Const kvalifikatori yordamida konstantalarni quyida berilgan misoldagidek etib foydalanamiz. Misoli: Const float pi=3.14159; Const int iMin=1, iSale=25;

2. #define direktivasi yordamida konstantalarni foydalanish quyidagicha:

#define Sales_Team 10 //Bu erda Sales_Team konstantaning nomi, 10 uning qiymati.

ASOSIY QISM

Qo’yilgan masala.

Z sonini quyidagi formula yordamida hisovlsh kerak:

- $z\begin{cases} c+y-a_{min}; \text{ agar } x<y \\ x*(b_{max}-b_{min}); \text{ agar } x>=y; \end{cases}$

bu yerda x= $\sum_{i=1}^{20} a_i$; a_i→A(20); y= $\sum_{i=1}^{10} b_i$; b_i→b(10);

Massivning eng katta va eng kichik qiymatini topish qism dastur yordamida tashkil etilsin. Massiv elementlarining son qiymatlari ixtiyoriy. C- tenglamalr sistemasi yechimlarining o’rtacha arifmetik qiymati. Tenglamalar sistemasi GAUSS usulida yechilsin.

$$\begin{cases} 7.09C_1 + 1.17C_2 - 2.23C_3 = 4.75 \\ 0.43C_1 + 1.4C_2 - 0.62C_3 = -1.05 \\ 3.21C_1 - 1.25C_2 + 2.13C_3 = -5.06 \end{cases}$$

Dastur haqida qisqacha nazariy ma’lumotlar.

Ko’rib turganingizdek, oldimizga qo’yilgan masalada bizdan so’ralayotgan ‘Z’ soni x va y ning bir biriga munosabati tufayli 2 hil bo’lishi kumkin. Demak biz avvalo if sturkturasiga alohida to’xtalishimiz shart.

if STRUKTURASI

Biz shartga ko'ra bir necha harakat yo'lidan bittasini tanlaymiz. Misol uchun: agar bolaning yoshi 7 ga teng yoki katta bo'lsa u maktabga borishi mumkin bo'lsin. Buni C++ da if ni qo'llab yozaylik.

```
if (yosh >= 7)    maktab();
```

Bu yerda shart bajarilishi yoki bajarilmasligi mumkin. Agar yosh o'zgaruvchisi 7 ga teng yoki undan katta bo'lsa shart bajariladi va maktab() funksiyasi chaqiriladi. Bu holat true (to'g'ri) deyiladi. Agar yosh 7 dan kichik bo'lsa, maktab() tashlab o'tiladi. Yani false (noto'g'ri) holat yuzaga keladi. Aslida esa shartdagi ifodaning ko'rinishi muhim emas – agar ifodani nolga keltirish mumkin bo'lsa false bo'ladi, noldan farqli javob bo'lsa, musbatmi, manfiymi, true holat paydo bo'ladi va shart bajariladi. Bunga qo'shimcha qilib o'tish kerakki, C++ da mahsus bool tipi mavjud. Bu tipdagi o'zgaruvchilarning yordamida

bul (mantiqiy) arifmetikasini amalgam oshirish mumkin. bool o'zgaruvchilar faqat true yoki false qiymatlarini olishlari mumkin.

if/else STRUKTURASI

if ni qo'llaganimizda ifoda faqat shart haqiqat bo'lgandagina bajariladi, aks holda tashlanib o'tiladi. if/else yordamida esa shart bajarilmaganda (false natija chiqqanda) else orqali boshqa bir yo'ldan borishni belgilash mumkin. Misolimizni takomillashtirsak. Bola 7 yosh yoki undan katta bo'lsa maktabga, 7 dan kichkina bo'lsa bog'chaga borsin.

```
if (yosh >= 7)   maktab(); //nuqta-vergul majburiydir

else   bogcha();
```

Yuqorida if ga tegishli bo'lgan blok bitta ifodadan (maktab()) iborat. Shu sababli nuqta-vergul qo'yilishi shart. Buni aytib o'tishimizning sababi, masal Pascalda hech narsa qo'yilmasligi shart. C++ da bitta ifosa turgan joyga ifodalar guruhini {} qavslarda olingan holda qo'ysa

bo'ladi. Masalan:

```
if (yosh >= 7){   cout << "Maktabga!\n";   maktab(); }

else{   cout << "Bog'chaga!\n";   bogcha();  }
```

Aslida har doim {} qavslarni qo'yish yahshi odat hisoblanadi; keyinchalik bir ifoda turgan joyga qo'shimcha qilinganda qavslardan biri unutilib qolmaydi.

