MUTAXASSISLIK MASALARINI YECHISHDA MASSIVLARDAN FOYDALANISH

3.5.1. Massivlar

Kundalik hayotimizda ko’p turdagi jadvallardan foydalanamiz:

Dars jadvali, shaxmat yoki futbol o’yinlari bo’yicha musobaqa jadvali, Pifagor(karra) jadvali, kelishiklar jadvali, coha masalalarida ishlatiladigan ma’lumotlarining jadvallari va boshqalar. Jadvalni tashkil etuvchi ma’lumotlar uning elementlari deb ataladi.

C++ dasturlash tilida jadvaldagi ma’lumotlarni ifodalash, umuman bir turdagi bir necha o’zgaruvchini ta’riflab guruhlash, hamda shu ma’lumotlar bilan ishlash qulay bo’lishi uchun massiv tushunchasi kiritilgan.

Xotirada ketma-ket (regulyar) joylashgan bir xil turdagi qiymatlarga massiv deb ataladi.

Odatda massivlarga zarurat, katta hajmdagi, lekin cheklangan miqdordagi va tartiblangan turdagi qiymatlarni qayta ishlash bilan bog`liq masalalarni yechishda yuzaga keladi. Faraz qilaylik, talabalar guruhining reyting ballari bilan ishlash masalasi qo`yilgan. Unda guruhning o`rtacha reytingini aniqlash, reytinglarni kamayishi bo`yicha tartiblash, konkret talabani reytingi haqida ma’lumot berish va boshqa masala ostilarini yechish zarur bo`lsin. Qayd qilingan masalalarni yechish uchun berilganlarning (reytinglarning) tartiblangan ketma-ketligi zarur bo`ladi. Bu yerda tartiblanganlik ma’nosi shundaki, ketma-ketlikning har bir qiymati o`z o`rniga ega bo`ladi (birinchi talabaning reytingi massivda birinchi o`rinda, ikkinchi talabaniki - ikkinchi o`rinda va hokazo). Berilganlar ketma-ketligini ikki xil usulda hosil qilish mumkin. Birinchi yo`l - har bir reyting uchun alohida o`zgaruvchi aniqlash - reyting1, reyting2, … reytingn. Lekin, guruhdagi talabalar soni yetarlicha ko`p bo`lganda, bu o`zgaruvchilar qatnashgan dasturni tuzish katta texnik qiyinchiliklarni yuzaga keltiradi. Ikkinchi yo`l - berilganlar ketma-ketligini yagona nom bilan aniqlab, uning qiymatlariga murojaatni, shu qiymatlarning ketma-ketlikda joylashgan o`rnining nomeri (indeksi) orqali amalga oshirishdir. Reytinglar ketma-ketligini reyting deb nomlab, undagi qiymatlariga 1 reyting1 , 2

reyting2,…,reyting n ko`rinishida murojaat qilish mumkin. Odatda berilganlarning bunday ko`rinishiga massivlar deyiladi. Massivlarni matematikadagi sonlar vektoriga o`xshatish mumkin, chunki vektor ham o`zining individual nomga ega va tuzilma fiksirlangan miqdordagi bir turdagi qiymatlardan - sonlardan iboratdir.

Demak, massiv - bu fiksirlangan miqdordagi ayrim qiymatlarning (massiv komponentalari) tartiblangan majmuasidir. Barcha komponentalar bir xil turda bo`lishi kerak va bu tur komponent turi yoki massiv uchun asos tur deb nomlanadi. Yuqoridagi keltirilgan misolda reyting - haqiqiy turdagi vektor deb nomlanadi.

Dasturda ishlatiladigan har bir konkret massiv o`zining individual nomiga ega bo`lishi kerak. Bu nomni to`liq o`zgaruvchi deyiladi, chunki uning qiymati butun massiv bo`ladi. Massivning har bir komponentasi massiv nomi, hamda kvadrat qavsga olingan va komponent selektori deb nomlanuvchi indeksni ko`rsatish orqali oshkor ravishda belgilanadi. Murojaat sintaksisi:

<massiv nomi >[<indeks>].

Massivni tashkil etgan o’zgaruvchilar uning elementlari deb ataladi.

Elementning tartib nomerini ko’rsatuvchi kattalik shu elementning indeksi deb ataladi va u o’rta qavs yordamida yoziladi, ya’ni Massiv quyidagi ko`rinishda yoziladi: **o'zgaruvchi_turi massiv_nomi[massiv_uzunligi].**

O'zgaruvchining turidan keyin massiv nomi, so'ngra kvadrat qavslarda uning o'lchamlari ko`rsariladi. Massivlar bir, ikki va ko’p o’lchovli b’lishi mumkin.

Bir o’lchovli massivlar

Bir o’lchovli massivlar C++ tilida quyidagicha tavsiflanadi: ***tur*** *identifikator*[*o’lchami*];

Bu yerda ***tur*** – massivning elementlarining turi(barcha ma’lumot bitta umumiy turda ifodalanadi), *identifikator* – massivning nomi, *o’lchami* – massiv elementlarining soni.

Masalan: **int a[5]** – **a**-massiv nomi, **5** massiv elementlari soni, **int** massiv elementlarining turi butun tip.

Masalan, 4 ta sondan iborat massivni aniqlaymiz: **int mas [4].** Ushbu massiv to'rtta elementga ega, ammo ularning qiymatlari aniqlanmagan. Bu elementlarga o`rta qavslar orqali boshlang'ich qiymatlarni berishimiz mumkin: ***int mas [4] = {1,2,3,4};***

Masalan, A[10], X[100], b[123] bir o’lchovli massivlarlar dasturda quyidagicha tavsiflanadi:

```
# include <iostream>
Using namespace std;
Int main()
{
Int a[10];
Double x[10x10];
Bool b[123];
Return 0;
}
```

C++ da massiv elementlari 0-indeksdan boshlanadi.

Bu ko`rinishga xususiy o`zgaruvchi deyiladi, chunki uning qiymati massivning alohida komponentasidir. Bizning misolda reyting massivining alohida komponentalariga reyting[1], reyting[2],…, reyting[n] xususiy o`zgaruvchilar orqali murojaat qilish mumkin. Boshqacha bu o`zgaruvchilarni indeksli o`zgaruvchilar deyiladi.

Umuman olganda indeks sifatida ifoda ishlatilishi mumkin. Ifoda qiymati massiv komponentasi nomerini aniqlaydi. Ifodaga o`zgaruvchi ham kirishi mumkinki, o`zgaruvchi qiymatini o`zgarishi bilan murojaat qilinayotgan massiv komponentasi aniqlovchi indeks ham o`zgaradi. Shunday qilib, dasturdagi bitta indeksli o`zgaruvchi orqali massivning turli komponentalarini belgilash mumkin. Masalan, reyting[i] o`zgaruvchisi orqali i o`zgaruvchining qiymatiga bog`liq ravishda reyting massivining turli (barcha) komponetalariga murojaat qilish mumkin. Shuni qayd qilish kerakki, aksariyat hollarda massiv indeksi sifatida butun son indekslari qo`llaniladi.

