ЖИТОМИРСЬКА ПОЛІТЕХНІКА
ДЕРЖАВНИЙ УНІВЕРСИТЕТ

# Лекція 13.
# Двовимірні масиви.

| -5 | 6 | 3 | -7 | 3 | 1 | 10 | 1 | -3 | 5 | 8 |
|---|---|---|---|---|---|---|---|---|---|---|

**Масив** - це безперервна ділянка пам'яті, що містить послідовність об'єктів однакового типу і позначається одним ім'ям

**Адреса масиву** - адреса початкового елемента масиву.

**Ім'я масиву** - ідентифікатор, який використовується для звернення до елементів масиву.

**Розмір масиву** - кількість елементів масиву.

**Розмір елемента** - кількість байт, що займає один елемент масиву.

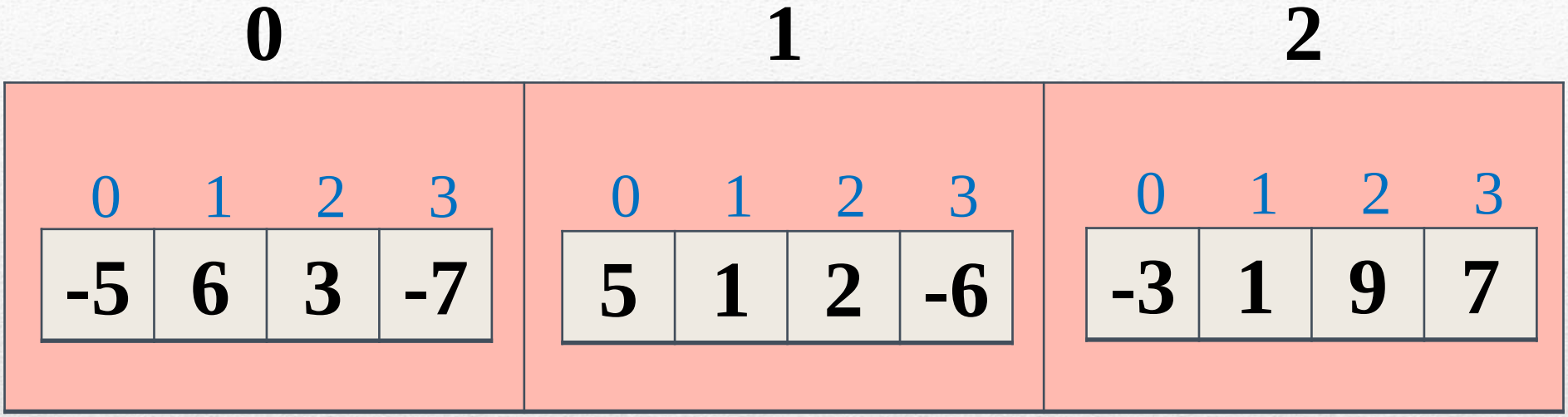
В СІ масиви зберігаються у вигляді групи послідовних комірок одного типу, об'єднаних під одним єдиним ім'ям.

Ім'я масиву є покажчиком (вказує на адресу в пам'яті).

