

Масиви



Лекція №10

Дисципліна «Програмування»



Основні поняття

Масив – це неперервний фрагмент пам'яті, який містить послідовність об'єктів однакового типу та позначається одним іменем.

Елемент масиву (значення елемента масиву) – значення, яке зберігається у визначеній комірці пам'яті, розташованій в межах масиву, а також адреса цієї комірки пам'яті.

Кожен елемент масиву характеризується трьома величинами:

- **адресою** елемента – адресою початкової комірки пам'яті, в якій розташований цей елемент;
- **індексом** елемента (порядковим номером елемента в масиві);
- **значенням** елемента.



Основні поняття

Адреса масиву – адреса початкового елемента масиву.

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

Розмір масиву – кількість елементів масиву.

Розмір елемента масиву – кількість байтів, які займає один елемент масиву.

Оголошення масиву

```
float mas[20];
```

повідомляє про те, що **mas** є масивом, який складається з **20** елементів, кожен з яких може містити у собі значення типу **float**. Першим елементом масиву буде **mas[0]**, другим – **mas[1]**, останнім – **mas[19]**.



Робота з елементами масиву

Кожному елементу масиву може бути присвоєно значення типу **float**.

Наприклад, можна записати наступний код:

```
mas[5] = 32.54;  
mas[6] = 1.2e+21;
```

Можна прочитати значення типу **float** та помістити його в конкретний елемент масиву:

```
// зчитування значення типу float та розміщення  
// його у 5-му елементі масиву
```

```
scanf ("%f", &mas[4]);
```



Символьні масиви

В масиві типу **char** можна зберігати як символи, так і рядки. Вміст масиву **char** формує рядок, якщо масив містить нульовий символ (**'\0'**).

Масив символів, який не утворює рядок

'М'	'а'	'с'	'и'	'в'	'_'	'с'	'и'	'м'	'в'	'о'	'л'	'і'	'в'	'.'
-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----

Масив символів, який утворює рядок за рахунок символу **'\0'**

'М'	'а'	'с'	'и'	'в'	'_'	'с'	'и'	'м'	'в'	'о'	'л'	'і'	'в'	'.'	'\0'
-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	------

▲
нульовий
символ

Зберігання масивів в пам'яті

Числа, які застосовуються для ідентифікації елементів масиву, звуться **індексами** або **зміщеннями**.

Індекси повинні бути цілими числами. Елементи масиву зберігаються в пам'яті поруч один з одним

`int mas1[4];` (4 байта на значення типу `int`)

2980	45	4728	5
<code>mas1[0]</code>	<code>mas1[1]</code>	<code>mas1[2]</code>	<code>mas1[3]</code>

`char mas2[4];` (1 байт на значення типу `char`)

'л'	'і'	'т'	'о'
<code>mas2[0]</code>	<code>mas2[1]</code>	<code>mas2[2]</code>	<code>mas2[3]</code>



Використання циклів з масивами

```
#define SIZE 10

...

int main(void)
{
    int index, score[SIZE];
    int sum = 0;
    ...    // Запис цілих чисел до елементів масиву score
    for(index = 0; index < SIZE; index++)
        scanf("%d", &score[index]);
    ...    // Вивід значень елементів масиву на консоль
    for(index = 0; index < SIZE; index++)
        printf("%5d", score[index]);
    ...    // Використання елементів масиву у виразах
    for(index = 0; index < SIZE; index++)
        sum += score[index];
```



Ініціалізація масивів

Масиви часто використовуються для зберігання даних, які необхідні для подальшої роботи програми.

Зручно ініціалізувати масив на початку програми:

```
int main(void)
{
    int powers[8] = { 1, 2, 4, 6, 8, 16, 32, 64 };
}
```

Першому елементу (`powers[0]`) присвоюється значення **1**, другому (`powers[1]`) – значення **2** і т. д.



