ШАГ компьютерная академия

ПРОГРАММИРОВАНИЕ НА ЯЗЫКЕ С

Dərs №6

C

dilində
proqramlaşdırma

Mündəricat

1. Verilənlərin qruplaşdırılmasının zərruriliyi

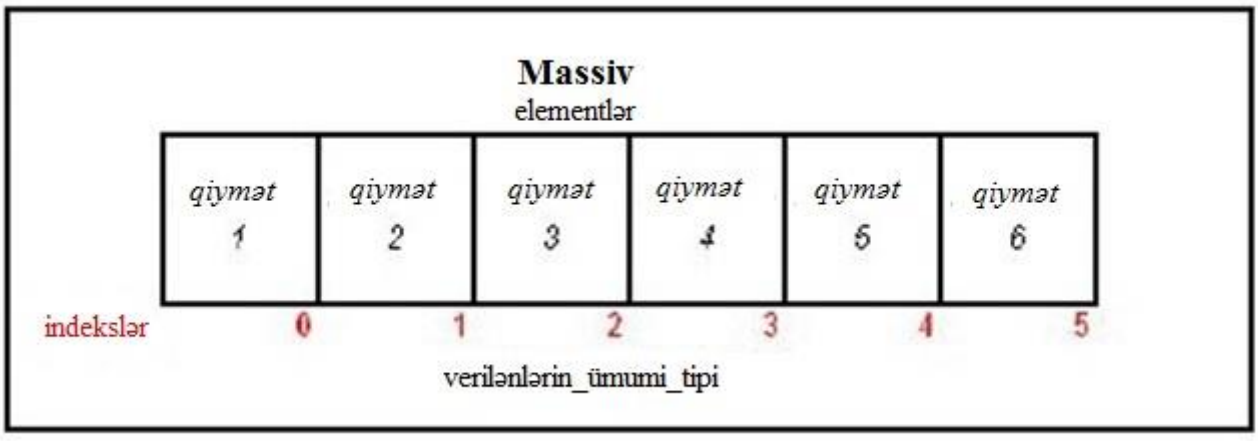
Bu gün verilənlərin saxlanılması mövzusunda danışacağıq. İlk məşğələlərimizin birində biz dəyişənin varlığı haqqında danışdıq və ona informasiyanı yerləşdirmək üçün əməli yaddaşın bir hissəsi kimi tərif verdik. Şübhəsiz ki, normal proqram dəyişənsiz mövcud ola bilməz. Lakin, bəzən sadə dəyişənlər verilənlər üzərində əməliyyat aparma problemlərini həll etmirlər. Məsələ burasındadır ki, əvvəlki dərslərdə baxdığımız dəyişənlərdən hər biri eyni zamanda informasiyanın yalnız bir elementini saxlamaq imkanına malikdir. İkinci elementi saxlamaq üçün daha bir dəyişən yaratmaq lazımdır. Bəs həmcins tipli verilənlərdən ibarət elementlər çoxluğunu saxlamaq üçün nə edək? Hər element üçün bir dəyişən yaratmaq məqsəduyğun deyil. Bəs əgər bir neçə yüz elementlə işləmək tələb olunarsa nə etmək olar? Məsələ sürətli həll olunmaz vəziyətə gəlib çıxar. Bir neçə yüz dəyişən yaratmaq isə ağlasığmazdır. Belə heç də sadə olmayan məsələni necə həll etmək olar? Belə vəziyyətdə həll yolunu massivlərdə axtarmaq lazımdır. Bəzi tərifləri və xüsussiyətləri nəzərdən keçirək.

Massiv anlayışı.

1. Massiv - bir neçə eyni tipli qiymətləri saxlamağa imkan verən dəyişənlər toplusudur.
2. Bu toplunun bütün qiymətləri bir ad altında birləşir.

3. Bu halda massivdə hər bir dəyişən element adlanan sərbəst qiymətdir.
4. Hər bir element öz sıra nömrəsinə - indeksə malikdir. İndeksə görə massivin hər hansı bir konkret elementinə müraciət etmək olar.
5. Massivdə elementlərin nömrələnməsi 0-dan başlanır.

