



ПРОГРАММИРОВАНИЕ НА ЯЗЫКЕ C

Dərs №7

Çoxölçülü massivlər

Mündəricat

1. Təsadüfə ədəd generatoru	3
2. Təsadüfə ədəd generatorundan istifadə	10
3. Çoxölçülü massivlər , ikiölçülü massivlər özəl hal kimi.....	12
4. Tətbiqi nümunə	16
5. Ev tapşırığı.....	18

1. Təsadüfi ədəd generatoru

Keçən dərsdə biz sizinlə massiv anlayışıyla tanış olduq və onu dəyərlərlə doldurmağı öyrəndik. Amma bütün dəyərləri biz özümüz yazırdıq, yəni qabaqcadan bilirdik onlar necə olacaqlar. Dəyişkənlərin və massivlərin doldurulmasının belə üsulu proqramın imkanlarını məhdudlaşdırır. əgər istifadəçidən asılı olmayan məlumatı almaq üsulu yoxdursa süni intellektə malik olan nəyisə yaratmaq mümkün deyil. Bəs bizə təsadüfi üsulla seçilmiş dəyərlər lazım olduqda nə edək?

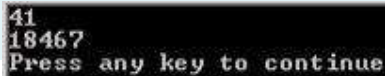
Rand funksiyasından istifadə

C dilində təsadüfi rəqəmi yaratmaq imkanı var. Bu əməliyyat üçün **rand()** adlı funksiyadan istifadə olunur. Bu funksiya kitabxana — **stdlib.h** faylında , beləliklə onun işi üçün bu faylı **# include** təlimatın köməyi ilə qoşmaq lazımdır . **Rand()** çağırışının yerinə ,proqramda təsadüfi rəqəm 0-dan 32767-ə qədər diapazonda qoyulur. Sadə nümunəyə baxaq:

```
#include // bu faylda rand funksiyası saxlanılır rand using namespace std; void  
main() { int a;  
    // Təsadüfi ədədin və yazının a dəyişgəninə generasiyası.  
    a=rand(); cout<<a<<"\n";
```

```
/* Təsadüfi ədədin və yazının a dəyişgəninə təkrar generasiyası.  
*/ a=rand(); cout<<a<<"\n"; }
```

Siz layihəni yaradıb, bizim yerinə yetirilməsi üçün misalları yığıb və buraxdınız halda, məlum olur ki, proqram iki təsadüfi ədədi ardıcılıqla yaradırdı. Məs belə, deyilmi?:



```
41  
18467  
Press any key to continue
```

Yenə qoşun. Və yenidən eyni şəkil. Deməli, təsadüfi rəqəmlər — təsadüfi deyil, bundan əlavə onlar həmçinin qoşulmadan qoşulmaya təkrarlanırlar. Bu iddianın haqsızlığına baxmayaraq — bu belədir və tamamilə aydın niyə. Əgər siz istənilən insana küçədə yaxınlaşıb və tam ədədi sərbəst anlatmaqı xahiş etsəz, bu insan şübhəsiz məhz təsadüfi rəqəmi adlandıracaq.

Ola bilsin yoldan keçən lövhəyə və ya saata baxacaq və görülmüşdən bu ədədi çıxardacaq.

Kompüter canlı varlıqdan fərqli olaraq assosiativ düşüncəyə qadir deyil, buna görə **rand()** funksiyası havadan ədəd almır, təsadüfi ədədlər generatorunun alqoritminin yazılması zamanı təyin edilmiş nöqtəni başlanğıc nöqtəsi kimi istifadə edərək işləyir. Artıq özümüzdə görə bildiyimiz kimi bu nöqtədən irəlilənərək, proqramın müxtəlif çağırışları zamanı bu funksiya eyni ədəd çağırır. Bundan nəticə çıxartmaq olar ki: proqramın müxtəlif çağırışları vaxtı **rand()** müxtəlif ədədlərini verməsi üçün generasiyanın başlanğıc nöqtəsini dəyişdirmək lazımdır.