Strukrurali dasturlashning yana bir harakterli joyi shundaki tabulyatsiya, bo'sh joy va yangi satrlar ko'p qo'llaniladi. Bu programmani o'qishni osonlashtirish uchun qilinadi. C++ uchun bo'sh joyning hech ahamiyati yo'q, lekin dasturni tahrir qilayatgan odamga buyruqlar guruhini, bloklarni tabulyatsiya yordamida ajratib bersak, unga katta yordam bo'ladi. Yuqoridagini quyidagicha ham yozish mumkin:

```
if(yosh>=7){cout<<"Maktabga!\n";maktab()}else{cout<<"Bog'chaga!\n";bogcha()};
```

Biroq buni o'qish ancha murakkab ishdir.

C++ da if/else strukturasiga o'hshash ?: shart operatori (conditional operator) ham bordir. Bu C++ ning bittagina uchta argument oluvchi operatori. Uch operand va shart operatori shart ifodasini beradi. Birinchi operand orqali

shartimizni beramiz. Ikkinchi argument shart true (haqiqat) bo'lib chiqqandagi butun shart ifodasining javob qiymatidir. Uchinchi operand shartimiz bajarilmay (false) qolgandagi butun shart ifodasining qiymatidir.

if/else strukturalarini bir-birining ichida yozishimiz mumkin. Bunda ular bir-biriga ulanib ketadi. Misol uchun tezlikning kattaligiga qarab jarimani belgilab beruvchi blokni yozaylik. if (tezlik > 120) cout << "Jarima 10000 so'm";

else if (tezlik > 100) cout << "Jarima 7000 so'm";

else if (tezlik > 85) cout << "Jarima 3000 so'm";

else cout << "Tezlik normada";

Agar tezlik 120 dan katta bo'lsa birinchi if/else strukturasining haqiqat sharti bajariladi. Va bu holda albatta tezlik o'zgaruvchimizning qiymati ikkinchi va uchinchi if/else imizni ham qoniqtiradi. Lekin solishtirish ulargacha bormaydi, chunki ular birinchi if/else ning else qismida, yani noto'g'ri javob qismida joylashgandir. Solishtirish birinchi if/else da tugashi (aynan shu misolda)tanlash amalini tezlashtiradi. Yani bir-biriga bog'liq if/else lar alohida if struktura-lari blokidan tezroq bajarilishi mumkin, chunki birinchi holda if/else blokidan vaqtliroq chiqish imkoni bor. Shu sababli ich-ichiga kirgan if/else lar guruhida true bo'lish imkoni ko'proq bo'lgan shartlarni oldinroq tekshirish kerak.

Bundan tashqari SWITCH strukturasi ham mavjud bo’lib, bu struktura shartlar 3 va undan ortiq hollarda ishlatilishi maqsadga muvofiq. Biz bu strukturaga qo’yilgan masaladan chetga chiqib ketmaslik uchun alohida to’xtalib o’tirmaymiz.

O’z navbatida x va y qiymatlar A va B massivlarning hadlari yig’indisi bo’lgani uchun biz massiv tushunchasi, hadlari yig’indisi, eng katta va eng kichik qiymatlar haqida ma’lumot berishimiz kerak.

MASSIV TUSHUNCHASI

Massiv - o’zida bir turga tegishli ma'lumotlarni, tartiblangan ko’rinishda saqlovchi o’zgaruvchi sifatida qarasa bo’ladi. Massivning xar bir elementiga uning adresi bo’yicha murojaat kilish mumkin.

Massiv – bir turdagi ma'lumotlarning tartiblangan ko’rinishi.

C va C++ tillarida massivlar bilan ishlash va tuzilishi deyarli bir xil.