Ініціалізація масивів

```
#include <stdio.h>
#include <windows.h>
#define MONTHS 12

int main(void)
{
    const int days[MONTHS] = { 31, 28, 31, 30, 31, 30,
                               31, 31, 30, 31, 30, 31 };

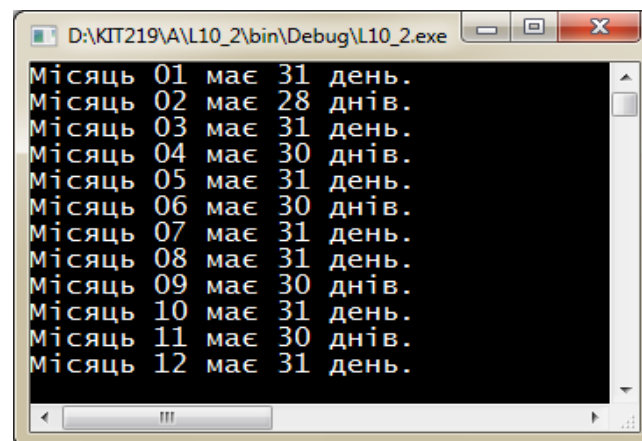
    int index;

    SetConsoleOutputCP(1251);

    for(index = 0; index < MONTHS; index++)
    {
        switch(index + 1)
        {
            case 1:
            case 3:
```

Ініціалізація масивів

```
case 5:
case 7:
case 8:
case 10:
case 12: printf("Місяць %02d має %2d день.\n",
               index + 1, days[index]);
        break;
default: printf("Місяць %02d має %2d днів.\n",
               index + 1, days[index]);
        break;
    }
}
return 0;
}
```



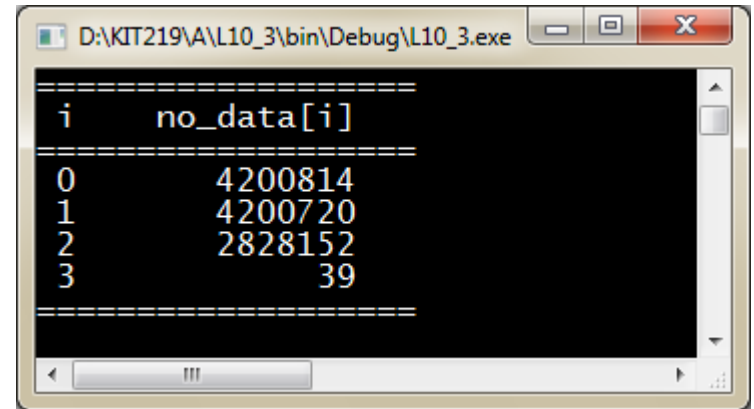
```
D:\KIT219\A\L10_2\bin\Debug\L10_2.exe
Місяць 01 має 31 день.
Місяць 02 має 28 днів.
Місяць 03 має 31 день.
Місяць 04 має 30 днів.
Місяць 05 має 31 день.
Місяць 06 має 30 днів.
Місяць 07 має 31 день.
Місяць 08 має 31 день.
Місяць 09 має 30 днів.
Місяць 10 має 31 день.
Місяць 11 має 30 днів.
Місяць 12 має 31 день.
```

Якщо не виконати ініціалізацію

```
#include <stdio.h>
#define SIZE 4

int main(void)
{
    // масив без ініціалізації
    int no_data[SIZE];
    int i;

    printf("=====\n");
    printf("%2s%14s\n", "i", "no_data[i]");
    printf("=====\n");
    for(i = 0; i < SIZE; i++)
        printf("%2d%14d\n", i, no_data[i]);
    printf("=====\n\n");
    return 0;
}
```