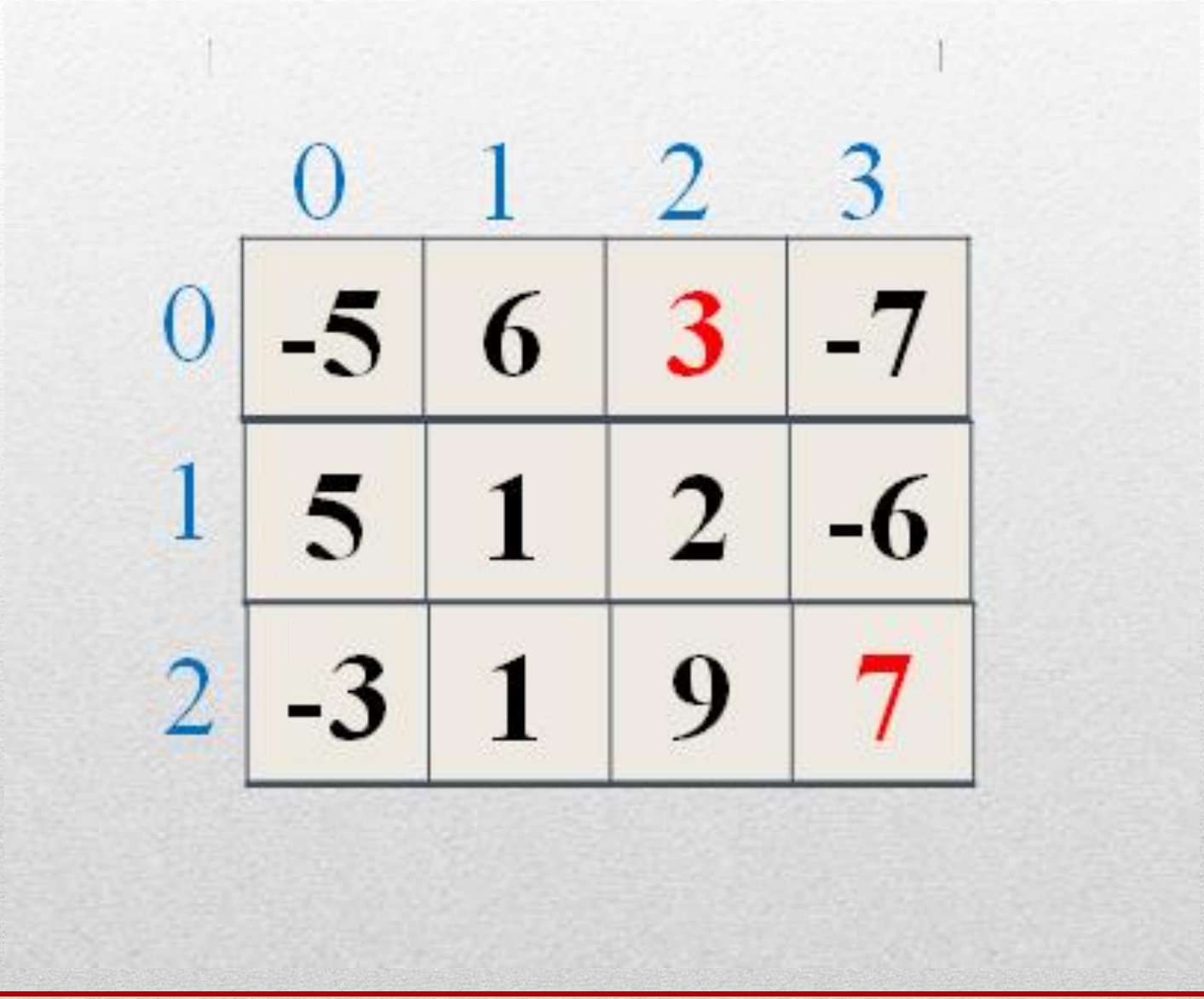
Масиви можуть мати як одне, так і більш одного вимірювань. Залежно від кількості вимірювань масиви діляться на одномірні, двовимірні, тривимірні і так далі до n-мірного масиву

