

Міністерство освіти і науки України
Національний університет “Львівська політехніка”



Лабораторна робота №15-16
з дисципліни «Програмування
частина 2»

Виконав:
Студент групи АП-11
Братейко Вадим

Прийняв:
Чайковський І.Б.

Львів 2024

«Дослідження використання одновимірних та багатовимірних масивів»

Мета роботи: навчитися використовувати одновимірні та багатовимірні масиви у процесі програмування для обробки великої сукупності значень.

Теоретичні відомості

В усіх програмах, що розглядалися у лабораторних роботах, оброблялися поодинокі значення. На практиці часто виникає потреба обробити єдиним алгоритмом велику сукупність однорідних значень. В математиці такі сукупності мають позначення на зразок x_1, x_2, \dots, x_n . Для підтримки обробки таких сукупностей в мові C існує поняття масиву. Масив - це сукупність даних одного типу, що об'єднані спільним ім'ям.

```
1 int m[10], k=3;  
2 m[0]=1;  
3 m[k]=8;  
4 ++ k;  
5 m[k]=8;  
6 m[(k+2)%3+1]=17;  
7 m[k+3]=m[0]+m[k];  
8 scanf("%d",&m[k+1]);  
9 printf("%d\n",m[k]);
```

В першому рядку оголошується масив m з 10 елементів та допоміжна змінна k , яка одразу отримує початкове значення 3.

В рядку 2 показано, як присвоїти значення елементу масиву, номер якого заздалегідь відомий: в якості індексу використано константу, число 0. Оскільки нумерація елементів починається з 0, то даний оператор означає, що значення присвоюється першому елементу масиву.

Рядок 3 ілюструє, що індекс може бути не константою, а значенням змінної. Оскільки в даний момент змінна k має значення 3, даний оператор означає, що значення 8 присвоюється у четвертий від початку (а не третій!) елемент масиву.

Оператор в рядку 4 збільшує значення змінної k на 1, отже, воно тепер дорівнює 4. Тому, хоча оператор в рядку 5 повністю співпадає за написанням з оператором в рядку 3, тепер вираз в лівій частині присвоювання означає вже не четвертий, а п'ятий від початку елемент масиву.

Рядок 6 є прикладом того, що в якості індексу може використовуватися не лише значення змінної, але і складний вираз. Підставивши поточне значення змінної k , маємо, що значення 17 буде присвоєно елементові з індексом 1, тобто другому елементу масиву.

В рядку 7 показано, що звертання до елементів одного й того самого масиву може здійснюватися і в лівій, і в правій частинах присвоювання. В перший (з індексом 0) елемент раніше було занесене значення 1, поточне значення змінної k дорівнює 4, а елементу з індексом 4 було присвоєно значення 8. Отже, елемент з індексом 7 (восьмий від початку) отримає значення 9

Значення елементів масиву можна вводити з клавіатури так само, як і значення звичайних змінних, за допомогою функції `scanf`, що показано в рядку 8.

Як і завжди, перед іменем змінної, в яку треба розмістити введене значення, ставиться знак & - амперсанд.

З рядка 9 видно, що значення елементів масиву можна передавати до функцій в якості аргументів, в тому числі - друкувати на екран. Одразу ж при оголошенні масиву можна присвоювати значення його елементам, або, як кажуть, ініціалізувати масив.

Функції для обробки рядків

Функція	Виконувана дія
strcpy(s1,s2)	Копіювання s2 в s1
strcat(s1,s2)	Конкатенація (приєднання) s2 в кінець s1
strlen(s1)	Повертає довжину рядка s1
strcmp(s1,s2)	Повертає 0, якщо s1 і s2 збігаються, негативне значення, якщо s1 < s2 і позитивне значення, якщо s1 > s2
strchr(s1,ch)	Повертає вказівник на перше входження символу ch в рядок s1
strstr(s1,s2)	Повертає вказівник на перше входження рядка s2 в рядок s1

Графічне представлення двовимірного масиву

Масив a	Стовбець 0	Стовбець 1	Стовбець 2
Рядок 0	a[0][0]	a[0][1]	a[0][2]
Рядок 1	a[1][0]	a[1][1]	a[1][2]
Рядок 2	a[2][0]	a[2][1]	a[2][2]
Рядок 3	a[3][0]	a[3][1]	a[3][2]

Приклад 1

```
#include <stdio.h>
#include <string.h>
int main(void) {
    char s1[80], s2[80];
    printf("Введіть перший рядок:");
    fgets(s1, sizeof(s1), stdin);
    printf("Введіть другий рядок: ");
    fgets(s2, sizeof(s2), stdin);
    printf("Довжина:%zu %zu\n", strlen(s1), strlen(s2));
    if (!strcmp(s1, s2))
        printf("Рядки рівні:\n");
    strcat(s1, s2);
    printf("%s\n", s1);
    strcpy(s1, "Перевірка.\n");
    printf("%s", s1);
}
```

```
    return 0;
}
```

```
Введіть перший рядок:88
Введіть другий рядок: 50
Довжина:3 3
88
50

Перевірка.
```