Massiv xossalari

Massivlar bilan ishlashda uning xossalariga aloxida e'tibor berishga to’gri keladi. Shularni xisobga olib quyida massivning asosiy xossalariga to’xtalib o’tamiz.

Massivda element deb nomlanuvchi aloxida qiymatlar saqlanadi

Massivning barcha elementlari bir xil turga tegishli bo’lishi lozim

Massivning barcha elementlari xotirada ketma-ket joylashadi va birinchi element nolinchi indeksga ega bo'ladi

Massiv nomi o'zgarmaydi, ya'ni dastur o'rinlanishi davomida oldindan ko'rsatilgan nom bilan foydalaniladi.

Massivlarni e'lon qilish. Massivlarni ta'riflash (e'lon qilish) o'zgarivchilarni e'lon qilishga o'xshab ketadi. Farqi massiv nomidan keyin kvadrat qavslar ichida, massiv xajmini ko'rsativchi uzgarmas ifoda beriladi. Masalan

```
int butun_mas[15]; // 15 ta butun sondan iborat massiv

char simral_mas [10]; // 10 ta simvoldan iborat massiv
```

Demak massiv elemenlari soni oldidan aniq bo'lishi lozim. Chunki kompilyator massiv elemenlari uchun xotiradan tegishli joy ajratadi. Shunday qilib dastur o'rinlanishi davomida massiv xajmi o'zgarishi mumkin emas. Shuning uchun ko'pincha massiv elemenlari sonini ko'rsatish o'zgarmas miqdor (konstanta) lardan foydalinadi.

O'zgarmas miqdorlardan foydalanishning axamiyati shundaki, massivda mavjud bo'lmagan elementlarga murojat qilishdan kelib chiqadigan xatoliklarni chetlab o'tish imkoniyatini beradi.

Massiv elementlariga qiymat berish va murojat qilish. C/C++ tillarida massivlarga qiymat boshqa dasturlash tillaridagidek bir nechta usullari mavjud. Shuning bilan birga ayrim o'zgachiliklarga ham ega. Jumladan Turbo Pascal tilida massiv elementlariga hech qanday qiymat bermasdan elementlar qiymatini ekranga chiqarib ko'radigan bo'lsak , massiv turiga mos har hil qiymatlarni chiqarishi mumkin. C++ tilida esa statik va umumiy (global) masalalarda bu xollar kuzatilmaydi, ya'ni tinch xolatida daslabki elemenlari uchun nol qiymati qabul qilinadi.

Dastur o'rinlanishi natijasida ekranda massiv elementlariga tinch xolatida qabul qilingan qiymatlari , nollar xosil bo'ladi . Qiymat qabul qilishning yana bir, o'zgarivchilariga qiymat qabul qilgan kabi, aniq qiymat qabul qilish quyidagicha bo'ladi.

```
Int butun_mas [ 5 ] = {10,-3,0,4,1};

Float haqiqiy_mas [ 3 ] = {3.1417, 2.7 , 0.25 };

static int butun_hato [ 2 ] = {9,8,7,6,5,4,3,2,1};
```

Bularga izox beradigan bo'lsak birinchi satirda o'zida 5ta butun sonlarni jamlashtirgan butun_mas massivi xosil qilinib , dastur o'rinlanishi bilan massiv elementlarining qiymatlari yacheyka(katakcha)lariga eziladiIkkinchi satir ham xuddi yuqoridagi kabi. Uchinchi satirda ko'rsatilganidan elementlar sonidan ko'p qiymatlar

berilgan. Bunday xolatda hatolik sodir bo'ladi. Agar qiymatlar ko'rsatilgan elementlar sonidan kam bo'lgan xolatlar elementlarning qolgan qismi nol qiymatini qabul qiladi.

Endi massiv elementlariga murojat qilish qonun-qoyidalariga to'xtalib o'tamiz. Bizga ma'lum , massivga murojat qilishda massiv elementining tartib nomerlarin ifodalovchi indeksni ko'rsatish lozim . Bu erda qatiy e'tibor berishimiz kerak bo'lgan narsa, massivning birinchi elementining indeksi hamma vaqt noldan boshlanadi. Boshlavchi programmistlar ko'pincha birinchi element indeksi 1ga teng deb hatoga yo'l qo'yishadi.