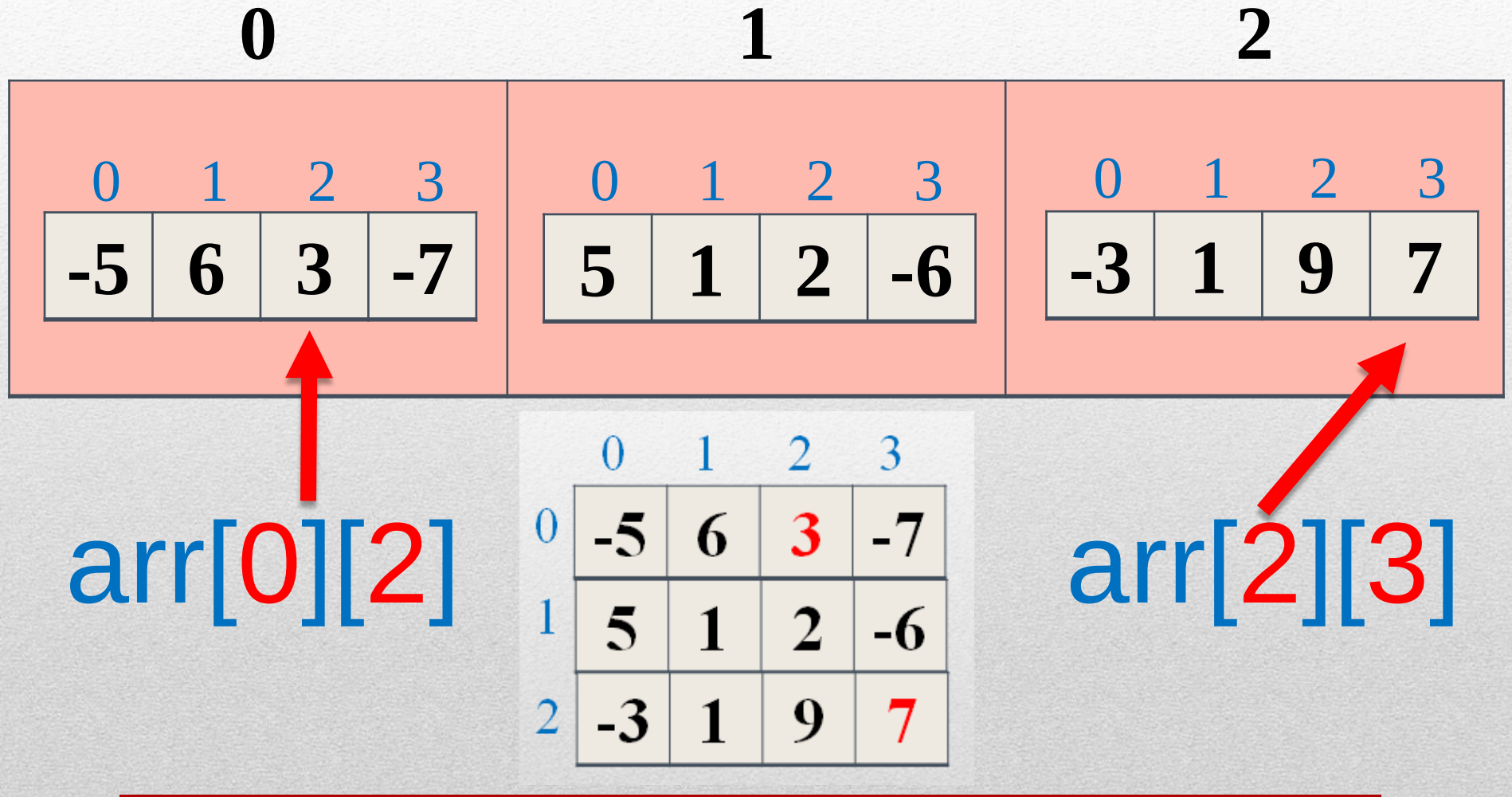
# Двовимірні масиви

У двовимірному масиві, крім кількості елементів масиву, є такі характеристики як, *кількість рядків і кількість стовпців* двовимірного масиву. Тобто, візуально, двовимірний масив - це звичайна таблиця, з рядками і стовпцями.



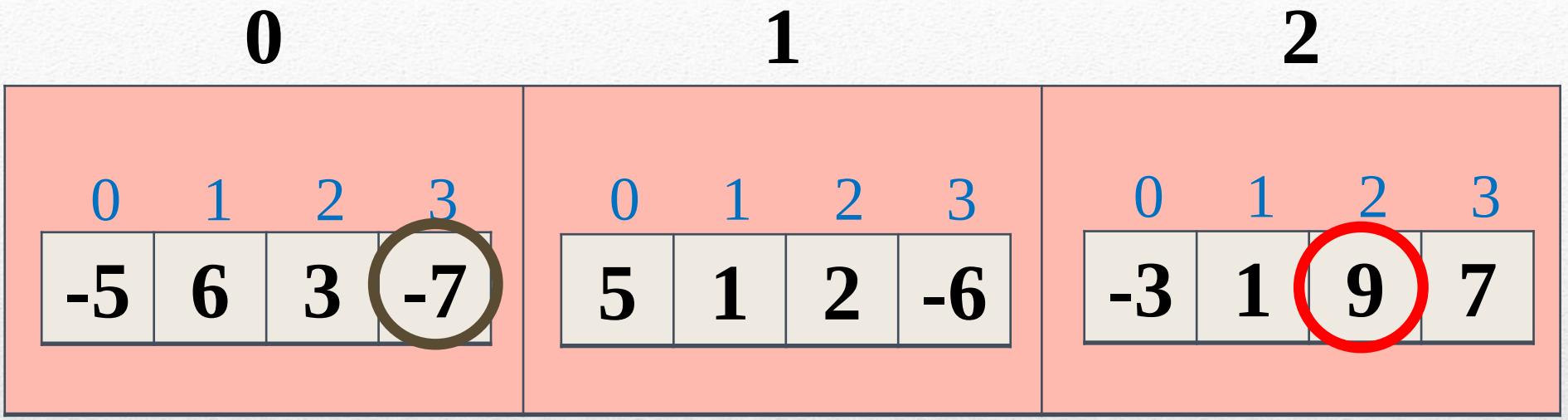

***Двовимірний масив*** *– це набір однотипних елементів, доступ до яких здійснюється з використанням двох індексів.*


0 1 2 3
0 -5 6 3 -7
1 5 1 2 -6
2 -3 1 9 7


0
1
2
0 1 2 3
-5 6 3 -7
0 1 2 3
5 1 2 -6
0 1 2 3
-3 1 9 7
arr[0][2]
arr[2][3]
0 1 2 3
0 -5 6 3 -7
1 5 1 2 -6
2 -3 1 9 7

