

ЛЕКЦІЯ 6

1. ФУНКЦІЇ В МОВІ С. ОГОЛОШЕННЯ ФУНКЦІЇ
2. ОБЛАСТЬ ДІЇ ТА ОБЛАСТЬ ВИДІМОСТІ. ПРАВИЛА ЛОКАЛІЗАЦІЇ
3. ПАРАМЕТРИ ФУНКЦІЙ
4. ПРИКЛАДИ ПРОГРАМ З ФУНКЦІЯМИ

ФУНКЦІЇ В МОВІ C/C++

(Підпрограми).

Функції - це будівельні блоки мови, самостійні програмні одиниці, що спроектовані для вирішення конкретних завдань, звичайно таких, які повторюються декілька разів.

ОГОЛОШЕННЯ ФУНКЦІЇ

тип <ім'я змінної – назва функції> (список параметрів) {тіло функції}.

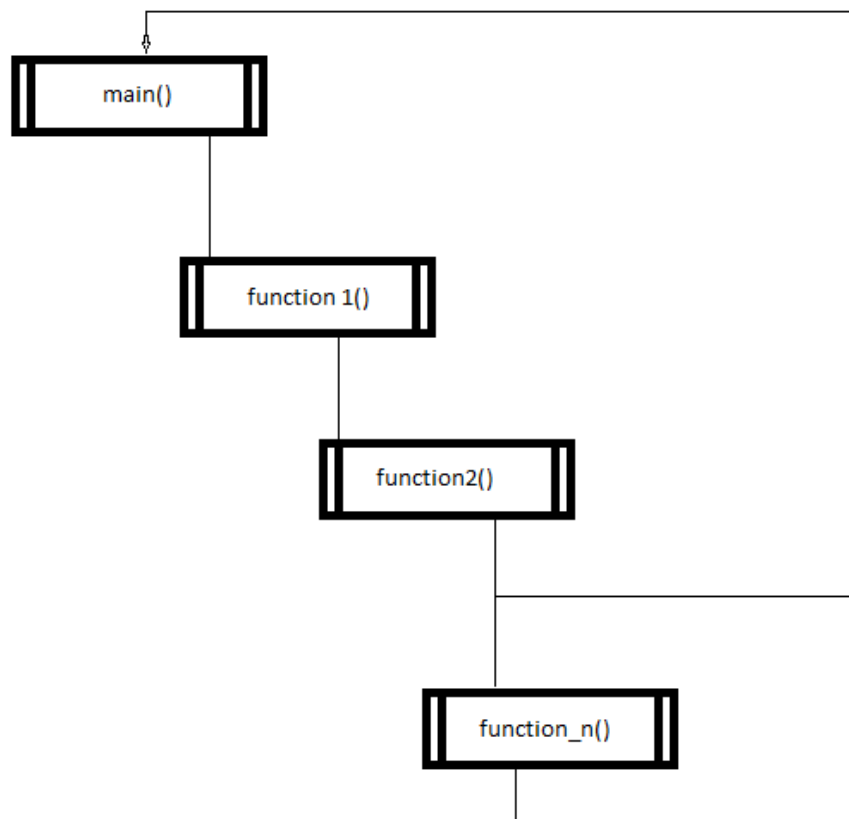
Тип визначає тип значення, що функція повертає. Якщо тип не вказано, то за замовчуванням його вважають *int*.

Список параметрів – перелік змінних з відповідним типом, розділений комами. Функція може не мати параметрів, але дужки () потрібні завжди. Саме вони покажуть, що це є функція. Наприклад

```
int function1(int x, int y, double z) {...
```

Тіло функції (її текст) є окремим блоком програми.

Блок-схема роботи програми, написаної з застосуванням методології структурно-логічного програмування

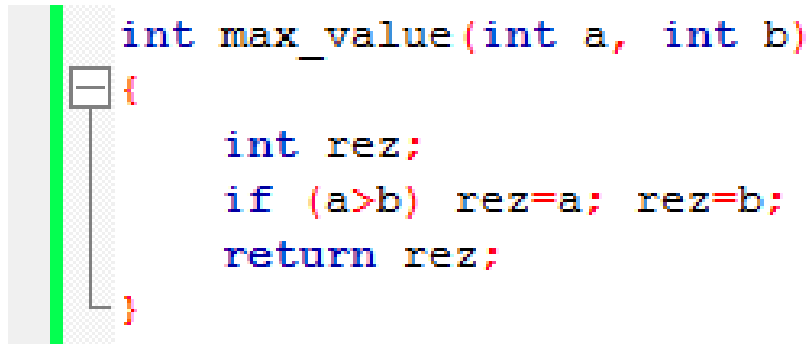


Оператор return

Використовується для повернення управління до тієї функції, яка викликала дану. Має два варіанти використання:

1. Виконує миттєвий вихід з функції та повернення до тієї, з якої вона була викликана
2. Повертає значення скалярної змінної, яке є результатом виконання, до функції, яка її викликала.

Приклад. Функція, що визначає, яке з двох чисел є більшим та повертає його значення.



```
1 int max_value(int a, int b)
{
    int rez;
    if (a>b) rez=a; rez=b;
    return rez;
}
```

Якщо оператор *return* відсутній, жодного значення не повертається. При цьому функцію необхідно описати типом *void*.

Прототипи функцій. Для створення правильного машинного коду програми у неї необхідно розмістити інформацію щодо назви функції, типу результату, який повертається, та переліку параметрів. [Без цього при компіляції буде визначено неоголошений ідентифікатор – ім'я функції]. Для передачі такої інформації використовують поняття прототипу функції

тип <ім'я змінної – назва функції> (список параметрів);

Використання прототипу є оголошенням функції.