Srand funksiyasından istifadə

Funksiyanın yerləşdiyi yer — `stdlib.h` kitabxanasıdır. `Srand` funksiyası başlanğıc nöqtəsini təsadüfi ədəd generasiyası üçün yaradır və növbəti sintaksisə malikdi

```
void srand(unsigned int start
```

Funksiya qəbul edən `start` parametri təsadüfi rəqəmin generasiyası üçün yeni nöqtənin elə özüdür. Gəlin düşünək, hansı ədədi bu parametrin yerinə yazmaq. Dəyəri proqramın yazılış dəqiqəsində müəyyən edilmiş tamədədli literal və ya dəyişən əlverişli adlanır. Onları göndərmə nöqtəsinin yerinə çatdırdıqda biz onu hökmən dəyişdirəcəyik, amma dinamik etməyəcəyik. Və digər ədədlər yaradılsa belə hələdə hər işə salınmada eyni olacaqlar.

Göndəriş nöqtəsi kimi literal ilə nümunə:

```
#include<iostream>
```

```
// Bu faylda rand və srand funksiyaları saxlanılır.
```

```
#include<stdlib.h>
```

```
using namespace
```

```
std; void main()
```

```

{
    srand(5)
    ; int a;
// təsadüfi ədədin yaradılması və onun dəyişgənə yazılması
    a=rand();
    cout<<a<<"\n";
}

```

Göndəriş nöqtəsi kimi dəyişkən ilə nümunə:

```

include<iostream>
// Bu faylda rand və srand funksiyaları saxlanılır.
#include<stdlib.h>

using namespace
std; void main()
{
    int start=25;
    srand(start);
    int a;
// təsadüfi ədədin yaradılması və onun dəyişgənə yazılması
    a=rand();
    cout<<a<<"\n";
}

```

Sizinlə bizə, daimi amillərdən asılı olmayaraq dəyişən nəşə lazımdır. Razılaşın ki, zaman belə bir ölçüdür. Və, məhz zamanı biz göndərmə nöqtəsi kimi istifadə edirik.

Time funksiyasından istifadə

Time funksiyasından istifadə. Funksiyanın yeri — **time.h** kitabxanasıdır. **Time** funksiyasının bir neçə təyinatı var və buna ətraflı baxmayacağıq.

Yalnız bizə iş üçün lazım olanı götürək. Və əgər məhz time funksiyasını NULL parametriylə çağırısaq, öz çağırışının yerinə proqramda bu funksiya millisaniyələrin 1970-ci ilin 1 yanvarından keçmiş miqdarını qaytaracaq. Razılaşın ki, bu ölçü hər dəfə müxtəlif olacaq. Və bu — tam, bizim axtardığımız idi. Alınmış informasiyanı topladıqda alınacaq:

```
srand(time(NULL));
```

Srand funksiyası start nöqtəsi kimi 1970 ilin 1 yanvardan keçən millisaniyələrin miqdarı ifadə edən ədədi təyin edir. Yoxlayaq:

```
#include<iostream>
#include<stdlib.h> // Bu faylda rand və srand funksiyaları
                  // saxlanılır
#include<time.h>   // Bu faylda time funksiyası
                  //
var using namespace std;
void main()
{
    srand(time(NULL))
    ; int a;
    // Təsadüfə ədədin yaradılması və onun
    // Dəyişgənə yazılması
    a=rand();
    cout<<a<<"\n";
}
```

Düzəldilmiş nümunəni yığıb, siz əlbəttə əmin oldunuz ki, indi proqramın müxtəlif iş salmaları vaxtı müxtəlif sayılar yaradılır, bununlada generatorun imkanları tükənmir.