Haqiqiy turdagi (float, double) qiymatlar to`plami cheksiz bo`lganligi sababli ular indeksi sifatida ishlatilmaydi.

Massivning har bir qiymati massivning elementi deyiladi. Har bir element o`z nomeriga (inleksiga) ega.

C++ tilida indeks doimo 0 dan boshlanib, butun musbat sonlar bo`lishi mumkin, uning qiymati massiv uzunligidan kichik bo`lishi kerak.

Massiv e’loni quyidagicha bo`ladi:

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

Bu yerda 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 dastur 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:

m	=>	m[0]	m[1]	m[2]	m[3]	m[4]	m[5]

Massivning i elementiga m[i] yoki *(m+i) – ko`rinishda murojaat qilish mumkin. Massiv uzunligini sizeof(m) amali orqali aniqladi.

Massiv e’lonida uning elementlariga boshlang`ich qiymatlar berish (inisializasiyalash) mumkin. Massivni bir necha variantlar bilan inisializatsiyalash mumkin.

1) o`lchami ko`rsatilgan massiv elementlarini to`liq inisializatsiyalash:

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

Bunda 5 ta elementdan iborat bo`lgan t nomli bir o`lchamli massiv e’lon qilingan va uning barcha elementlariga boshlang`ich qiymatlar berilgan: t[0]=-10; t[1]=5; t[2]=15; t[3]=4; t[4]=3;

2) o`lchami ko`rsatilgan massiv elementlarini to`liqmas inisializasiyalash:

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

Bu yerdafaqat massivning boshidagi uchta elementiga boshlang`ich qiymatlar berilgan. Shuni aytib o`tish kerakki, massivni boshidagi yoki o`rtadagi elementlariga qiymatlar bermasdan, uning oxiridagi elementlarga boshlang`ich qiymat berish mumkin emas. Agarda massiv elementlariga boshlang`ich qiymat

berilmasa, unda kelishuv bo`yicha static va extern modifikatori bilan e’lon qilingan massiv elementlarining qiymati 0 soniga teng deb, automatic massivlar elementlarining boshlang`ich qiymati noma’lum hisoblanadi.

3) o`lchami ko`rsatilmagan massiv elementlarini to`liq inisializasiyalash:

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 massivni uzunligi berilmasa, boshlang`ich qiymati berilishi shart. Massivni e’lon qilishga misollar:

```
shar ch[4] = {_a‘, _b‘,‘c‘,‘d‘}; // belgilar massivi e’lon qilingan
int in[6] ={10,20,30,40}; // butun sonlar massivi
char str[]=||abcd||; //satr uzunligi 5 ga teng, chunki uning oxiriga ‘\0‘
//belgisi qo`shiladi
char str[]={_a‘,‘b‘,‘c‘,‘d‘}; // yuqoridagi satrning boshqacha yozilishi.
```

Masala. Bir oylik kundalik harorat berilgan, oy uchun o`rtacha haroratni hisoblash dastursi:

```
void main()
{
const int n=30;
int temp[n];
int i,s,temp_o`rtacha;

cout <<« kunlik haroratni kiriting:\n»
for (i=0;i<n;i++)
{
cout << «\n temp[«<<i<<»]=»;
cin >> temp[i];
}
for (i=0,s=0; i<n;i++)s+=temp[i];
temp_o`rtacha=s/n;
cout <<«kunlik harorat :\n»;
for (i=0;i<n;i++) cout <<«\t temp[«<<i<<»]=»<<temp[i];
cout << « o\‘rta harorat =»<< temp_urtacha;
return; }
```

3.5.2. Ko`p o`lchovli statik massivlar

C++ tilida massivlar elementining turiga cheklovlar qo`yilmaydi, uning turi chekli o`lchamdagi obyektlarning turi bo`lishi kerak, ya’ni massivni xotirada

qancha joy (bayt) egallashini hisoblash imkoniyati bo`lishi kerak. Xususan, massiv komponentasi massiv bo`lishi mumkin, ya'ni «vektorlar-vektori» natijada matrisa deb nomlanuvchi ikki o`lchamli massiv hosil bo`ladi.

Agar matrisaning elementi ham vektor bo`lsa, uch o`lchamli massivlar - kub hosil bo`ladi. Shu usulda ixtiyoriy o`lchamdagi massivlarni yaratish mumkin. Massiv ko`rinishi konkret yechilayotgan masalaga bog`liq bo`ladi.

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

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

Masalan, 10 satr va 20 ustundan iborat haqiqiy sonlar massivini aniqlaylik:

float a[10][20];

Massiv elementlariga murojaat qilish uchun nomdan keyin kvadrat qavsda har bir o`lcham uchun indeks yozilishi kerak, masalan b[i][j][k]. Bu elementga vositali murojaat ham qilish mumkin va uning variantlari:

((*(b+i)+j)+k) yoki *(*(b[i]+j)+k) yoki *(b[i][j] +k);

3.5.3. Dinamik massivlar bilan ishlash

Statistik massivlarning kamchiliklari shundaki, ularning uzunligi oldindan ma'lum bo`lishi kerak, undan tashqari uzunlik berilganlarga ajratilgan xotiraning o`lchami bilan chegaralangan. Ikkinchi tomondan, yetarlicha katta o`lchamdagi massiv e'lon qilib, konkret masala yechilishida ajratilgan xotira to`liq ishlatimasligi mumkin. Bu kamchiliklarni dinamik massivlar foydalanish orqali bartaraf etiladi.

Ikki o`lchovlik dinamik massivni tashkil qilish uchun uni quyidagicha e'lon qilish kerak:

int **a;

Bu e'londa «ko`rsatkichga ko`rsatkich» berilgan.

Endi satrlar soniga qarab ko`rsatkichlar massiviga dinamik xotira ajratish kerak:

a=new int *[m] // bu yerda n satrlar soni

So`ng har bir satrga takrorlash operatori yordamida xotira ajratish kerak va uning boshlang`ich adresini a massiv elementlariga joylashtirish zarur:

for (int i=0; i<m;i++) a[i]=new int [n] ; // n ustunlar soni

Dinamik massivda har bir satr xotiraning turli joylarida joylashishi mumkin.