Sxem:
Yuxarıda yazılanlardan istifadə edərək, massivin təsvir sxemini belə verə bilərik:

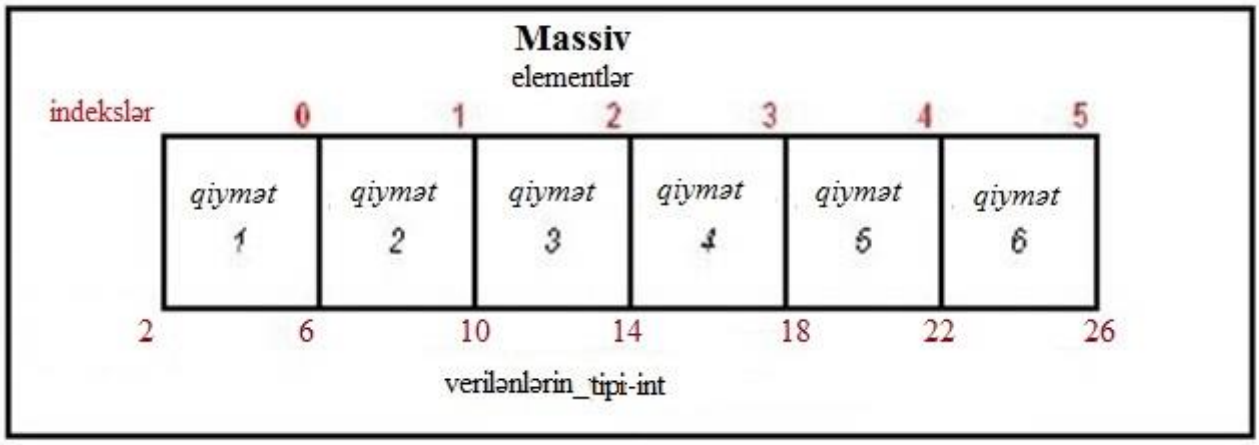


Massivin yaddaşda yeri:
Massiv yaddaşda sıra ilə yerləşir, yəni hər element digərinin ardinca gəlir. Əvvəlcə 0-cı element gəlir, sonra 1-ci və s. Elementlər öz ünvanlarının artma ardıcıllığına əsasən yerləşirlər: Hər bir element digərindən massivin baza tipinə bərabər olan bayt miqdarı qədər geri qalır. Massiv üzrə mövqeləşdirmə düsturu:

Baza ünvanı + baza tipinin ölçüsü * indeks;

Əgər ünvan düzgün göstərilməzsə, düsturla hesablanmış ünvana baza ünvanda mövqeləşməsi aparılır.

Bu halda proqram özunə aid olmayan yaddaş xanasının daxilinə tam çıxış əldə edir. Bunun nəticəsində icra mərhələsində xəta baş verə bilər.



Sonda qeyd etmək lazımdır ki, massivin hər bir elementinin birbaşa bütün massivin tipindən asılı olan ölçüsü var. Məsələn, əgər massiv int-verilənər tipinə malikdirsə, onun hər bir elementinin ölçüsü 4 baytdır. Beləliklə, massivin ümumi ölçüsü aşağıdakı düsturla hesablanır:

ÜMUMİ_ÖLÇÜ=VERİLƏNLƏR_TİPİNİN_ÖLÇÜSÜ*MASSİVDƏKİ_ELEMENTLƏRİN_SAYI

İndi biz nəzəri olaraq massiv haqqinda demək olar ki, hər şeyi bilirik. Yalnız praktik hissə ilə tanış olmaq və bu strukturun necə asan yaradılmasına və rahat istifadə edilməsinə əmin olmaq qalır. Bunun üçün biz dərsin növbəti bölməsinə keçirik.

3. Massivin yaradilması və onun verilənlər ilə doldurulması

Massivin verilmə sintaksisi.