***Двовимірні масиви*** *для зручності представляють у вигляді* ***матриць*** *(таблиць, де клітинки впорядковані за рядками і стовпцями)*

| | | | |
|---|---|---|---|
| **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]** |
| **a[3][0]** | **a[3][1]** | **a[3][2]** | **a[3][3]** |




| | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| 0 | -5 | 6 | 3 | -7 |
| 1 | 5 | 1 | 2 | -6 |
| 2 | -3 | 1 | 9 | 7 |

## Синтаксис оголошення матриці:

**тип** **ім‘я масиву** **[ n ] [ m ] ;**

*кількість рядків*

*кількість стовпців*

Кількість рядків та стовпців повинна бути вказана явно у вигляді числа

<Тип даних> <ім'я масиву> [*кількість рядків*] [*кількість стовпців*];

Двовимірний масив, який має п'ять рядків і три стовпці запишеться у вигляді:

```
int arr [5] [3];
```

де:

arr - ім'я целочисленного масиву,

число в перших квадратних дужках вказує кількість рядків двовимірного масиву, в даному випадку їх 5;

число у других квадратних дужках вказує кількість стовпців двовимірного масиву, в даному випадку їх 3.

## Приклади оголошення матриць:

```
double matrDbl[20][15];
int matrInt[12][50];
char matrChar[1][33];
```

Далі для роботи з елементом матриці (двовимірного масиву), потрібно вказувати у квадратних дужках номер рядка і номер стовпця елемента:

```
matrDbl[5][12] = 12.5;
```

# Приклади оголошення матриць:

```c
#include <stdio.h>
int main(){
    int a[2][3] = { 1, 2, 3, 4, 5, 6 };
    printf("%d %d %d\n", a[0][0], a[0][1], a[0][2]);
    printf("%d %d %d\n", a[1][0], a[1][1], a[1][2]);
return 0;
}
```

```
1 2 3
4 5 6
Press any key to continue . . .
```

```c
int main(){
    int a[][3] = { 1, 2, 3, 4, 5, 6 };
    printf("%d %d %d\n", a[0][0], a[0][1], a[0][2]);
    printf("%d %d %d\n", a[1][0], a[1][1], a[1][2]);
return 0;
}
```

```
1 2 3
4 5 6
Press any key to continue . . .
```

Допускається не вказувати кількість рядків в двовимірному масиві (вказуються порожні квадратні дужки).

У такому випадку розмір масиву буде визначено за кількістю ініційованих значень стовпців.

Кількість стовпців матриці завжди необхідно вказувати.

Наприклад:

```
double b[][4] = {{1,2,3,4},{5,6,7,8}}
```

```
int main(){
    int array[3][5] =
    {
    { 1, 2, 3, 4, 5 },//рядок №0
    { 6, 7, 8, 9, 10 }, //рядок №1
    { 11, 12, 13, 14, 15 } //рядок №2
    };

    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 5; j++)
                printf("%3d", array[i][j]);
        printf("\n");
    }
system("pause");
return 0;
}
```

```
  1  2  3  4  5
  6  7  8  9 10
 11 12 13 14 15
Press any key to continue . . .
```

```
int main(){
int array[3][5] =
{
{ 2, 4  }, // рядок №0
{ 1, 3, 7 }, // рядок №1
{ 8, 9, 11, 12 } // рядок №2
};

for (int i = 0; i < 3; i++) {
    for (int j = 0; j < 5; j++)
        printf("%3d", array[i][j]);
    printf("\n");
}
system("pause");
return 0;
}
```

```
  2  4  0  0  0
  1  3  7  0  0
  8  9 11 12  0
Press any key to continue . . .
```

***Введення і виведення*** матриць в мові СІ здійснюється поелементно. Так як матриця має подвійну розмірність, то введення і виведення здійснюється у вкладених циклах.