Dinamik massivlarni ishlatib bo`lgandan keyin albatta delete amali bilan uni yo`qotish (o`chirish) kerak. Yuqoridagi misolda ikki o`lchamli massiv uchun avval massivning har bir elementi, oxirida massivning o`zi yo`qotiladi:

```
for (i=0;i<m;i++) delete [] a[i];
delete []a;
```

Matrisa va vektorni ko`paytirish masalasi uchun dinamik massivlardan foydalanilgan dastur:

```
void main ()
{
int n,m;
int i,j; float s;
cout <<«\n n=»;
cin >> n; // matrisa satrlari soni
cout <<«\n m=»
cin >> m; // matrisa ustunlari soni
float *b=new float [n]; // b massivga dinamik xotira ajratish
float *c=new float [m]; // s massivga dinamik xotira ajratish
float ** a =new float *[m] ; // ko`rsatkichlar massiviga xotira ajratish
for (i=0;i<m;i++) // har bir satr uchun
a[i]=new float [n]; // dinamik xotira ajratish
// bu yerda a va b massivlar elementlarini o`qish amallari bo`lishi kerak
// asosiy hisoblash qismi
for (i=0; i<m;i++)
{
for (j=0,s=0; j<n; j++) s+=a[i,j]*b[j] ;
c[i]=s;
}
// chop etish qismi
for (i=0; i<m;i++) cout <<«\t c[«<<i<<»]=»<<c[i];
delete [] b;
delete [] c;

for (i=0;i<m;i++) delete [] a[i];
delete []a;
return;
```

```
}
```

3.5.4. Funksiya va massivlar

Massivlar funksiyani parametri sifatida ishlatilishi ham mumkin, va funksiyanatijasi sifatida massivni qaytarishi mumkin.

Agar massiv parametr orqali funksiyaga uzatilsa , elementlar sonini aniqlash muammosi tug`iladi, chunki C++ tilida massivni uzunlgi ismi bilan aniqlanmaydi. Satr bilan ishlaganda masala aniq chunki harqanday satr «\0» belgisi bilan tugaydi. Shuni hisobga olib satrni uzunligini aniqlash mumkin.Masalan:

```
#include <iostream.h>
int len(char s[]) //massivni parametr sifatida ishlatish bir usuli
{
int m=0;
while (s[m++]);
return m-1;
}
void main ()
{
char z[]= « satr uzunligini aniqlash»;
cout <<z<<len(z);
}
```

Parametr satr bo`lmagan holda yoki fiksirlangan uzunligni ishlatish kerak, yoki massivni o`lchovini parametr sifatida uzatish kerak, yoki global o`zgaruvchi ishlatib unga massivni uzunligini berish kerak.

Misol:

```
#include <iostream.h>
float sum(int n, float *x) // bu ikkinchi usul
{
float s=0; for (int i=0;i<n;i++) s+=x[i];
return s;
}
void main()
{
float e[] ={ 1.2,2.0,3.0,4.5,-4.0};
cout << sum(5,e);
}
```

Massivni ismi ko`rsatkich bo`lgan sababli massiv elementlarini funksiyada o`zgartirish mumkin va bu o`zgartirishlar funksiyadan chiqqanda saqlanib qoladi.

```
#include <iostream.h>
void vect(int n, int *x, int *y) // bu ikkinchi usul
{
for (int i=0;i<n;i++)
y[i]=x[i]>0?1:0;
}
void main()
int a[]=(1,2,-4,3,-5,0,4};
int c[7];
vect(7,a,c);
for(int i=0;i<7;i++) cout <<‘\t‘<<c[i];
}
```

Quyidagi misol amalda juda ko`p uchraydi. Bu misolda ikki tartiblangan bir o`lchovli massivlar ulanadi natijada tartib bilan joylashgan yangi massiv hosil bo`ladi.

```
#include <iostream.h>
int *massiv_ulash(int ,int * ,int , int *); \\ funksiya massiv qaytaradi
void main()
{
int c[]={-1,2,5,10},d[]={1,7,8};
int *h;
h=massiv_ulash(5,c,3,d);
for(int i=0;i<8;i++ ) cout<<‘\t‘<<h[i];
delete [] h;
}
int *massiv_ulash(int n ,int *a ,int m, int *b); \\ funksiya aniqlash
{
int *x=new int [n+m];
int ia=0,ib=0,ox=0;
while (ia<n && ib<m)
a[ia]>b[ib]?x[ix++]=b[ib++]: x[ix++]=a[ia++];
while (ib<m) x[ix++]=b[ib++]:
while (ia<n) x[ix++]=a[ia++]:
return x;
}
```

Ko`p o`lchovli massivlar bilan ishlash murakkabroq bo`ladi, chunki bunday massivlarni xotirada joylash tartibi murakkab. Masalan parametrda to`gridan tug`ri x[n][n] massivni quyidagicha yozib bo`lmaydi:

```
float sum( int n, float x[][]).
```

Ko`p o`lchovli massivlarni parametr sifatida ishlatish bir nechta variantlari bor, ularni ko`rib chiqamiz:

1 variant:

Ikkichi o`lchovini aniq o`zgarmas ifoda bilan ko`rsatish (son)

```
float sum( int n, float x[][10]).
{
float s=0.0;
for (int i=0;i<n;i++)
for (int j=0;j<n;j++)
s+=x[i][j];
return s;
}
```

2 variant:

Ko`rsatkichlar massivini ishlatib chaqirishda har bir elementiga mos satr ostini manzilini berish yo`li bilan

```
float sum( int n, float *p[]).
{
float s=0.0;
for (int i=0;i<n;i++)
for (int j=0;j<n;j++)
s+=p[i][j]; \\ oldiga * ko`yilmaydi chunki murojat massiv sifatida
return s;
}
void main()
{
float x[ ][4]={{11,-12,13,14},{21,22,23,24},{31,32,33,34},{41,42,43,44}}
float * ptr[];
for (int i=0;i<4;i++)ptr[i]=(float *)&x[i];
cout << sum(4,ptr)<<endl;
}
```

3 variant:

Ko`p o`lchovli dinamik massivlarni ishlatish

```
float sum( int n, float **x).
{
float s=0.0;
for (int i=0;i<n;i++)
for (int j=0;j<n;j++)
```

```
s+=x[i][j]; \\ oldiga * qo`yilmaydi chunki murojaat massiv sifatida
return s;
}
void main()
{
float **ptr;
int n;
cin >>n;
ptr=new float *[n];

for(int i=0;i<n;i++)
{
ptr[i]=new float [n];
for (int j=0;j<n;j++) ptr[i][j]=(float)((i+1)*10+j);
}
cout <<sum(n,ptr);
for (int i=0; i<n;i++)delete ptr[i];
delete [] ptr;
}
```

Oxirida funksiya ikki o`lchovli massivni natijasifatida qaytarishiga misol keltiramiz. Bu misolda elementlarni qiymatlari tasodifiy sonlarni tashkil qiladi, tasodifiy sonlar math.h kutubxonasidagi random funksiya orqali hisoblanadi:

```
#include <vcl.h>
#pragma hdrstop
#include <iostream.h>
#include <conio.h>
#include <math.h>
#pragma argsused
int **rmatr(int n,int m)
{
int ** ptr;
ptr=new int *[n];
for (int i=0;i<n;i++)
{ ptr[i]=new int[m];
for (int j=0;j<m;j++) ptr[i][j]=random(100);
}
return ptr;
}
int sum( int n, int m, int **ix).
{
float s=0;
for (int i=0;i<n;i++)
```

```
for (int j=0;j<m;j++)
s+=ix[i][j];
return s;
}
void main()

{int n,m;
cin >> n>> m;
int ** matr;
randomize();
matr=rmatr(n,m);
for (int i=0;i<n;i++)
{
cout <<endl<<i<<" chi satr:"
for (int j=0;j<m;j++) cout '\t'<<matr[i][j];
}
cout <<endl<<"summa="<< sum(n,m,matr);
getch();
for (int i=0;i<n;i++)delete matr[i];
delete [] matr;
}
```

2-misol. Bir o‘lchovli haqiqiy ***B(N), n=20*** massivning eng katta elementini va uning tartib raqamini toping.

