КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
імені ТАРАСА ШЕВЧЕНКА

Програмування на мові C++

Лектор: проф. Васильєв О.М.

Лекція №4.

Функції в С++.

Функції

Функція – іменований блок програмного коду, до якого можна звертатись в програмі через ім'я.

Приклад оголошення функції:

```
int factorial(int n){
   int i,s=1;
   for(i=1;i<=n;i++){
      s*=i;
   }
   return s;
}
```

Функція повертає як результат значення виразу після інструкції **return**.

Приклад оголошення функції:

```
void show(){
   cout<<"No result here!";
}
```

Якщо функція не повертає результат, ідентифікатором типу результату вказують ключове слово **void**.

Приклад: сума натуральних чисел

AddNums.cpp

```cpp
#include <iostream>
#include<iostream>
using namespace std;
// функція для розрахунку суми (має аргумент і повертає результат)
int AddNums(int num){
   int s=0;
   for(int i=1;i<=num;i++){
      s+=i;
   }
   return s;
}
// функція для розрахунку суми (без аргументів і не повертає результат)
void AddNumsAndShow(){
   int s=0,n,i;
   cout<<"Hello! We Want to Add Some Numbers."<<endl;;
   cout<<"Please, enter n=";
   cin>>n;
   for(i=1;i<=n;i++){
      s+=i;
   }
   cout<<"1+2+...+"<<n<<"="<<s<<endl;
}
// головна функція програми
int main(){
   int n=10,result;
   result=AddNums(n); // викликаємо функцію
   cout<<"1+2+...+"<<n<<"="<<result<<endl;
   AddNumsAndShow();  // викликаємо функцію
   system("PAUSE");
   return 0;
}
```

Показати результат

Показати програмний код

Механізми передачі аргументів

В C++ існує два механізми передачі аргументів:

- **За значенням –** В функцію передається копія змінної-аргументу (режим за умовчанням)
- **За посиланням –** В функцію передається безпосередньо змінна-аргумент
- Для передачі аргументу за посилання перед іменем аргументу в прототипі функції вказують оператор **&**
- Різницю в механізмах передачі слід враховувати тоді, коли в функції міняються її аргументи

Приклад: механізми передачі аргументів

MakeSwap.cpp

```
#include <iostream>
using namespace std;

//void swap(int &a,int &b){
void swap(int a,int b){
   cout<<"inside the swap-function:\n";
   // аргументи функції "міняються" значеннями
   int c=b;
   b=a;
   a=c;
   cout<<"a= "<<a<<endl;
   cout<<"b= "<<b<<endl;
}
int main(){
    int a=10,b=20;
    // значення змінних
    cout<<"a= "<<a<<endl;
    cout<<"b= "<<b<<endl;
    // змінні "міняються" значеннями
    swap(a,b);
    cout<<"after swap-function:"<<endl;
    // перевіряємо результат
    cout<<"a= "<<a<<endl;
    cout<<"b= "<<b<<endl;
    system("PAUSE");
    return 0;
}
```

Показати результат

Показати програмний код

Приклад: аргумент - вказівник

MakeZero.cpp

```
#include <iostream>
using namespace std;
// фаргумент функції - вказівник
void MakeZero(int *p){
    *p=0;
}
// головна функція програми
int main(){
     int n=100; // змінна
     int *q;    // вказівник
     q=&n;
     // виклик функції з аргументом - вказівником
     MakeZero(q);
     // перевіряємо результат
     cout<<"n="<<n<<endl;
     system("PAUSE");
     return 0;
}
```

Показати результат

Показати програмний код

Приклад: аргумент - масив (одновимірний)

ArrayParameter.cpp

```
#include <cstdlib>
#include <ctime>
#include <iostream>
using namespace std;
// функція для заповнення масиву випадковими числами
void SetRandom(int *p,int n){
   srand(time(NULL)); // ініціалізація генератора
   int i;
   for(i=0;i<n;i++){
      p[i]=rand()%10; // заповнюємо масив
   }
}
// функція для відображення елементів масиву
void ShowArray(int *p,int n){
   for(int i=0;i<n;i++){
      cout<<p[i]<<" ";
   }
   cout<<endl;
}
int main(){
   // розмір масиву
   const int size=15;
   // масив
   int nums[size];
   // масив передається аргументом функціям
   SetRandom(nums,size); // заповнюємо
   ShowArray(nums,size); // відображаємо
   system("PAUSE");
   return 0;
}
```

При передачі **аргументом масиву** насправді треба передавати **два параметри**:

- вказівник на перший елемент масиву та
- розмір масиву.