Приклад 2

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

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

```
int x[100]; //оголошення масиву цілого типу з 100 чисел
```

```
int t;
```

```
for(t=0;t<100;++t) x[t]=t; //присвоєння значення від0 до 99
```

```
for(t=0;t<100;++t) printf("%d\n",x[t]);}
```

```
1      39      76
2      40      77
3      41      78
4      42      79
5      43      80
6      44      81
7      45      82
8      46      83
9      47      84
10     48      85
11     49      86
12     50      87
13     51      88
14     52      89
15     53      90
16     54      91
17     55      92
18     56      93
19     57      94
20     58      95
21     59      96
22     60      97
23     61      98
24     62      99
25     63
26     64
27     65
28     66
29     67
30     68
31     69
32     70
33     71
34     72
35     73
36     74
37     75
```

Приклад 3

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

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

```
int t,i, num[3][4];
```

```

for(t=0;t<3;++t)
for(i=0;i<4;++i)
num[t][i]=(t*4)+i+1;
//вивід на екран
for(t=0;t<3;++t){
for(i=0;i<4;++i)
printf("%3d",num[t][i]);
printf("\n");
}
return 0;
}

```

```

1  2  3  4
5  6  7  8
9 10 11 12

```

Приклад 4

```

#include<stdio.h>
#define MAX 100
#define LEN 80
char text[MAX][LEN];
int main(void) {
    int t, i, j;
    printf("Для виходу введіть пустий рядок.\n");
    for (t = 0; t < MAX; t++) {
        printf("%d: ", t);
        gets(text[t]);
        if (!*text[t]) break; } // вихід при пустому рядку
    for (i = 0; i < t; i++) {
        for (j = 0; text[i][j]; j++)
            putchar(text[i][j]);
        putchar('\n');    }
    return 0;}

```

```

Для виходу введіть пустий рядок.
0: Vadym
1: Brateiko
2: 1357
3:
Vadym
Brateiko
1357

```

Приклад 5

```

#include<stdio.h>
#define SIZE 10
int main() {
    int arr[SIZE];
    int sum = 0;

```

```

// Заповнення масиву
printf("Введіть %d цілих чисел:\n", SIZE);
for (int i = 0; i < SIZE; i++) {
    printf("Елемент %d: ", i + 1);
    scanf("%d", &arr[i]);
    sum += arr[i]; // Додавання поточного елемента до суми
}
// Виведення масиву
printf("Введений масив: ");
for (int i = 0; i < SIZE; i++) {
    printf("%d ", arr[i]);
}
// Виведення суми елементів масиву
printf("\nСума елементів масиву: %d\n", sum);
return 0;}

```

```

Введіть 10 цілих чисел:
Елемент 1: 1
Елемент 2: 2
Елемент 3: 5
Елемент 4: 9
Елемент 5: 2
Елемент 6: 4
Елемент 7: 8
Елемент 8: 7
Елемент 9: 3
Елемент 10: 12
Введений масив: 1 2 5 9 2 4 8 7 3 12
Сума елементів масиву: 53

```

Відповіді на контрольні запитання

1) Дайте визначення поняття масив в мові C:

У мові програмування C масив - це збірник однотипних елементів даних, які розміщуються у послідовності в пам'яті і ідентифікуються за допомогою одного і того ж імені.

2) Назвіть види масивів:

В мові C існують два основних види масивів:

Одновимірні масиви: Масив, що містить елементи одного типу та індексується одним індексом.

Багатовимірні масиви: Масив, що містить елементи одного типу, індексується декількома індексами та має декілька розмірностей.

3) Назвіть перевагу використання багатовимірних масивів:

Використання багатовимірних масивів дозволяє представляти більш складні дані, такі як матриці або об'єкти, які мають багатовимірну структуру, та спрощує роботу з такими даними у програмах.

4) Для чого у масивах використовується матриця:

Матриця - це один з типів багатовимірних масивів у мові C. Вона використовується для зберігання та опрацювання даних у вигляді двовимірного набору елементів, організованих у вигляді рядків та стовпців.

5) Яка загальна форма ініціалізації масиву:

Загальна форма ініціалізації масиву виглядає наступним чином:
тип_даних ім'я_масиву[розмір] = {елемент1, елемент2, ...,

Тип_назва масиву[розмірність];

Висновок: : на цій лабораторній роботі я ознайомився і навчився використовувати одновимірні та багатовимірні масиви у процесі програмування для обробки великої сукупності значень.