Əvvəlcə massivi yaratmağı öyrənməliyik. Bunun üçün isə ilk olaraq massivin ümumi quruluşunu müəyyən etməliyik.Daha sonra isə aydinlaşdırmalıyıq ki, o hansı qaydalara və məhdudiyyətlərə tabe olur.

```
Verilənlər_tipi massivin_adı[elemetnlərin_sayı];
```

1. Verilənlərin_tipi - sizə məlum olan mövcud verilən tiplərindən istənilən biri. Massivin hər bir elementi məhz bu tipə malikdir.
2. Massivin_adı - «dəyişənlərin adları qaydaları»na tabe olan istənilən ad. (Bu qaydalara biz birinci dərsimizdə baxmışdıq).
3. Elementlərin_sayı - massivdəki elemetlərin sayı. Burada tam sabit kəmiyyət yerləşməlidir. Belə kəmiyyət ya tam qiymətli hərf, ya da tam sabit dəyişən ola bilər.

Qeyd: Diqqət yetirin ki, massivdəki elementlərin sayı proqramın yaradılması mərhələsində təyin edilməlidir. Başqa sözlə, massivin ölçüsünü hər hansı şərtdən və ya istifadəçinin qərarından asılı olaraq vermək mümkün deyil. Bu, kompilyasiya mərhələsində xətaya gətirib çıxarar.

Birinci variant.

Elementlərinin hər biri int verilənlər tipinə malik olan 5 elementdən ibarət ar massivi verilmişdir.

```
int ar[5];
```

İkinci variant.

Qiyməti 3-ə bərabər olan size sabiti verilmişdir. Daha sonra isə elementlərinin hər biri double verilənlər tipinə malik olan 3 elementdən ibarət br massivi verilir.

```
const int size=3;
double br[size];
```

Qeyd: Sizə ikinci yazılış formasını məsləhət gorürük belə ki, o daha rahat və yığcamdır.

Massivin elementlərinə müraciət.

Massivin hər hansı konkret bir elementinə müraciət edilməsinə baxaq.

```
Qitmətin yazılışı
Massivin_adı[elementin indeksi]=qiymət;
Qiymətin alınması
cout<<massivin_adı[elementin_adı];
```

Burada elementin indeksinin yerinə İSTƏNİLƏN tam qiymətli ədəd və ya nəticəsi tam ədəd olan ifadə yazmaq olar.

```
const int size=5;
int ar[size]; // massivin yaradılması
ar[2]=25; 2 indeksli elementə 25 qiymətinin yazılışı
// 2 indeksli elementin qiymətinin ekrana çıxarılması - 25
cout<<ar[2]<<"\n";
```

Qeyd: Bir daha qeyd etmək istərdik ki, massivdə elementlərin nömrələnməsi 0-dan başlanır! Beləliklə, 5 elementdən ibarət massivdə axırıncı elementin indeksi 4-dür. Massivin hüdudlarından kənara çıxmaq olmaz. Bu icra mərhələsində səhvlərə gətirib çıxara bilər.

Massivin inisiallaşdırılması variantları:

Massivi iki üsulla doldurmaq olar:

Birinci üsul – massivi yaradanda inisiallaşdırılmaq oalr.

Verilənlər_tipi massivin_adı[elementlərin_sayı]={qiymət1, qiymət2, … qiymət n};

```
const int size=3;
int ar[size]={1,30,2};
```

İnisiallaşdırılmanın bu formasının bir neçə xüsusiyyəti vardır:

1. İnisiallaşdırılma siyahısındakı bütün qiymətlərinin tipi massivin verilənlər tipi ilə eynidir. Buna görə də yaradılma zamanı elementlərin sayını göstərməmək də olar. Əməliyyat sistemi özü İnisiallaşdırılma siyahısındakı elementlərin sayına əsasən massivin ölçüsünü təyin edəcəkdir.