Generator üçün diapozunun müəyyən edilməsi

Rand funksiyasının çağırışı yolu ilə alınan ədədlər, 0-dan 32767-ə qədər diapazonda olur. Amma bizə həmişə belə geniş diapason lazım olmur. Bəs 0-dan 10-a qədər və ya 0-dan 100-ə qədər və sair ədədləri yaratmaq lazım olsa biz nə edək?! Yardıma belə hallarda sadə modul üzrə bölmə gəlir. Nümunə üçün sərbəst 23 ədədini götürək. Razılaşın ki, hansı ədədi siz modul üzrə 23-ə bölməsəz, siz ya 0 (əgər qalıq yoxdursa), və ya 1-dən 22-ə qədər diapazonda qalıq alacaqsız. Təsadüfi yaradılmış ədədi modul üzrə bölüb, bu əlamətdən istifadə edək:

```
int a=rand()%23;
```

Bu qaydaya əsasən düsturu çıxarmaq olar:

```
0-dan X-a qədər diapazonda  
    ədəd X: rand() % X
```

Bu qaydaya əsasən düsturu çıxarmaq olar:

0-dan X-a qədər diapazonda ədəd:
`rand() % X`

Amma, diapazon həmişə sıfırdan başlanmır. Deyək ki, bizə 11-dən 16-a qədər diapazon lazımdır. Bu qədər sadə. 0-dan 5-ə qədər ədəd yaratmaq lazımdır (aralarındakı fərq 16 və 11), və sonra alınmış nəticəni 11 vahid tərpətmək.

```
int a=rand()%5+11;
```

Və, artıq dəyisilmiş qaydaya əsaslanaraq düstur çıxarmaq olar:

ЧИСЛО В ДИАПАЗОНЕ ОТ Y ДО X:

$$\text{rand}() \% (X - Y) + Y$$

Beləliklə, biz təsadüfi ədəd generatoru ilə tanış olduq və indi massivlərlə işimizi yüngülləşdirə bilərik. Necə? Dərsin növbəti bölməsində bu barədə danışılacaq.

2. Təsadüfi ədəd generatorundan istifadə

Gəlin təsadüfi ədəd generatorundan istifadənin nümunəsinə baxaq, məhz təsadüfi ədədlərlə massivin doldurulmasına:

```
#include<iostream>
// bu funksiya rand() və srand() funksiyaları saxlanılır
#include<stdlib.h>
#include<time.h> // bu faylda time() funksiyası var
using namespace std;
void main()
{
    srand(time(NULL))
    ; int array[10];
    for (int i=0;i<10;i++)
    {
        // Təsadüfi ədədin və massivin
        // Cari elementinə yazılışı
        array[i]=rand()%100;
        // elementin dəyərini ekrana çıxarılışı
        cout<<array[i]<<"\n";
    }
}
```

1. Yuxarıda gətirilmiş nümunəyə əsasən ekrana təsadüfi doldurulmuş ədədlərlə 10 elementli massiv çıxarılacaq.
2. Dövrün hər iterasiyasında təsadüfi yeni ədəd yaranır.

3. Srand (time (NULL)) sətirinin sayəsində, proqramın hər işə salması zamanı massiv müxtəlif cür doldurulacaq.

4. Massivdə 0-dan 100-ə qədər diapazonda yerləşən ədədlər dəyişiləcək, çünki generasiyanın nəticəsi modul üzrə 100-ə bölünür .

Gördüyünüz kimi, təsadüfi ədəd generatorun istifadəsi sadədir, və indi sizin əlinizdə sizə yalnız məlumatları klaviaturadan daxil etmədən test etmək deyil

3. İkiölçülü massivlər, çoxölçülü massivlərin özəl halı kimi

Bizim artıq massivlər haqda anlayışımız var, keçən dərsdə biz birölçülü massivi müzakirə etdik. Birölçülü massiv — hər bir dəyərin yalnız bir xarakteristikaya - sıra nömrəsi (indeksə) malik olduğu məlumat massividir. Məhz bu indekslə biz konkret elementə müraciət edirik.

Bu gün biz çoxölçülü massivlər haqqında danışacağıq, yəni hər element bir neçə xarakteristika ilə təsvir edilən massivlər haqqında . Çoxölçülü massivin dəyərinin nümunəsi hər şey ola bilər:

1. Şahmat lövhəsi — hər xana iki E2(hərf və rəqəm) ölçüsünə malikdir
2. KVN-in qiyməti—üç ölçülü: jüri heyəti, müsabiqə, komanda .