Endi quyidagi dasturga to'xtalib o'taylik. Massiv elementlarining ichida eng katta va eng kichik elementlarini topish uchun dastur tuzing.

```
# include < >

# define mas_max 10

main ( )

{int mas [mas_max ] = { 4,-3,0,9,7,10,2,-1,15,5 }

 int min qmas [0] , max q mas [0] ;

 int index ;

 for (index = 1 ; index < mas_max ; index ++)

 {if (mas [index] < min) min = mas [index] ;

  if (mas [index] > max) max = mas [index] ; }

  cout <<" eng katta element '' << max << " ga teng\n" ;

  cout <<" eng kichik element'' << min << '' ga teng\n'';}
```

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

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

```
char c[3][4] = {{  2,  3,9, 5}, // birinchi satr qiymatlari

                {-10, 77,5, 1}, // ikkinchi  "      "

                {90,233,3,-3}};  // uchinchi  "      "
```

int m[2][2] = {56,77,8,-3}; //oldin birinchi satrga qiymatlar beriladi, keyin ikkinchi satrga

double d[4][3][6] = {2.55, -46,0988}; // birinchi satrning dastlabki ikkita elementi qiymat

//oladi, massivning qolgan elementlari esa nolga tenglashtiriladi

Massivning har bir indeksi alohida [] qavslar ichiga olinishi kerak. Yuqoridagi c[][] massivining ikkinchi satr, birinchi ustunidagi elementi qiymatini birga oshirish uchun

++c[1][0]; // yoki c[1][0]++; // c[1][0] += 1; // c[1][0] = c[1][0] + 1;

deb yozishimiz mumkin. Massiv indekslari 0 dan boshlanishini unutmaslik zarur.

Agar ++c[1,0]; deb yozganimizda hato bo'lar edi. C++ bu yozuvni ++c[0]; deb tushunar edi, chunki kompilyator vergul bilan ajratilgan ro'yhatning eng ohirgi elementini qabul qilardi. Hullas, C++ dagi ko'p indeksli massivlar dasturchiga behisob imkoniyatlar beradi. Undan tashqari, ular hotirada statik joylashganligi uchun ularning ishlash tezligi kattadir. C++ dagi ko'p indeksli massivlar hotirada ketma-ket joylashgandir. Shu sababli agar massiv funksiyaga kirish parametri sifatida berilsa, faqat birinchi indeks tushurilib qoldiriladi, qolgan indekslar esa yozilishi shartdir. Aks taqdirda funksiya massiv kattaligini to'g'ri keltirib chiqarolmaydi.

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

Oldimizga qo’yilgan masalada massiv eng kichik va eng katta elementlari qism dastur yordamida tuzilishi shart. Shuning uchun biz funksiya va qism dastur haqida ma’lumot beramiz.

FUNKTSIYALAR. QISM DASTURLAR.

C/C++ dasturlash tillari dastur kodining asosiy qismini funktsiyalar tashkil etadi. Ular dasturni bir necha bloklarga bo’lish imkoniyatini beradi. Bizga ma'lumki bu tillardagi ixtiyoriy dastur main( ) funktsiyasini o’zida mujassamlashtiradi. Funktsiyalarni yaxshi tayorlashi, dasturning effektli va ishonchli ishlashini ta'minlaydi.

Funktsiyalar tuzilishining umumiy ko’rinishi.

Funktsiya dasturining nom berilgan shunday qismiki, unga dasturning boshqa Funktsiyalarning tuzilish usuliga to'xtalib o'tishdan oldin funktsiya tushinchasiga bir bo'limidan qancha talab qilinsa, shuncha murojat qilish mumkin.

ANSI C standartiga mos holda xoxlagan funktsiyaning argument turlari va chiqarivchi tur nomi oldindan (asosiy blokdan oldin) elon qilinishi lozim. Quyida funktsiyaning umumiy holda elon qilamiz:

Natija _turi funktsiya_nomi (argument_turi shart_bo'lmagan_argument_nomi [,…]);

Funktsiya void, int, float, char, va h.k turdagi qiymatlardan birini berishi mumkin (qaytarishi). Ushbu ko'rinish main() funktsiyasi (asosiy dastur) dan oldin yoziladi.