Verilənlər_tipi massivin_adi[]={qiymət1, qiymət2, qiymət3, … qiymət n};

```
int ar[]={1,30,2};                /*verilmiş sətirdə massiv avtomatik 3 ölçüsünü alacaq.*/
```

2. Əgər inisiallaşdırılma siyahındakı elemetlərin sayı, massivdəki elemetlərin sayından azdırsa, bu halda qalan qiymətlər avtomatik olaraq 0-larla doldurulur:

```
int ar[5]={1,2,3}
```

Bu yazı, aşağıdakına ekvivalentdir :

```
int ar[5]={1,2,3,0,0};
```

3. Əgər İnisiallaşdırılma siyahısındakı qiymətlərin sayı massivin elementlərinin sayından çox olarsa, kompilyasiyanın bu mərhələsində xəta baş verir:

```
int array[2]={1,2,3};              // kompilyasiya mərhələsində  xətə
```

4. Massivlərin inisiallaşdırılması zamanı artıq sizə məlum olan vahidləşmiş inisiallaşdırılmadan da istifadə etmək olar.

```
int arr[]{1, 2, 3};
int arr2[4]{11, 21, 31};
```

İkinci üsul - dövr vasitəsilə massivin İnisiallaşdırılması. Bu halda, massivi istifadəçinin köməyi ilə qiymətlər ilə doldurmaq olar. Layihənin adı InitArray-dır.

```
#include<iostream>
using namespace std;
void main()
{
    const int size=3;
    int ar[size]; //3 elementdən ibarət massivin yaradılması
    //massivin elementlərini seçən dövr
    for (int i=0;i<size;i++)
    {
        cout<<"Enter element\n";
    /* dövrdəki hər bir təkrar üçün istifadəçiyə massivi doldurmaq üçün i indeksli element verilir. Belə ki,
i hər dəfə yeni qiymət alır. */
        cin>>ar[i];
    }
}
```

Massivin qiymətlərinin ekrana verilməsi.

Siz yəqin təxmin edirsiz ki, massivlər üzərindəki bir çox əməliyyatları dövrlərin köməyi ilə (elementləri növbə ilə seçən) həyata keçirmək daha rahatdir. Bu həqiqətən belədir və ekrana verilmə istisna deyil. Gəlin, elə bir proqram nümunəsi göstərək ki, o massivi yaradır, doldurur və ekrana verir. Layihənin adı ShowArray-dir.

```
#include<iostream>
using namespace std;
void main()
{
    const int size=3;
    int ar[size]; //3 elementdən ibarət massivin yaradılması
    // massivin elementlərini seçən dövr
    for (int i=0;i<size;i++)
    {
        cout<<"Enter element\n";
    /* dövrdəki hər bir təkrar üçün istifadəçiyə massivi doldurmaq üçün i indeksli element verilir.
     Belə ki, i hər dəfə yeni qiymət alır. */
        cin>>ar[i];
    }
    cout<<"\n\n";
    // massivin elementlərini seçən dövr
    for (int i=0;i<size;i++)
    {
        //i indeksli elementlərin ekrana verilməsi
        cout<<ar[i]<<"\n";
    }
}
```

Siz massivlərlə tanış oldunuz. İndi isə dərsin digər bölmələrinə keçək və bir neçə praktik nümunələrə baxaq.

3. Massivin mənfi elementlərinin cəminin tapılması üçün proqram nümunəsi

Məsələnin qoyuluşu:

Massivdə olan bütün mənfi elementlərin cəmini tapan proqram yazın.

Layihənin adı AmountOfNegative.

Kodun reallaşdırılması:

```
#include <iostream>
using namespace std;
void main ()
{
        //Massivin ölçüsünün təyin edilməsi
        const int size=5;


            //verilənlər ilə massivin yaradılması və inisiallaşdırılması
            int ar[size]={23,-11,9,-18,-25};

            //cəmlənmə üçün dəyişən
            int sum=0;

            // massivin elementlərini sıra ilə seçən dövr
            for (int i=0;i<size;i++)
            {
                //əgər elementin qiyməti mənfidirsə (sıfırdan kiçik)
                if(ar[i]<0)
                   sum+=ar[i]; // onun qiymətinin ümumi cəmə əlavə edilməsi
            }


            //cəmin qiymətinin ekrana nümayişi
            cout<<"Sum = "<<sum<<"\n\n";
}
```

Kodun şərhi:

1. Dövr növbə ilə 0-dan size qədər olan elementləri seçir. Size yoxlanılan diapazona daxil deyil, belə ki, axırıncı size elementinin indeksi 1-dir.