Як видно, прототип повторює заголовок функції. Він закінчується ;

Прототип для згаданої функції

```
int max_value(int a, int b);
```

Імена параметрів не обов'язкові. Компілятору потрібна інформація щодо типів параметрів та їхньої кількості. Частіше використовується така форма:

```
int max_value(int, int);
```

ОБЛАСТЬ ДІЇ ТА ОБЛАСТЬ ВИДИМОСТІ

Функція в мові С – це окремий блок програми. Передати управління до нього можливо тільки через оператор виклику даної функції. При цьому при виклику функції є необхідним передати до неї ті значення параметрів, які ця функція має обробити. Ці передані значення називають ФАКТИЧНИМИ ПАРАМЕТРАМИ.

Параметри, які вказані у заголовку функції, називають ФОРМАЛЬНИМИ. Формальні параметри та усі змінні, описані у функції, є ЛОКАЛЬНИМИ ЗМІННИМИ.

(Якщо змінну описано глобально, вона є доступною у кожній функції. Глобальний опис змінних є недоречним та унеможлиблює використання функції у різних програмах, що є одною з цілей структурно-логічного програмування).

ПРАВИЛА ЛОКАЛІЗАЦІЇ

1. Ідентифікатор визначено тільки в межах того блоку, де його оголошено.
2. Один й той самий ідентифікатор може бути оголошений по-різному у різних блоках
3. Якщо у деякому операторі використовується ідентифікатор, оголошений у декількох блоках, то він інтерпретується у відповідності з описом у тому блоці, в якому його використовують.
4. На рівні блоку ідентифікатор може бути оголошеним один раз.

Параметри-значення та параметри-змінні

Інша класифікація параметрів функцій.

Якщо параметр функції не змінюється при її роботі, він називається ПАРАМЕТРОМ-ЗНАЧЕННЯМ.

Якщо параметр функції змінюється при її роботі, він називається ПАРАМЕТРОМ-ЗМІННОЮ.

Для повернення одного значення (параметру-змінної), отриманого у функції шляхом розрахунку, до функції, що її викликала, використовують оператор *return*.

Для передачі декількох значень, визначених при роботі функції, використовують інший механізм, який буде розглянуто пізніше.

ВИКЛИК ФУНКЦІЇ

Оператор виклику функцій складається з імені (назви) функції з переліком біля нього фактичних параметрів, розташованих у дужках.

Існує різниця між викликом типізованих функцій (описаних будь-яким типом, крім void) та, відповідно, нетипізованих (типу void).

Виклик останніх є окремим оператором , що завершується ;

```
function2(перелік фактичних параметрів через кому);
```

Виклик типізованих функцій може розташовуватись у тих місцях, в яких за правилами мови може бути присутня змінна.

```
a= function1(перелік фактичних параметрів через кому);  
cout<< function1(перелік фактичних параметрів через кому);
```

Розглянемо повний текст програми, написаної з використанням функції.

```
1  #include <iostream>
2
3  using namespace std;
4
5  int max_value(int, int); прототип функції
6  int main()
7  { int x=4, y=5;
8    cout<<max_value(x, y);
9    return 0;
10 }
11 int max_value(int a, int b)
12 {
13     int rez;
14     if (a>b) rez=a; rez=b;
15     return rez;
16 }
17
18
```

фактичні параметри

формальні параметри заголовок функції

тіло функції

виклик функції

Цей самий приклад
з двома функціями.

```
main.cpp x
1      #include <iostream>
2
3      using namespace std;
4
5      int max_value(int,int);
6      void show_max_value(int);
7      int main()
8      { int x=4, y=5;
9
10         show_max_value(max_value(x,y));
11         return 0;
12     }
13     int max_value(int a, int b)
14     {
15
16         if (a>b) return a;
17         return b;
18     }
19     void show_max_value(int v)
20     {
21         cout<<v;
22     }
23
```

Приклад 2. За допомогою функції визначити кількість елементів в одновимірному масиві, що є більшими одного та меншими іншого числа. Масив отримується з файлу у функції.

```
1  #include <iostream>
2  #include <fstream>
3  using namespace std;
4
5  int find_number(double dmin, double dmax);
6  int main()
7  {
8      double dmin, dmax;
9      cout<<"Input dmin:";
10     cin>>dmin;
11     cout<<"Input dmax:";
12     cin>>dmax;
13
14     cout<<"Number of elements bigger than "<<dmin<<
15     " and less than "<<dmax<<" is equal to "<<find_number(dmin,dmax);
16
17     return 0;
18 }
```

```
19         int find_number(double dmin1, double dmax1)
20     {
21         /**функція find_number, в якій для введенного з файлу масиву
22         визначається кількість його елементів, що є меншими dmax1 та
23         більшими dmin1, переданих з іншої функції */
24         double arl[6];
25         int k, number=0;
26         ifstream in("ar.txt");
27         for (k=0;k<6;k++)
28             in>>arl[k];
29         in.close();
30
31         for (k=0;k<6;k++)
32             if ((arl[k]>dmin1)&&(arl[k]<dmax1)) number++;
33         return number;
34     }
35
36 }
```

ПРОГРАМУВАННЯ

ar.txt

83 -114 -8 3 1 6

```
Input dmin:1
Input dmax:25
Number of elements bigger than 1 and less than 25 is equal to 2
Process returned 0 (0x0)   execution time : 7.115 s
Press any key to continue.
```

Функції є необхідними будівельними блоками, з яких мають складатись програми, починаючи з середнього ступеню складності.

Вони є основою для побудови методів в об'єктно-орієнтованій складовій мови програмування C++.

ДЯКУЮ ЗА УВАГУ!