Funktsiya kodi main ( ) funktsiyasidan avval yoki keyin yoziladi:

Natija_turi funktsiya_nomi (argument_turi argument_nomi [,…])

{…..

funktsiya_tanasi

}…..

Etibor berib qarasak bu erda funktsiyaning nomlanish satri bilan elon qilish satri orasida kichgina farq bor: nomlanish satiri da ” ; ” qo'yilmagan.

Ba'zi masalalarda funksiya tashkil etilganda funksiyani tashkil etadigan parametr qiymatalrini o'zgartirishga to'g'ri keladi, ya'ni natija 1 emas, birnecha chiqishi kerak bo'ladi. Bunday funksiyalarni proceduralar deb yuritiladi. Procedura parametrlari qatorida natijalar nomlari ham ko'rsatiladi. Shuning uchun procedura tashkil qilayotganimizda uning toifasini void deb ko'rsatgan ma'qul (return kerak bo'lmaydi);

Proceduraga murojat qilganda '=' kerak emas. Procedura tashkil qilishda agar natijalar birne4ta bo'lsa ko;rsatkichlardan foydalaniladi. Ko'rsatkich – bu biror o'zgaruvchining adresini o'zida saqlovchi kattalik. Ko'rstakichni e'lon qilishda <toifa *o'zgaruvchi nomi> dan foydalaniladi. Ko'rsatkichlardan foydalanilganda ylarni osonlashtirish uchun 'adresni ol' (&) belgisi orqali ham amalgam oshirsa bo'ladi. Lekin bu amal faqat o'zgaruvchilarga qo'llanadi, songa qo'lllash mumkin emas. Proceduralar hosil qilishda * va & amallaridan yoki to'g'ridan to'g'ri & operatsiyasi orqali ishlatishimiz mumkin.

Misol:

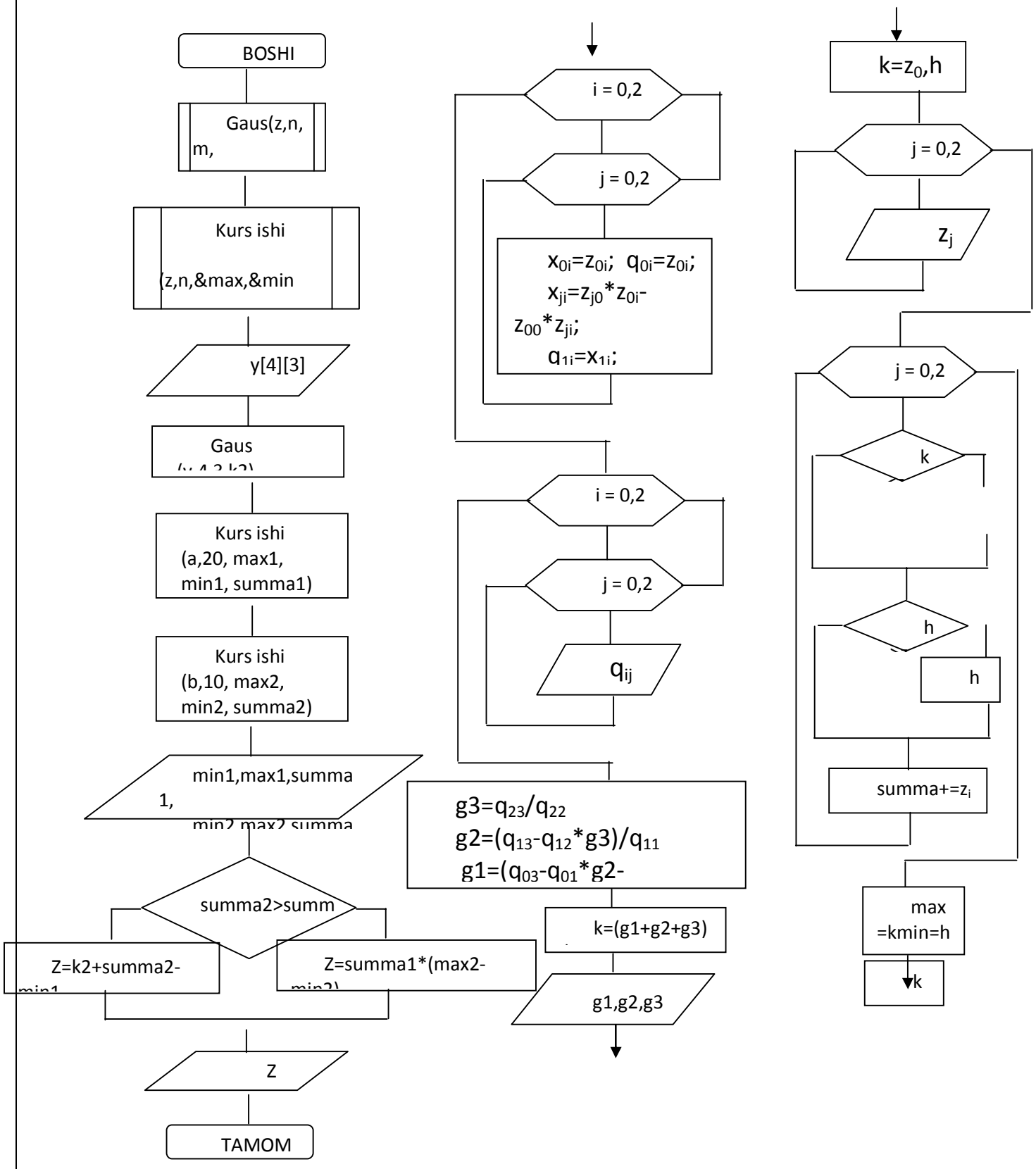
```
Void top(float a, float b, float *s,float *p)
{*s=a*b; *p=2*(a+b);}
main()
```

```
{ —//—
  top (23,4,&s1,&p1); cout<<s1<<p1;}
yoki
Void top(float a, float b, float &s,float &p)
{s=a*b; p=2*(a+b);}
main()
{ —//—
  top (23,4,s1,p1); cout<<s1<<p1;}
```