Massivin C-da maksimal ölçüsü. Biz ikinci adı matrisa olan ikiölçülü massivdə dayanacağıq.

İkiölçülü massiv.

Yaddaşda elan və yerləşmə

İkiölçülü massiv kəsişməsində konkret dəyər yerləşən sətirlər və sütunların cəmidir. İkiölçülü massivi elan etmək asandır, sətirlərin və sütunların miqdarını göstərmək lazımdır.

Bununla belə, burada təkölçülü massivləri elan etdikdə işləyən qaydalar işləyir, yəni sətirlərin və sütunların miqdarları yerinə konstant olmayan və tamədədli dəyərlər göstərmək olmaz.

Ümumi sintaksis:

```
məlumat_tip massivin_adı [sətirlərin_ədədi]
                               [sütunların_ədədi];
```

Nümunə:

```
const int row=3; // sətirlər
const int col=4; // sütunlar
int array[row][col]; // ölçülü massiv row
                     // на col (3x4)
```

	Столбец 0	Столбец 1	Столбец 2	Столбец 3
Строка 0	a [0] [0]	a [0] [1]	a [0] [2]	a [0] [3]
Строка 1	a [1] [0]	a [1] [1]	a [1] [2]	a [1] [3]
Строка 2	a [2] [0]	a [2] [1]	a [2] [2]	a [2] [3]

İndeks столбца
İndeks строки
Имя массива

Baxmayaraq ki, biz massivi matrisa şəklində təqdim edirik, əslində — istənilən ikiölçülü massiv yaddaşda sətir-bə-sətir yerləşir: əvvəlcə sıfırıncı sətir, sonra birinci və sair. Bu barədə unutmayın, çünki massivin hüdudlarından kənara çıxış-proqramın donmasına gətirə bilər. Bununla belə səhv göstərilməyəcək.

a[0][0] a[0][1] a[0][2] a[0][3] a[1][0] a[1][1] a[1][2] a[1][3] a[2][0] a[2][1] a[2][2] a[2][3]

İnizializasiya

İkiölçülü massivin inisializasiyası həmçinin birölçülünün inisializasiyasına analojidir:

1. Yaradılma zamanı inisializasiya..

Hər sətir dalğalı mötərizəyə alınır:

```
int array[2][2]={{1,2},{7,8}};
```

Dəyərlər ardıcıl göstərilir və sətirbəsetir massivə yazılır:

```
int array[2][2]={7,8,10,3};
```

Əgər dəyər buraxılmışdısa, o sıfırla inisializasiya ediləcək:

```
int array[3][3]={{7,8},{10,3,5}};
```

2. . Dövrün köməyi ilə inisializasiya.

Bir sirri açaq ki— ikiölçülü massivi sadəcə sətirlətin yox, birölçülü massivli məcmusi kimi də baxmaq olar. Yəni, bir birölçülü massivi, biz konkret elementlərini seçərək sadə dövrlə doldururuq, amma məcmuda biz başqa massivləri seçməliyik.

Qeyd: Massivin konkret elementinə müraciət sətirin və sütunun nömrəsi ilə edilir, məsələn — `mr [2] [1]` — ikinci sətir və birinci sütunun kəsişməsində uzanan dəyərdir . İkiölçülü massivlə iş birölçülüdən çoxda çətin deil— bunu təcrübədə sübut edək. a məcmuda, biz həmçinin ayrı massivləri seçməliyik.


```
#include<iostream>
#include<stdlib.h> // bu faylda rand
                  // və srand
#include<time.h>   // funksiya saxlanılır
                  //

time using namespace std;

void main()
{
    const int row=3; // sətir
    const int col=3; // sütun
    int mr[row][col]; // row ھا col ölçülü massiv

    /* sətirlərə baxırıq (birölçülü massivlər cəmdə) */
    for(int i=0; i<row; i++)
    {
        // hər sətirin ayrı elementlərinə baxırıq
        for(int j=0; j<col; j++)
        {
            /* elementlərin 0-dan 100 qədər diapazonda
            inisializasiyası */
            mr[i][j]=rand()%100;
            // ekranda dəyərlərin
            göstərilməsi
            cout<<mr[i][j]<<" ";
        }
        // matrisanın o biri sətirinə
        keçid cout<<"\n\n";
    }
}
```

4. Praktiki nümunə

Məsələ.