Наприклад:

```
double a[5][10];
for(int i=0;i<5;i++)
    for(int j=0;j<10;j++)
        scanf("%lf”,&a[i][j]);
...
for(int i=0;i<5;i++){
    for(int j=0;j<10;j++)
        printf("%8.2f”,a[i][j]);
    printf("\n”);   }
```

```c
#include <stdio.h>
int main(){
    int a[2][3];
    int i, j;
    for (i = 0; i < 2; i++)
        for (j = 0; j < 3; j++) {
        printf("a[%d][%d] = ", i, j);
        scanf("%d", &a[i][j]);
        }
    for (i = 0; i < 2; i++) {
        for (j = 0; j < 3; j++)
        printf("%4d ", a[i][j]);
        printf("\n");
    }
return 0;
}
```

```
a[0][0] = 41
a[0][1] = -5
a[0][2] = 6
a[1][0] = -78
a[1][1] = 12
a[1][2] = 5
  41   -5    6
 -78   12    5
```

```
const int n = 3, m = 5;
int arrA[n][m], arrL[n][12];
double arrX[10][12];
for (int i = 0; i < n; i++) {
    for (int j = 0; j < m; j++) {
        arrA[i][j] = rand() % 10;
        printf("%3d", arrA[i][j]);
    }
    printf("\n");
}
for (int i = 0; i < n; i++) {
    for (int j = 0; j < 12; j++) {
        arrL[i][j] = rand() % 10-5;
        printf("%3d", arrL[i][j]);
    }
    printf("\n");
}
for (int i = 0; i < 10; i++) {
    for (int j = 0; j < 12; j++) {
        arrX[i][j] = (double)rand()*(5-2.85)/RAND_MAX+2.85;
        printf("%5.2f", arrX[i][j]);
    }
    printf("\n");
}
```

```
 5  6  5  3  2
 4  2  5  1  6
 5  9  4  5  1
 1 -4  0 -5 -5  3  3 -5  3 -5  2 -4
 0 -4 -1 -2 -2 -3  1  2 -3  2  0  0
-4 -1  0  2  0 -1 -3  4 -3 -5 -2  2
4.52 3.07 4.09 3.60 3.48 4.17 4.87 3.66 4.01 4.88 4.16 4.59
4.10 4.48 3.82 2.87 2.91 4.23 2.98 4.33 4.44 3.98 4.02 3.27
3.01 4.17 4.97 4.65 2.98 3.88 3.18 3.40 3.70 4.41 4.46 3.67
4.27 4.75 4.43 2.93 4.38 4.87 4.99 4.26 4.18 3.22 4.71 3.27
2.85 4.21 4.64 4.52 3.52 2.96 4.86 2.94 4.64 3.83 3.03 4.63
4.65 3.32 4.95 3.02 4.74 4.82 4.31 4.32 3.99 4.37 4.23 3.07
3.98 3.96 3.25 3.82 4.33 4.71 3.41 4.95 4.14 4.01 3.82 3.98
3.56 4.90 2.99 4.99 3.36 3.84 3.61 3.49 3.16 3.63 3.96 4.80
4.74 4.96 4.41 3.83 4.01 3.85 4.47 3.96 3.67 4.35 3.63 4.86
4.58 3.01 3.24 4.91 3.36 3.65 3.17 3.88 3.77 4.09 3.26 4.10
Press any key to continue . . .
```

## Увага!!!

```
for (int i = 0; i < rows; i++)
     for (int j = 0; j < cols; j++)
         arr[i][j] = rand() % 21 - 10;
```

```
for (int j = 0; j < cols; j++)
     for (int i = 0; i < rows; i++)
         arr[i][j] = rand() % 21 - 10;
```

Перший буде здійснювати проходи по рядках, а другий – по стовпцях:

```c
#define MAX_ROWS 50
#define MAX_COLS 50
int main() {
   int rows, cols;
   int arr[MAX_ROWS][MAX_COLS];
   printf("rows = "); scanf("%d", &rows);
   printf("cols = "); scanf("%d", &cols);
   srand(time(0));
   printf("arr = {");
   for (int i = 0; i < rows; i++)
      for (int j = 0; j < cols; j++)
         arr[i][j] = rand() % 21 - 10;
return 0;
}
```

*обхід матриці по рядках*

```
for (int i = 0; i < rows; i++)
     for (int j = 0; j < cols; j++)
```

| | | | | |
|---|---|---|---|---|
| 1 | 12 | 7 | 11 | 9 |
| -3 | -4 | 7 | 10 | -2 |
| 4 | -6 | -1 | -5 | 11 |
| 1 | 4 | 6 | 12 | 9 |
| 13 | 11 | 13 | 14 | 12 |
| 3 | -3 | 8 | 13 | -5 |
| 9 | 16 | 10 | 21 | 14 |