2. Dövrdəki hər bir təkrar üçün daxildəki elementin qiymətinin mənfi olması halı yoxlanılır.

3. Əgər qiymət 0-dan kiçikdirsə, o, cəmə əlavə edilir.

Göründüyü kimi massiv üzərində iş hər hansı bir diapazonun analizinə bənzəyir. Lakin verilmiş halda diapazonun aşağı sərhədi 0-dır, yuxarı sərhədı isə massivdəki elementlərin sayı ilə müəyyən olunur.

Massivin minimal və maksimal elementlərinin tapılması üçün proqram nümunəsi.

Məsələnin qoyuluşu:

Massivin minimum və maksimum qiymətini tapan və ekrana verən proqram yazın. Lahiyənin adı MinMaxElement.

Reallaşdırma kodu:

```
#include <iostream>
using namespace std;
void main ()
{
          //massivdəki elementlərin sayının müəyyən edilməsi
         const int size=5;


              //massivin yaradılması və inisiallaşdırılması
              int ar[size]={23,11,9,18,25};

              int max=ar[0]; // 0 elementini maksimum kimi götürək
              int min=ar[0]; // 0 elementini minimum kimi götürək

              //massivdəki 1-dən başlayan elementləri seçən dövr
              for (int i=1;i<size;i++)
              {
                  //əgər cari element minimumdan kiçikdirsə
```

```
            if(min>ar[i])
              //minimum qiymətinin yenidən yazılması
              min=ar[i];


            // əgər cari element maksimumdan böyükdürsə
            if(max<ar[i])
              // maksimum qiymətinin yenidən yazılması
              max=ar[i];
        }

        // nəticənin ekrana çıxarılması
        cout<<"Max = "<<max<<"\n\n";
        cout<<"Min = "<<min<<"\n\n";
}
```

Kodun şərhi:

1. Əvvəlcə belə bir mülahizə irəli sürək. Tutaq ki, massivin 0 indeksli elementi minimaldır.
2. 0 indeksli elementin qiymətini min dəyişəninə yazırıq.
3. Bu faktı təsdiq və ya təkzib etmək üçün 1 indeksli elementdən başlayaraq massivin bütün elementlərini dövrdə seçək.
4. Dövrün hər bir təkrarında fərz edilən minimumu massivin cari elementi (i indeksli element) ilə müqayisə edək.
5. Əgər fərz edilən minimumdan kiçik qiymət rast gəlirsə onda min olaraq bu qiymət götürülür və tədqiqat davam etdirilir.

Yuxarıda şərh olunanlar maksimum üçün də doğrudur, yalnız bu halda böyük qiyməti axtarmaq lazımdır.

Siz artıq massivlərlə tanışsınız və bir neçə nümunəyə də baxmısınız. İndi özünüzü sınamaq vaxtıdır. Ev tapşırıqlarının yerinə yetirilməsində və testlərin həllində uğurlar arzulayırıq.

4. Ev tapşırığı

Aşağıda təsvir olunan tapşırıqlarda giriş veriləni olaraq elementləri istifadəçi tərəfindən klaviaturadan daxil edilən 10 elementdən ibarət massiv götürülür.

1. Massivin elementlərini əks istiqamətdə çıxaran proqram yazın.

Nümunnə: massiv 23 11 6 çevrilir 6 23 11-ə.

2. Massivin tək və cüt elementlərinin cəmini hesablayan proqram yazın.
3. Elə bir proqram yazın ki, massivdə iki və ya daha artıq dəfə təkrarlanan qiymətləri tapsın və ekrana çıxarsın.
4. Elə bir proqram yazın ki, massivdə olan ən kiçik tək ədədi tapsın və ekrana versin.