İkiölçülü massivdə hər sətirin maksimal elementi tapan proqramı yazın.

Realizasiya kodu

```
include<stdlib.h> //bu faylda rand və srand funksiyası saxlanılır
#include<time.h> // bu faylda time funksiyası saxlanılır

using namespace
std; void main()
{
    // Massivin ölçüsünü təyin
    edirik const int m = 3;
    const int n = 2;
    int A[m][n]; // ikiölçülü massiv elan edirik

    // Massivin təsadüfə ədədlərlə doldurulması
    // və ekrana çıxarılması

    // ayrı sətirlərə baxaq
    for(int i=0; i<m; i++)
    {
        // hər sətirin ayrı elementinə baxaq
        for(int j=0; j<n; j++)
        {
            // 0dan 100 qədər elementlərin
            // Dəyərlə inisializasiyası
            A[i][j]=rand()%100;
            // Dəyərlərin ekrana
            çıxarılması
            cout<<A[i][j]<<" ";
        }
    }
```

```

        // matrisanın o biri sətirinə
        keçid cout<<"\n\n";
    }
    cout << "\n\n";

    // Sətirlərdə maksimal elementin axtarışı
    // Ayrı sətirlərə baxaq
    for (int i=0; i<m; i++){
        // Ehtimal edirik ki sətirin -
        // Maksimal-sıfırıncı elementi
        int max = A[i][0];
        // Cari sətirdə ki maksimal
        // Elementin axtarışı

        // Cari sətir üçün sütunun
        // Indeksini dəyişməsi
        for (int j=0; j<n; j++){
            {
                if (A[i][j] > max)
                    max = A[i][j];
            }
            cout << "Maksimal element " << i
                << "-ой строки = " << max << endl;
        }
    }
}

```

Diqqət yetirin!

1.Dövrün hər təkrarında maksimum şəkilində cari sətirin sıfırıncı elementi seçilir.

2. Konkret sətirin analizindən sonra tapılmış maksimum ekrana çıxarılır.

5. Ev tapşırığı

1. 3×4 -ölçüsündə ikiölçülü massiv verilib. Dəyəri sıfıra bərabər olan elementlərin miqdarını tapmaq lazımdır.
 2. N (sətirlərin n -i, sütunların n -i) sıralı kvadrat matrisi verilmişdir. Matrisin tünd-göy hissələrində yerləşdirilmiş elementin ən böyük dəyərini tapın.
- Bu ev tapşırığında bütün massivlər təsadüfən dolur.



a



b



v



z



d



e



ж



3



u



κ



Dərs №7

Çoxölçülü massivlər

© Компьютерная Академия
«Шаг» www.itstep.org

Все права на охраняемые авторским правом фото-, аудио- и видеопроизведения, фрагменты которых использованы в материале, принадлежат их законным владельцам. Фрагменты произведений используются в иллюстративных целях в объеме, оправданном поставленной задачей, в рамках учебного процесса и в учебных целях, в соответствии со ст. 1274 ч. 4 ГК РФ и ст. 21 и 23 Закона Украины «Про авторське право і суміжні права». Объем и способ цитируемых произведений соответствует принятым нормам, не наносит ущерба нормальному использованию объектов авторского права и не ущемляет законные интересы автора и правообладателей. Цитируемые фрагменты произведений на момент использования не могут быть заменены альтернативными, не охраняемыми авторским правом аналогами, и как таковые соответствуют критериям добросовестного использования и честного использования.

Все права защищены. Полное или частичное копирование материалов запрещено. Согласование использования произведений или их фрагментов производится с авторами и правообладателями. Согласованное использование материалов возможно только при указании источника.

Ответственность за несанкционированное копирование и коммерческое использование материалов определяется действующим законодательством Украины