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.


BOSHI
Gaus(z,n,
m,
Kurs ishi
(z,n,&max,&min
y[4][3]
Gaus
Kurs ishi
(a,20, max1,
min1, summa1)
Kurs ishi
(b,10, max2,
min2, summa2)
min1,max1,summa
1,
summa2>summ
Z=k2+summa2-
Z=summa1*(max2-
Z
TAMOM
i = 0,2
j = 0,2
x0i=z0i; q0i=z0i;
xji=zj0*z0i-
z00*zji;
q1i=x1i;
i = 0,2
j = 0,2
qij
g3=q23/q22
g2=(q13-q12*g3)/q11
g1=(q03-q01*g2-
k=(g1+g2+g3)
g1,g2,g3
k=z0,h
j = 0,2
zj
j = 0,2
k
h
h
summa+=zi
max
=kmin=h
k

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\t213-08 GURUH O'QUVCHISI SODIQOV SAIDAKBARNING 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\t213-08 GURUH O'QUVCHISI SODIQOV SAIDAKBARNING 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();}
```

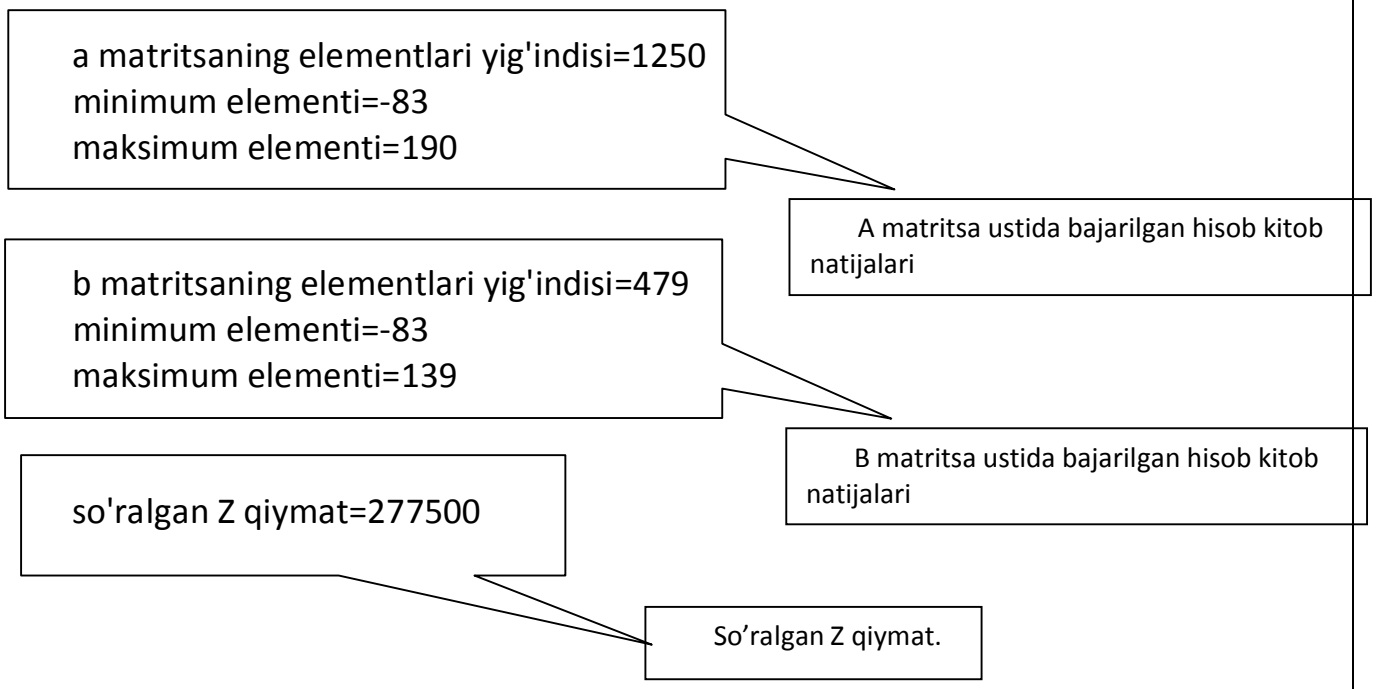
Natija va uning tahlili.

```
berilgan tenglamalar sistemasi:
┌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
1-hisoblashdan keyin:
┌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
2-hisoblashdan keyin:
┌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
```

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

Tenglamalar sistemasining GAUSS usulidagi yechimi

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.

Hulosa.