i	no_data[i]
0	4200814
1	4200720
2	2828152
3	39

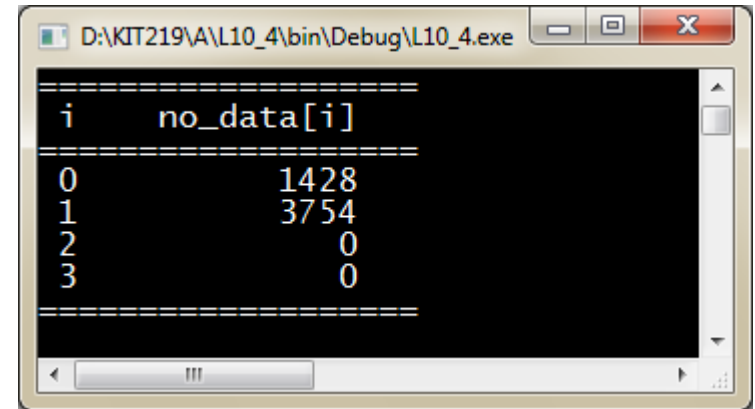


Часткова ініціалізація

```
#include <stdio.h>
#define SIZE 4
```

```
int main(void)
{
    int no_data[SIZE] = { 1428, 3754 };

    printf("=====\n");
    printf("%2s%14s\n", "i", "no_data[i]");
    printf("=====\n");
    for(int i = 0; i < SIZE; i++)
        printf("%2d%14d\n", i, no_data[i]);
    printf("=====\n\n");
    return 0;
}
```



```
=====  
i      no_data[i]  
=====  
0      1428  
1      3754  
2       0  
3       0  
=====
```



Визначення кількості елементів

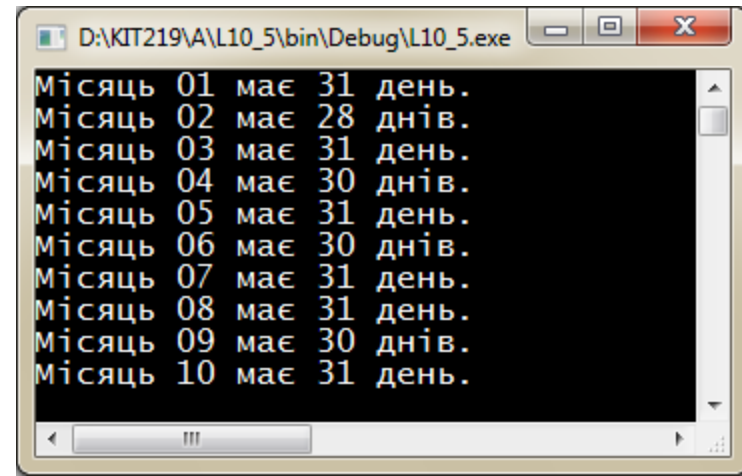
```
#include <stdio.h>
#include <windows.h>

int main(void)
{
    int days[] = { 31, 28, 31, 30, 31, 30, 31, 31, 30, 31 };
    int index;

    SetConsoleOutputCP(1251);

    for(index = 0; index < sizeof days/sizeof days[0]; index++)
    {
        switch(index + 1)
        {
            case 1:
            case 3:
            case 5:
            case 7:
            case 8:
```

Визначення кількості елементів



```
D:\KIT219\A\L10_5\bin\Debug\L10_5.exe
Місяць 01 має 31 день.
Місяць 02 має 28 днів.
Місяць 03 має 31 день.
Місяць 04 має 30 днів.
Місяць 05 має 31 день.
Місяць 06 має 30 днів.
Місяць 07 має 31 день.
Місяць 08 має 31 день.
Місяць 09 має 30 днів.
Місяць 10 має 31 день.
```

```
case 10: printf("Місяць %02d має %2d день.\n",
               index + 1, days[index]);
        break;
default: printf("Місяць %02d має %2d днів.\n",
               index + 1, days[index]);
        break;
```

```
    }
}
return 0;
```

```
}
```



Призначені ініціалізатори

В стандарті **C99** додана нова можливість – **призначені ініціалізатори**. Цей засіб дозволяє обирати, які саме елементи будуть ініціалізовані. Припустимо, що необхідно ініціалізувати тільки останній елемент в масиві. За допомогою традиційного синтаксису ініціалізації мови **C** знадобиться також ініціалізувати усі елементи, що передують останньому:

```
int mas[6] = { 0, 0, 0, 0, 0, 212 };
```

Стандарт **C99** дозволяє застосовувати в списку ініціалізації індекс у квадратних дужках, щоб вказати конкретний елемент:

```
// ініціалізація елемента mas[5] значенням 212  
int mas[6] = { [5] = 212 };
```