Belgilashlar: ***bmax*** va ***imax –B(N)*** massivning eng katta elementi va uning tartib raqami; ***k*** – massiv elementlari soni bo‘lsin.

Dastlab ***b*** massivning barcha elementlari qiymatini kiritamiz. Massiv birinchi elementini eng katta element deb faraz qilamiz: ***bmax=b1, imax=1***. ***i*** sikl parametri ***2*** dan ***k*** gacha o‘zgaradi. Har gal ***bi >bmax*** shart tekshiriladi, agar bu shart bajarilsa, ***bmax=bi*** va ***imax=i*** almashtirishlar bajariladi.

```
#include <iostream>
using namespace std;
int main()
{
// const int n=5;
int i,k,imax;
float bmax;
//float b[n];
cout<<"k ni kiriting"<<endl; cin>>k;
float b[k];
cout<<"b massivni kiriting \n";
for (i=1;i<=k;i++)cin>>b[i];
{bmax=b[1]; imax=1;}
for (i=1;i<=k;i++)    {
if (b[i]>bmax)
```

```
 {bmax=b[i]; imax=i;}
}
cout<<"bmax="<<bmax<<"\t imax="<<imax;
    }
```

3-misol. Haqiqiy *a(n), n≤15* massivning barcha musbat elementlarining o‘rta arifmetigini hisoblang.

Belgilashlarni kiritamiz. ***i*** **-** massiv elementlari indeksklarining qiymatlarini aniqlovchi o‘zgaruvchi, ***i=1,2,..k (k*** - massiv elementlari soni, ***k≤15), S*** va ***kn*** mos ravishda musbat elementlarning o‘rta arifmatik qiymati va ularning soni. ***a[i]>0*** shart bajarilsa ***S=S+a[i], kn=kn+1*** yig‘indi hisoblanishi kerak. Ushbu yig‘indini hisoblash tugagandan keyin ***S=S/kn*** hisoblanadi, ya’ni musbat elementlar yig‘indisi musbat elementlar soniga bo‘linadi. Agar ***kn=0*** bo‘lsa, massivning musbat elementlari mavjud emas.

```
#include <iostream>
using namespace std;
int main()
{
int i,k,kn;
float s;
cout<<"k ni kiriting \n"; cin>>k;
float a[k]; s=0; kn=0;
cout<<"a massivni kiriting \n";
for (i=1;i<=k; i++) cin>>a[i];
for (i=1;i<=k; i++)
if (a[i]>0)
{s=s+a[i]; kn=kn+1; }
if (kn==0){
cout<<"Musbat elementlari yo`q \n";
goto t;}
else
{s=s/kn;
cout<<"s="<<s
}
t:;}
```

Massivlarga misollar

Xotirada ketma-ket (regulyar) joylashgan bir xil turdagi qiymatlarga massiv deb ataladi. Odatda massivlarga zarurat, katta hajmdagi, lekin cheklangan miqdordagi va tartiblangan turdagi qiymatlarni qayta ishlash bilan bog`liq masalalarni yechishda yuzaga keladi. Massivlarni matematikadagi sonlar vektoriga o`xshatish mumkin, chunki vektor ham o`zining individual nomga ega va tuzilma

fiksirlangan miqdordagi bir turdagi qiymatlardan - sonlardan iboratdir. Demak, massiv - bu fiksirlangan miqdordagi ayrim qiymatlarning (massiv komponentalari) tartiblangan majmuasidir.

Bir o'lchamli massivga murojaat sintaksisi: *<massiv nomi >[<indeks>]*. Masalan: a[5]

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

<tur><nom> [<uzunlik >] [<uzunlik>], yoki b[n][m].

Masalan, 10 satr va 20 ustundan iborat haqiqiy sonlar massivini aniqlaylik:

float b[10][20];

1. Masalaning qo'yilishi

Elementlari soni N(1<=N<=10000) ta bo'lgan butun sonli massiv berilgan. Massivning eng kichik elementini indeksini hamda uning birinchi va ikkinchi manfiy elementi orasida joylashgan massiv elementlarini yig'indisini toping. Agarda massivni ikkita manfiy element bo'lmasa 0 chiqaring. Massiv elementlarini qiymatlari $[-10^9, 10^9]$ oralig'ida joylashgan.

2. Masalani echish usulini tanlash

Masala bir o'lchovli butun sonli massivning eng kichik elementi indeksini aniqlash, uning dastlabki ikkita manfiy elementlari orasidagi elementlar yig'indisini topishdan iborat. Agar manfiy elementlar ikkitadan kam bo'lsa, yig'indining qiymatini 0 deb olish kerak. Masalani echish uchun ketma-ketlikning eng kichik elementini va uning tartib nomerini aniqlash algoritmini qo'llaymiz. Buning uchun ketma-ketlikning birinchi elementini minimal deb olamiz, qolgan elementlarni mimimal element bilan taqqoslaymiz. Agar boshqa element minimal elementdan kichik bo'lsa, shu elementni minimal deb olamiz va bu jarayon ketma-ketlikning oxirgi elementigacha davom ettiriladi. Massivning ikkita manfiy elementlari orasidagi elementlar yig'indisini hisoblash uchun shartli o'tish operatoridan foydalanib, manfiy elementlar va ularning sonini aniqlaymiz.