Biz hozir gauss usulida berilgan 3 noma’lumli istalgan tenglamani yecha oladigan, istalgancha qiymatli massivni summasi, maksimal va minimal elementlarini hisoblash uchun xizmat qiladigan dastur tuzdik. Bu insonni ortiqcha ovoragarchilik yoki hisob kitobdagi chalkashib ketish yoki shunga o’xshash tushunmovchiliklardan qutqaradi. Balki bu dasturning ba’zi kamchiliklari bo’lishi, u ba’zi qiymatlarda noto’g’ri natija chiqarishi yoki shunga o’xshash noqulayliklar yaratishi mumkin. Ishonamanki, u boshqa shunga o’xshash masalalarni hal qilishda meni, ya’ni o’z egasini yuzini yerga qaratmaydigan dasturdir. Lekin har holda u oldimizga qo’yilgan masalani yechishda katta yordam berdi.

FOYDALANILGAN ADABIYOTLAR.

1. Jess Liberti, “Освой самостоятельно C++ за 21 день”, Sankt Peterburg 2000, 815 c.
2. Liberti D. Освой самостоятельно C++: 10 минут на урок. Ingl. Tarjima. Vilyams, 374 стр,2004 г.
3. Sayfiev J. F., «C++ tiliga kirish», Buxoro 2004 y.
4. Shmidskiy Ya.K. Прораммирование на языке C++: Самоучитель. Учебное пособие. Диалектика. 361 стр, 2004 г.
5. Kimmel P., «Borland C++5» . – СПб.: BHV, 1997.
6. Bryan Straustrap. Введение я язык C++ http://www.infocity.kiev.ua/
7. Kris Pappas, Uilyam Myurrey. Программирование на C и C++ “Ирина”, BHV, Kiev 2000 y.
8. Fridman Aleksandr L’vovich, Язык программирование C++ http://www.intuit.ru/