Призначені ініціалізатори

```
#include <stdio.h>
#define MONTHS 12
```

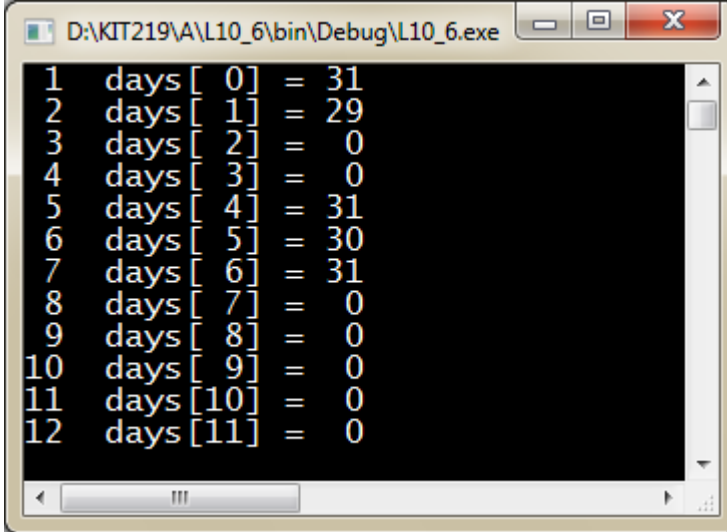
```
int main(void)
{
```

```
    int days[MONTHS] = { 31, 28, [4] = 31, 30, 31, [1] = 29 };
    int i;
```

```
    for(i = 0; i < MONTHS; i++)
```

```
        printf("%2d  days[%2d] = %2d\n", i + 1, i, days[i]);
    printf("\n");
    return 0;
```

```
}
```



```
1  days [ 0] = 31
2  days [ 1] = 29
3  days [ 2] =  0
4  days [ 3] =  0
5  days [ 4] = 31
6  days [ 5] = 30
7  days [ 6] = 31
8  days [ 7] =  0
9  days [ 8] =  0
10 days [ 9] =  0
11 days [10] =  0
12 days [11] =  0
```




Присвоивания значений элементам

```
#include <stdio.h>
#define SIZE 10

int main(void)
{
    int counter, evens[SIZE];

    for(counter = 0; counter < SIZE; counter++)
    {
        evens[counter] = 2 * counter;
        printf("evens[%d] = %3d\n", counter, evens[counter]);
    }
    printf("\n");
    return 0;
}
```

```
D:\KIT219\A\L10_8\bin\Debug\L10_8.exe
evens[0] = 0
evens[1] = 2
evens[2] = 4
evens[3] = 6
evens[4] = 8
evens[5] = 10
evens[6] = 12
evens[7] = 14
evens[8] = 16
evens[9] = 18
```



Масиви змінної довжини

```
#include <stdio.h>
#include <windows.h>
#include <stdlib.h>
#include <time.h>

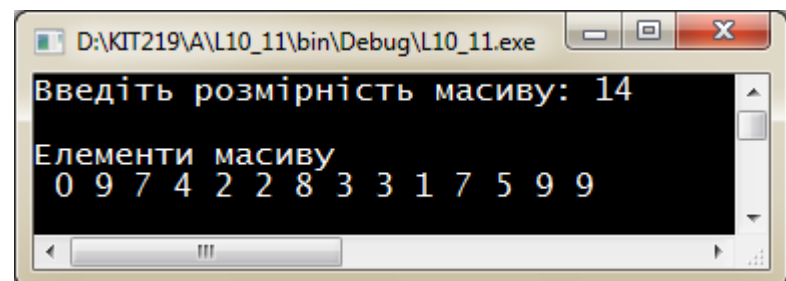
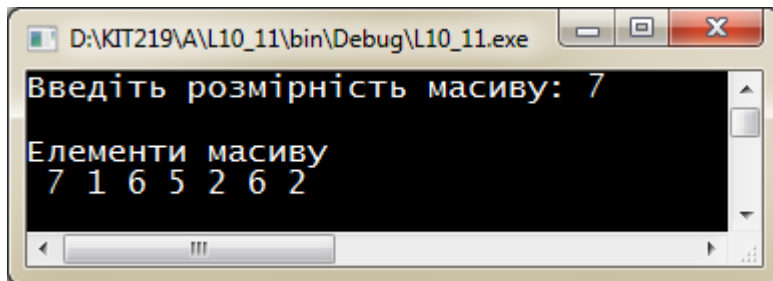
int main(void)
{
    int n;

    SetConsoleOutputCP(1251);

    printf("Введіть розмірність масиву: ");
    scanf("%d", &n);
    int mas[n];           // Масив змінної довжини оголошується
                          // лише після введення розмірності n

    srand(time(NULL));
```