Dastur matni:

[*] L2120.cpp

```cpp
#include <iostream>
using namespace std;
int main()
{
    int a[10001],i,j,n;
    cin>>n;
    long long sum=0;
    int ab=-1,ao=-1,soni=0;
    cin>>a[0];
    int min=a[0], ind=0;
    if(a[0]<0) { ab=0; soni=1;}
    for(i=1; i<n; i++)
    {
        cin>>a[i];
        if(min>=a[i])
        {ind=i;}
        if(a[i]<0 && soni==0)
         { ab=i; soni=1;}
            else  if(a[i]<0 && soni==1)
                {ao=i; soni=2;}
     }
     if (soni<2) {sum=0 ;}
else {
   for(i=ab+1; i<ao; i++)
    sum+=a[i];
     }
    out<<ind<<endl<<sum;
    return 0;
}
```

1. **Echim va uning tahlili**

L2120.cpp

```
#include <iostream>
using namespace std;
int main()
{
    int a[10000],i,j,n;
    cin>>n;
    long long sum=0;
    int ab=-1,ao=-1,soni=0;
    cin>>a[0];
    int min=a[0], ind=0;
    if(a[0]<0) { ab=0; soni=1;}
    for(i=1; i<n; i++)
    {
        cin>>a[i];
        if(a[i]<=min)
        {min=a[i]; ind=i;}
        if(a[i]<0 && soni==0)
         { ab=i; soni=1;}
            else  if(a[i]<0 && soni==1)
                {ao=i; soni=2;}
     }
     if (soni<2) {sum=0 ;}
else {
   for(i=ab+1; i<ao; i++)
    sum+=a[i];
     }
    cout<<ind<<endl<<sum;
    return 0;
}
```

```
D:\Sharofiddin\ерырър-2016\C++_misollar\L2120\L2120.exe
5
-1 0 5 -8 0
3
5
--------------------------------
Process exited with return value 0
Press any key to continue . . .
```

Elementlar soni N=5 бўлганда - 1 0 5 -8 0 sonlar uchun eng kichik element indeksi 3 ga teng, dastlabki ikkita manfiy sonlar orasidagi sonlar yig'indisi 5 ga teng bo'ladi.

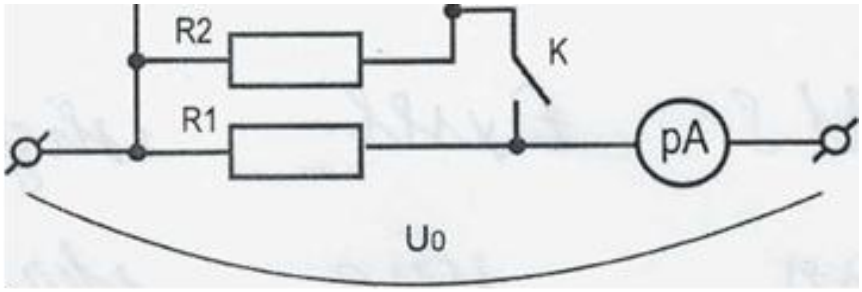
```
D:\Sharofiddin\ерырър-2016\C++_misollar\L2120\L2120.exe
5
2 4 -5 6 7
2
0
--------------------------------
Process exited with return value 0
Press any key to continue . . .
```

Elementlar soni N=5 бўлганда 2 4 -5 6 7 sonlar uchun eng kichik element indeksi 2 ga teng, manfiy sonlar bitta bo'lganligi uchun dastlabki ikkita manfiy son orasidagi sonlar yig'indisi masala shartiga ko'ra 0 ga teng bo'ladi.

3.2.5. Elektr zanjirlaridagi tokni hisoblash dasturi

Hozirgi zamon elektroenergetika tizimlarida elektr zanjirlarini zamonaviy elementlar asosida loyihalash va tahlil qilish, murakkab elektr zanjirlarining matematik modellarini yaratishda dastur vositalarining ahamiyati juda katta bo'lmoqda. Shuning uchun ushbu masala elektr zanjirlarini hisoblash va ularni hisoblash jarayonlarini dasturlashga bag'ishlanadi. Bunda asosiy e'tibor turli elektr zanjirlarini hisoblash va tahlil qilish usullarining asosini tashkil etuvchi ektrotexnik va matematik prinsiplarga qaratiladi. Shuning uchun elektr zanjirlarini hisoblash va tahlil qilishda nisbatan oddiy chiziqli elektr zanjirlarini hisoblashni dasturlarini ishlab chiqishdan bir muncha murakkab bo'lgan elektr zanjirlarini hisoblash dasturlariga ketma-ket ravishda o'tish tartibini saqlab qolishga harakat qildik.

1-misol. Quyida 3.2.1 - rasmda oddiy o'zgarmas tok zanjiri keltirilgan bo'lib, uni biror bir o'zgarmas U_0 kuchlanish manbaiga ulab, K kalit yopilgunga qadar ampermetr ko'rsatishini va kalit K yopilganda bu ampermetr qanday qiymat ko'rsatishini hisoblash talab etilsin. Bunday masalalarni yechishda uni hisoblashning dastur vositasini ishlab chiqishning afzalligi shundan iboratki dastur vositasi yordamida manba kuchlanishi U_0 ning yoki zanjir qarshiliklari R1, R2 va R3 larning har qanday qiymatlarida tok qiymatini qisqa vaqt oralig'ida tezda hisoblashinggiz mumkin. Masalan, U_0 = 10 Volt, R1 = 1 Om, R2 = 2 Om va R3 = 2Om qiymatlarga ega bo'lganda zanjirdagi kalit K uzilgan holda yoki K ulangan holda tok qiymati qanday bo'lishini hisoblash dastursini tuzamiz. Quyida elektr zanjirini hisoblashning C++ tilida yozilgan dastursi keltirilgan[ ].



3.5.1-rasm

Zanjirdagi tokni hisoblash dastursi

```
1  //Program developed by Pustam Baratov
2  //Elektr zanjirini hisoblash
3
```

```
4  #include<iostream>
5  Using namespace std;
6  Int main()
7  {
8  float r1, r2, r3, u0, R, I1, I2;
9  int k;
10 cout<<″Manba kuchlanisi U0 qiymatini kiriting!"end1;
11 cout<<"U0;
12 cin>>O0;
13 cout<<"Birinchi r1 qarshilik qiymatini kiriting:"<<end1;
14 cout<<"r1="; cin >>r1;
15 cout <<"ikkinchi r2 qarshilik qiymatini kiriting:"<<end1;
16 cout<<"r2=";cin>>r2;
17 cout<<"Uchunchi r3 qarshilik qiymatini kiriting:"<<end1;
18 cout <<"r3=";cin>>r3;
19 cout <<"Kalit k ga o yoki 1 qiymat kiriting:"<<end1;
20 cout <<"K=";cin>>K;
21 R=(r1+r2*r3)/(r1*r2+r1*r3+r2*r3);
22 I1=U0/r1;
23 I2=U0/R;
24 If(K==0)
25 {
26 cout<<",I1="<<I1<<end1;
27 }
28 If(K==1)
29 {
30 cout<<",I2="<<I2<<end1;
31 }
32 system("pause");
33 return 0;
34 }
```

Ushbu elektr zanjirida kalit K ochiq yoki uzilgan holni k=0 deb hisoblaymiz va u holda zanjirda qarshilikning qiymati faqat R1=1 Om bo’ladi. Shuning uchun Om qonuniga binoan zanjirdagi tok qiymati I=10 Amperga teng. Quyida zanjirda kalit K uzilgan holda tok qiymatini hisoblash dasturi natijasi keltirilgan.