*обхід матриці по рядках*

```
#define MAX_ROWS 50
#define MAX_COLS 50
int main() {
   int rows, cols;
   int arr[MAX_ROWS][MAX_COLS];
   printf("rows = "); scanf("%d", &rows);
   printf("cols = "); scanf("%d", &cols);
   srand(time(0));
   printf("arr = {");
   for (int j = 0; j < cols; j++)
      for (int i = 0; i < rows; i++)
         arr[i][j] = rand() % 21 - 10;
   return 0;
}
```

*обхід матриці по стовпцях*

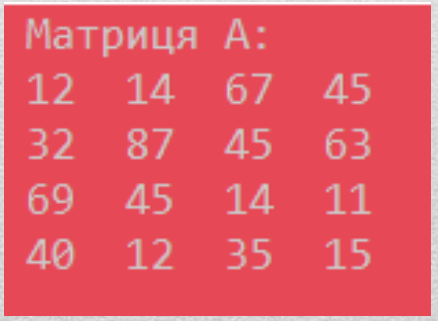
```
for (int j = 0; j < cols; j++)
    for (int i = 0; i < rows; i++)
```

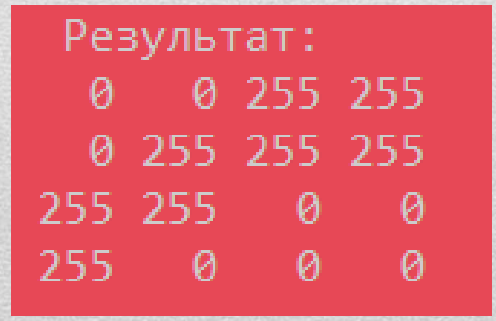
| | | | | |
|---|---|---|---|---|
| 1 | 12 | 7 | 11 | 9 |
| -3 | -4 | 7 | 10 | -2 |
| 4 | -6 | -1 | -5 | 11 |
| 1 | 4 | 6 | 12 | 9 |
| 13 | 11 | 13 | 14 | 12 |
| 3 | -3 | 8 | 13 | -5 |
| 9 | 16 | 10 | 21 | 14 |

*обхід матриці по стовпцях*

Яскравість пікселів зображення закодовані числами від 0 до 255 у вигляді матриці. Перетворити зображення в чорно-біле за наступним алгоритмом:

- обчислити середню яскравість пікселів по всьому зображенню
- всі пікселі, яскравість яких менша за середню, зробити чорними (записати код 0), а решта - білими (код 255)



```
system("chcp 1251"); system("cls");
const int n = 4; int summ = 0;
int arr[n][n]={12,14,67,45, 32,87,45,63, 69,45,14,11, 40,12,35,15};
printf("  Матриця A:\n");
for (int i = 0; i < n; i++) {
     for (int j = 0; j < n; j++)
         printf("%4d", arr[i][j]);
    printf("\n");
}
for (int i = 0; i < n; i++)
    for (int j = 0; j < n; j++)
            summ += arr[i][j];
double avg = (double)summ / (n*n);
printf("Середнє значення= %f", avg);
for (int i = 0; i < n; i++)
    for (int j = 0; j < n; j++)
        if (arr[i][j]<avg) arr[i][j]=0;
        else arr[i][j] = 255;
printf("\n  Результат:\n");
for (int i = 0; i < n; i++) {
    for (int j = 0; j < n; j++) printf("%4d", arr[i][j]);
printf("\n");
}
```

?

Знайти максимальний елемент і суму елементів матриці розміром [m x n]

Для роздумів.
Яка помилка у наступному коді?

```
for (int i = 0; i < cols; i++)
     for (int j = 0; j < rows; j++)
         arr[i][j] = rand() % 21 - 10;
```

У мові СІ допускається створення масивів розмірністю три і більше (тобто тривимірних, чотиривимірних тощо). Наприклад, оголошення тривимірного цілочисленного масиву з ініціалізацією матиме вигляд:

```
int a[2][2][2]={ //це тривимірний масив
    {{1,2},{3,4}},
    {{5,6},{7,8}}
};
```

Введення, виведення та інша обробка такого масиву здійснюється аналогічно обробці двовимірного масиву, тільки вже в трьох вкладених циклах.