Масиви змінної довжини



```
printf("\nЕлементи масиву\n");  
for(int i = 0; i < n; i++)  
{  
    mas[i] = rand() % 10;  
    printf("%2d", mas[i]);  
}  
printf("\n\n");  
return 0;  
}
```



Двовимірні масиви

// масив з 5 масивів по 6 елементів типу float

```
float rain[5][6];
```

rain[0][0]	rain[0][1]	rain[0][2]	rain[0][3]	rain[0][4]	rain[0][5]
rain[1][0]	rain[1][1]	rain[1][2]	rain[1][3]	rain[1][4]	rain[1][5]
rain[2][0]	rain[2][1]	rain[2][2]	rain[2][3]	rain[2][4]	rain[2][5]
rain[3][0]	rain[3][1]	rain[3][2]	rain[3][3]	rain[3][4]	rain[3][5]
rain[4][0]	rain[4][1]	rain[4][2]	rain[4][3]	rain[4][4]	rain[4][5]

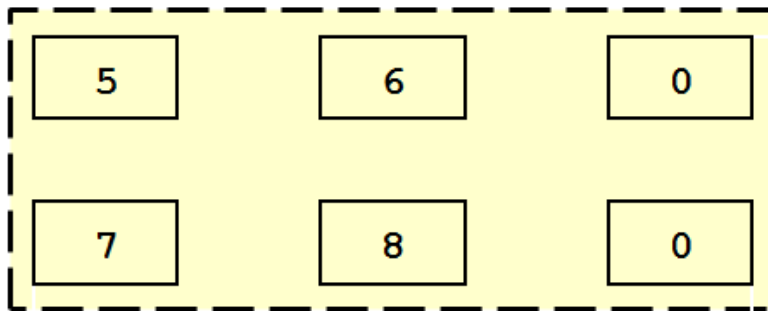
```
const float rain[][6] =  
{  
    { 4.3, 4.3, 4.3, 3.0, 2.0, 1.2 },  
    { 8.5, 8.2, 1.2, 1.6, 2.4, 0.0 },  
    { 9.1, 8.5, 6.7, 4.3, 2.1, 0.8 },  
    { 7.2, 9.9, 8.4, 3.3, 1.2, 0.8 },  
    { 7.6, 5.6, 3.8, 2.8, 3.8, 0.2 }  
};
```



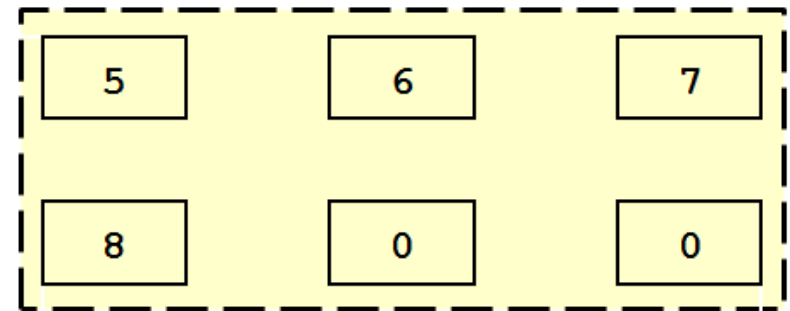
Двовимірні масиви

```
for(year = 0, total = 0; year < YEARS; year++)
{
    for(month = 0, subtot = 0; month < MONTHS; month++)
        subtot += rain[year][month];
    printf("%5d %15.1f\n", 2014 + year, subtot);
    total += subtot;
}
```

Методи ініціалізації двовимірного масиву



```
int sq[2][3] = { {5,6},{7,8} };
```



```
int sq[2][3] = { 5,6,7,8 };
```