Manba kuchlanishi U0 qiymatini kiriting?
U0=10
Birinchi r1 qarshilik qiymatini kiriting:
r1=1
Ikkinchi r2=2

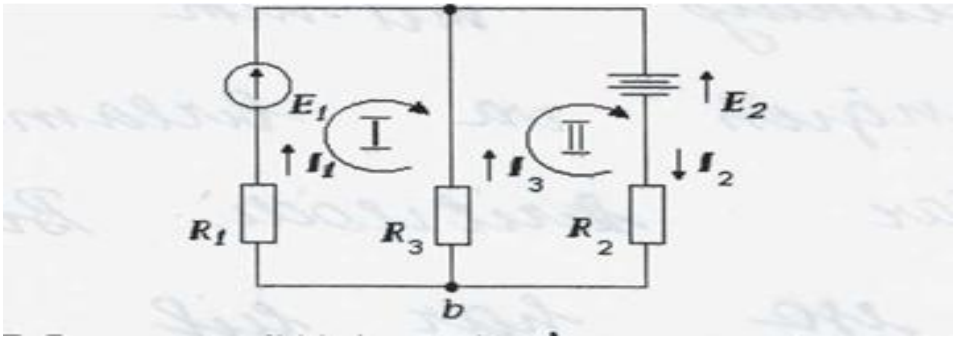
Uchinchi r3 qarshilik qiymatini kiriting:
r3=2
Kalit K ga 0 yoki 1 qiymat kiriting?
K=0
I1=10
Для продолжения нажмите любую клавишу . . . .

Kalit K ulangan holatni esa k=1 deb hisoblaymiz va u holda R1 = 1 Om qarshilikka yana ikkita R2 = 2 Om hamda R3= 2 Om qarshiliklar parallel ulanadi va zanjirning umumiy qarshiligi qiymati o'zgaradi, shuning uchun zanjirning toki l = 20 A ga teng bo'ladi. Quyida zanjirda kalit K ulangan holda tok qiymatini hisoblash dasturi natijasi keltirilgan.

Manba kuchlanishi U0 qiymatini kiriting?
U0=10
Birinchi r1 qarshilik qiymatini kiriting:
r1=1
Ikkinchi r2=2
Uchinchi r3 qarshilik qiymatini kiriting:
r3=2
Kalit K ga 0 yoki 1 qiymat kiriting?
K=0
I2=20
Для продолжения нажмите любую клавишу . . .

2-misol. Superpozitsiya usulida elektr zanjirini hisoblash dastuni tuzish

Masalaning qo'yilishi: Elektr zanjirining parametrlari E1=12V, E2=24V, R1=R2=0,2 O
topishning shunday dasturi tuzish kerakki, dastur tarmoqlardagi toklarni yuqorida keltirilgan zanjir parametrlari har qanday qiymatga ega bo'lgan holda hisoblay oladigan.



3.5.2-rasm. Ikki konturli o’zgarmas tok zanjiri

Bunday elektr zanjiriarini hisoblashda injenerlar “ Eiektrotexnikaning nazariy asoslari fani bo’yicha, jumladan “Elektr zanjiriarini hisoblash usullari”*n*i bilishi zarur. Ammo biz ushbu misolni yechishda e’tiborni yuqorida keltirilgan elektr zanjirini hisoblash usulini emas balki uni hisoblashning tayyor tenglamalari va hisoblash ketma - keltligi berilgan holda hisoblashning dastur vositasini ishlab chiqishga qaratamiz. Shuning uchun yuqorida keltirilgan zanjirni “Ustma - ustlash” (superpositsiya) usulining tayyor tenglamalari hamda uning ketma - ketligini keltiramiz. Superpozitsiya usuliga binoan zanjirdagi har bir elektr yurituvchi kuch (EYUK) dan hosil bo’lgan toklar alohida hisoblanadi va keyin esa natijaviy toklar hisoblanadi. Hisoblash quyidagi ketma - ketlikda amalga oshiriladi: E2=0, ya’ni elektr zanjiridagi ikkinchi EYUK olib tashlanadi va E1 EYUK dan hosil bo’ladigan toklar hisoblanadi. Hisoblash formulalarini quyida keltiramiz:

$$\boldsymbol{I_{11} = \frac{E_1}{R_{11}} = \frac{12}{0{,}2+0{,}198} = \frac{12}{0{,}398} = 30{,}15A}$$

Bunda, $\boldsymbol{R_{11} = R_1 + \frac{R_2 * R_3}{R_2+R_3} = 0{,}2 + \frac{0{,}2+20}{0{,}2+20} = 0{,}198\ Om}$

$$\boldsymbol{U_{1ab} = I_{11} \cdot \frac{R_2 \cdot R_3}{R_2+R_3} = 30{,}13 \cdot 0{,}198 = 5{,}9697\ V}$$

$$\boldsymbol{I_{12} = \frac{U_{Iab}}{R_2} = 29{,}85\ A\ ;} \quad \boldsymbol{I_{13} = \frac{U_{Iab}}{R_3} = 0{,}290\ A\ ;}$$

E1=0, E_2 EYUK manbaidan hosil bo’ladigan toklar hisoblanadi. Hisoblash formulalari quyidagicha:

$$\boldsymbol{I_{22} = \frac{E_2}{R_{22}} = \frac{24}{0{,}398} = 60{,}3A}$$

Bunda, $\boldsymbol{R_{22} = R_2 + \frac{R_1 * R_3}{R_1+R_3} = 0{,}2 + \frac{0{,}2+20}{0{,}2+20} = 0{,}398\ Om}$

$$U_{2ab} = I_{22} \cdot \frac{R_1 \cdot R_3}{R_3 + R_3} = 60,3 \cdot 0,198 = 11,94\,V$$

$$I_{12} = \frac{U_{2ab}}{R_1} = 59,7\,A\,; \quad I_{23} = \frac{U_{2ab}}{R_3} = 0,597\,A\,;$$

Har bir tarmoqdagi tokni esa shu tarmoqdagi toklarni algebraik qo'shish orqali topamiz. Hisoblash formulalari quyidagicha:

$$I_1 = I_{11} - I_{21} = 30,15 - 59,7 = -29,55\,A,$$

$$I_2 = I_{12} - I_{22} = 29,85 - 60,3 = -3045\,A,$$

$$I_3 = -(I_{13} + I_{23}) = -(0,296 + 0,597) = -0,893\;\;A,$$

Endi elektr zanjirining parametrlari E1=12V, E2=24V, R1=R2=0,2 Om, R3=20 Om kabi bo'lsa, ustma-ustlash(superpozitsiya) usuli yordamida zanjir parametrlari har qanday qiymatga ega bo'lgan holda zanjirning barcha tarmoqdagi toklar I1,I2,va I3 larni topishning dasturini dasturini keltiramiz[ ]:

```
1  //Program ATJMM
2  //Ustma-ustlash usulida elektr zanjirini hisoblash
3
4  #include<iostream>
5  Using namespace std;
6  Int main()
7  {
8  float E1,E2,
   R1, R2, R3, R11, R22, U1ab, I11, I12, I13, U2ab, I22, I21, I23, I1,
   I2, I3;
9  cout<< "EYUKlar qiymatlarini kiriting!" <<end1;
10 cout<<"E1= ";
11 cin>>E1;
12 cout<<"E2=";
13 cin>>E2;
14 cout<< "Qarshiliklar qiymatlarini kiriting!" <<end1;
15 cout<< "R1= ";
16 cin >>R1;
17 cout<<"R2= ";
18 cin >>R2;
19 cout<<"R3= ";
20 cin >>R3;
21
22 R22=R2+(R1*R3)/(R1+R3);
```

```
23 I22=E2/R22;
24 U2ab=I22*(R1*R3)/(R1+R3);
25 I21=U2ab/R1;
26 I23=U2ab/R3;
27
28 R11=R1+(R2*R3)/(R2+R3);
29 I11=E1/R11;
30 U1ab=I11*(R2+R3)/(R2+R3);
31 I12=U1ab/R2;
32 I13=U1ab/R3;
33
34 I1=I11-I21;
35 I2=I12-I22;
36 I3=(I13+I23);
37
38 If(E1==0)
 40   {
 41   cout<< "I22= <<I22 <<end1;
 42   cout<< "I21= <<I21 <<end1;
 43   cout<< "I23= <<I23 <<end1;
 44   }
 45   Else if(E2==0)
 46   {
 47   cout<< "I11= <<I11 <<end1;
 48   cout<< "I12= <<I12 <<end1;
 49   cout<< "I13= <<I13 <<end1;
 50   }
 51   Else
 52   {
 53   Cout<<"Natijaviy toklar quyidagicha:"<<end1;
 54   Cout<<"I1="<<I1<<end1;
 55   Cout<<"I2="<<I2<<end2;
 56   Cout<<"I3="<<I3<<end3;
 57   }
 58   system("pause");
 60   return 0;
 61   }
```

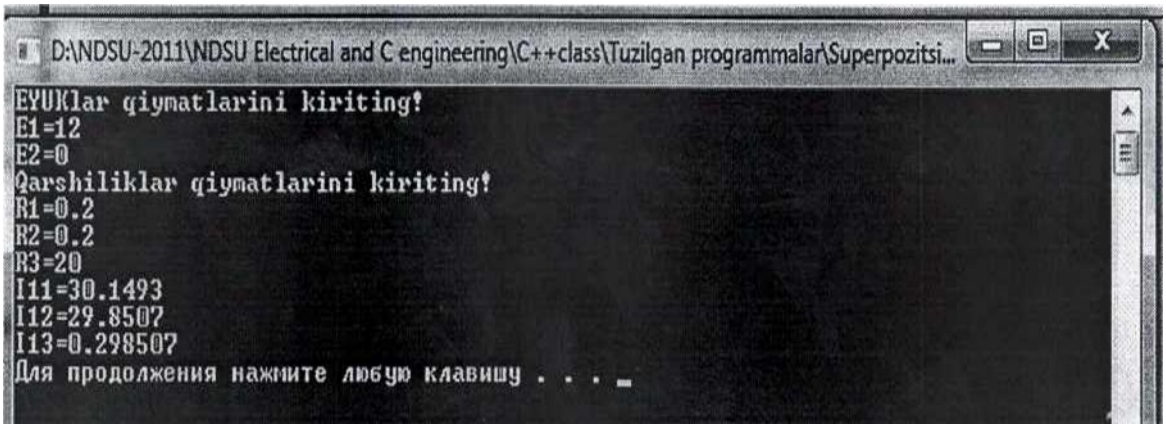
Bu dasturning Ms Vizual Studio 2010 integrallashgan muhitida quyidagi ko’rinishda bo’ladi.

Quyida elektr yurituvchi kuchlar har xil qiymatga ega, ya’ni E1=0, E2 = 24 V; E1 = 12 V, E2=0 hamda E1 = 12 V, E2 = 24 V bo’lganda dastur natijalari keltirilgan.

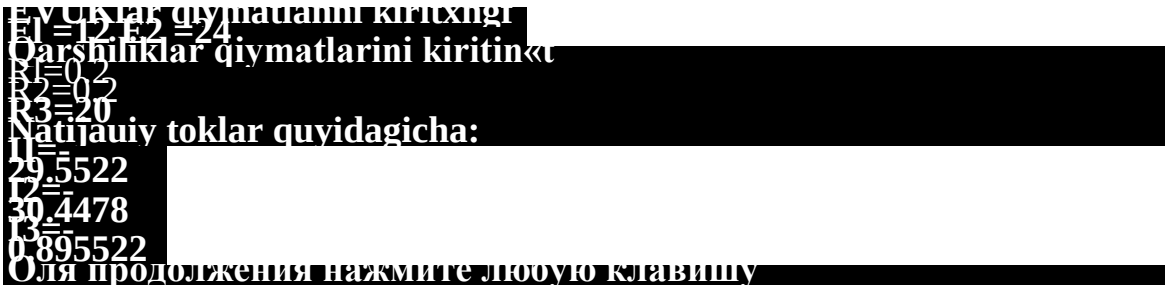
E1=0 bo’lganda zanjirdagi toklar qiymati quyidagicha natijaga ega bo’ladi.

EYUK qiymatlarini kiriting?
E1=0
E2=24
Qarshiliklar quymatlarini kiriting?
R1=0.2
R2=0.2
R3=20
I22=60.2985
I21=59.7015
I23=0.597015
Для продолжения нажмите любую клавишу . . .

E2=0 bo’lganda zanjirdagi toklar qiymati quyidagicha natijaga ega bo’ladi:

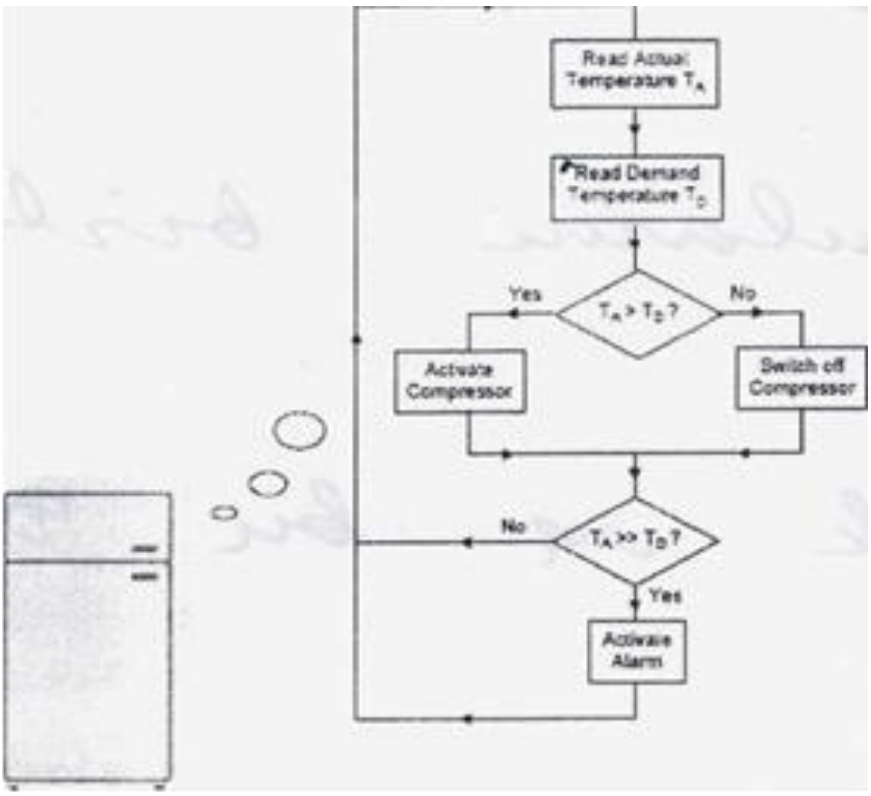


E1=12 V va E2=24 V bo’lganda zanjirdagi toklar qiymatini dasturiga natija quyida keltirilgan.



3.5.3. Texnologik jarayonlarqj avtomatik boshqarishda algoritmlar va dasturiash

Quyidagi rasmda muzlatgichning ish jarayonini algoritmlashning blok-sxemasi keltirilgan. Biz boshqarishning blok-sxemasini ishlab chiqamiz. Bunda foydalanuvchi bitta nazorat usuliga ega bo’ladi. Faqat rostlanadaigan resistor orqali siz xohlagan haroratga o’rnatishingiz mumkin. Muzlatgich ichida harorat datchigi joylashtirilgan. Haroratni rostlash sistemadagi kompressorni o’chirish yoki ishga tushirish orqali amalga oshiriladi. Kompressor ishlab turgan holda harorat pasayadi, u oochirilgan sholda esa harorat ko’tarila boshlaydi. Ishlab chiqilgan dastur joriy haroratni hamda talab qilinayotgan harorat qiymatini ham o’qiy olishi zarur. Bu harorat qiymatlarini o’qiydi va ularning qaysi biri katta yoki kichik ekanligini taqqoslaydi. Agar temperature qiymati joriy harorat qiymatiga teng bo’lsa u holda compressor ishga tushiriladi. Agar ular orasidagi farq juda katta bo’lsa u holda sistemadagi ogohlantirish ishga tushadi. Rasmda bayon etilgan jarayon algoritmini ko’rib turibsiz va unda faqat ikkita holatda shartni tekshirish imkoniyati mavjud. Ushbu shartlar bajarilishiga qarab jarayonda sikl takroriy ravishda ishlaydi va bu sikl uzluksiz sikldir. Bu blok - sxemani juda detallashtirilgan tarzda yoki juda sodda ravishda ham tasvirlash mumkin. Keyingi qadam esa ushbu blok-sxemani dasturda yozish va uni mashina tiliga aylantirishdan iborat.



1 - masala. Ikkita qarshiligi mavjud hamda ular ketma **-** ket va parallel anish mumkin bo'lgan elektr zanjirining umumiy (ekvivalent) qarshiligini soblash dastursini ishiab chiqing. Qarshiliklar qiymatini ixtiyoriy nlashingiz mumkin.

Ko'rsatma: **/.** *Berilgan ma iumotiarni kiritng. 2. Birinchi R1 qarshilik qiymati.*

. Ikkinchi R2 qarshilik qiymati. 4. Qarshiiikiarning uianish usu/i: (i ~ etma- ket, 2 - parallel).

2-**masala:** Agar elektr zanjirining quvvati P va kuchlanish U berilgan bo'lsa I elektr zanjiridagi tok kuchini hisoblash dastursini tuzing (l=P/U, bu srda I -tok kuchi, A; P - quvvat, W; U - kuchlanish, V). Dastur iydalanuvchi tomonidan kiritilgan ma'lumotlarni tekshirishi zarur. Agar kiritilgan la'lumotlar noto'g'ri bo'lsa u holda (masalan, maxraj 0 ga teng bo'lsa) <randa xatolik haqida xabar ko'rsatilishi zarur.

3- **masala:** Avtomobilning yoqilig'l quyishdagi narxini hisoblash dasturini tuzing. Berilgan ma'lumotlar: Yoqilg'i turi (benzin 80, 91, 95 yoki dizel yoqilg'i) va qancha litr quygan.

Ko'rsatma: *1. Benzin markasi 1 - 80, 2 - 91, 3 - 95, 4 - dizel yoqilg'i. 2. Sizning tanlovingiz. 3. Qancha iitr. 4. Ekranda aks etishi zarur: a) 1 litr narxi:___ so'm, b)*

Qancha iitr: ____iitr, c) To'iov summasi: ____so'm.

3- masala: Shaharlararo telefon gaplashuvi narxini hisoblash dastursini ishlab chiqing. Telefon so'zlashuvining 1 minutdagi narxi abonament joylashgan shahardan telefon qilingan shahargacha bo'lgan masofaga bog'liq. Dastur uchun boshlang'ich ma'lumotlar: shahar kodi va gaplashuv vaqti. Quyida shaharlar kodi va narxlar keltirilgan.

Shaharlar	**Kod**	**1 minut uchun so'ziashuv narxi (so'm)**
Toshkent	998 71	50
Qarshi	99875	100
Termiz	998 76	150
Samarqand	998 66	75
Buxoro	998 65	175

Ko'rsatma: *1.Shahar kodi. 2. So'ziashuv vaqti. 3. Shahar. 4. 1 minut narxi. 5. So'z/ashuv narxi.*

Tayanch so'z va iboralar

Massiv, ko'p o'lchovli massiv, **superpositsiya,** xotira, dinamik massivlar, satr, funksiya, tartiblangan, parameter, float, takrorlash operatori, Statistik massivlar, Ikki o`lchamli massiv.

Mustahkamlash uchun nazorat savollari

1. Dasturlashda ko`rsatkichlardan qanday foydalaniladi?
2. Operator deganda nimani tushinasiz?
3. Blok deganda nimani tushinasiz?
4. Kiritish va chiqarish operatorlari qanday ko`rinishda bo`ladi?
5. Tanlash operatorining umumiy ko`rinishi qanday?
6. Takrorlash operatorlari va ularnig turlarini qanday bo`ladi?
7. Massiv deb nimaga aytiladi?
8. Massivning qanday turlarini bilasiz?
9. Massivning elementlari deb nimaga aytiladi?

10. Massiv elementning tartib nomeri nima deb ataladi?
11. Massiv elementining indeksi qanday ifodalanadi?
12. A[10], X[100], b[123] – bular qanday massiv?
13. Bir o’lchovli massivga ta’rif bering?
14. Ko’p o’lchovli massivga ta’rif bering?