Показати результат

Показати програмний код

Приклад: аргумент - масив (двовимірний)

Array2DParameter.cpp

```
#include <cstdlib>
#include <ctime>
#include <iostream>
using namespace std;
// глобальна константа
const int size=5;
// функція для заповнення і відображення двовимірного масиву
void SetRandom2D(int nums[][size],int n){
   int i,j;
   // ініціалізація генератора випадкових чисел
   srand(time(NULL));
   for(i=0;i<n;i++){
      for(j=0;j<size;j++){
         nums[i][j]=rand()%2;    // значення елементу масиву
         cout<<nums[i][j]<<" "; // відображення елементу масиву
      }
      cout<<endl;
   }
}
// головна функція програми
int main(){
   const int n=4;      // константа
   int M[n][size];     // двовимірний масив
   SetRandom2D(M,n);   // заповнюємо і відображаємо масив
   system("PAUSE");
   return 0;
}
```

Показати результат

Показати програмний код

Приклад: аргумент - текстовий рядок

ArrayCharsParam.cpp

```
#include <iostream>
using namespace std;
// функція для пошуку символів в тексті
// перший аргумент - вказівник на символьний масив
// другий аргумент - символ для пошуку
// результат - кількість символів в тексті
int FindSymbol(char *txt,char s){
   int k,result=0;
   for(k=0;txt[k];k++){
      if(txt[k]==s){
         result++;
      }
   }
   return result;
}
int main(){
   // символ для пошуку
   char symb='i';
   // текст
   char str[100]="This is very simple text";
   // змінна для запису результату
   int n;
   // викликаємо функцію
   n=FindSymbol(str,symb);
   // відображаємо результат
   cout<<"There are "<<n<<" symbols "<<"\'"<<symb<<"\'";
   cout<<" in the text "<<"\""<<str<<"\"."<<endl;
   system("PAUSE");
   return 0;
}
```

Показати результат

Показати програмний код

Значення аргументів за умовчанням

- Для аргументів функцій можна вказувати **значення за умовчанням**: якщо аргумент явно не переданий функції, використовується значення за умовчанням.

- Аргументи зі значенням за умовчанням в списку аргументів в прототипі функції **вказуються останніми**.

- При оголошенні функції **значення за умовчанням присвоюється аргументу в прототипі функції**.

```
тип ім'я_функції(тип аргумен=значення){
    // код функції
}
```

Приклад: аргумент зі значенням за умовчанням

$z = x + iy$ $i^2 = -1$ $Re(z) = x$ $Im(z) = y$

DefaultParameters.cpp

```
#include <iostream>
using namespace std;
// функція для відображення комплексного числа
void ShowCompNum(double re,double im=0){
    cout<<re<<" + "<<im<<"i"<<endl;
}
int main(){
     // викликаємо функцію з 2-ма аргументами
     ShowCompNum(1,5);
     // викликаємо функцію з одним аргументом
     ShowCompNum(3);
     system("PAUSE");
     return 0;
}
```

Показати результат

Показати програмний код

Перевантаження функцій

● **Перевантаження функції** - створення різних варіантів функції з однаковими назвами але різними прототипами.

● Різні варіанти перевантаженої функції **відрізняються кількістю і типом аргументів**, а також типом результату.

● При виклику функції **варіант** програмного коду для перевантаженої функції **визначається в контексті команди виклику функції** (наприклад, по кількості аргументів).

Приклад: перевантаження функції

Overload.cpp

```
#include <iostream>
using namespace std;
// перевантажуємо функцію
void ShowMeText(){
   cout<<"There are no parameters!"<<endl;
}
void ShowMeText(int a){
   cout<<"There is only one int-parameter: "<<a<<endl;
}
void ShowMeText(int a, int b){
   cout<<"There are two int-parameters: "<<a<<" and "<<b<<endl;
}
void ShowMeText(double a){
   cout<<"There is only one double-parameter: "<<a<<endl;
}
int main(){
   // викликаємо перевантажену функцію
   ShowMeText();     // без аргументів
   ShowMeText(1);    // один int-аргумент
   ShowMeText(2,3); // два int-аргументи
   ShowMeText(4.0); // один double-аргумент
   system("PAUSE");
   return 0;
}
```

Показати результат

Показати програмний код

Рекурсія

Рекурсія - виклик функції в програмному коді, що визначає цю функцію. Рекурсія може бути пряма (безпосередній виклик) і непряма (виклик через інші функції).

```
// Факторіал числа:
int factorial(int n){
   if(n==1){
      return 1;
   }
   else{
      return n*factorial(n-1);
}
```

$$n! = n \cdot (n-1) \ldots \cdot 2 \cdot 1$$

$$n! = n \cdot (n-1)!$$

$$1, 1, 2, 3, 5, 8, 13, 21, 34, 55, \ldots$$

$$a_1 = a_2 = 1$$

$$a_n = a_{n-1} + a_{n-2}$$

```
// Числа Фібоначчі:
int fibs(int n){
   if(n==1||n==2){
      return 1;
   }
   else{
      return fib(n-1)+fib(n-2);
}
```

Приклад: рекурсія

Recursion.cpp

```
#include <iostream>
using namespace std;
// рекурсія
int AddNums(int n){
    if(n<=0){
        return 0;
    }else{ // функція викликає сама себе
        return n+AddNums(n-1);
    }
}
// головна функція програми
int main(){
    int s,n=100;
    // виклик функції з рекурсією
    s=AddNums(n);
    cout<<"1+2+...+"<<n<<" = "<<s<<endl;
    system("PAUSE");
    return 0;
}
```

$$S_n = n + (n-1) + \cdots + 2 + 1$$

$$S_{n-1} = (n-1) + \cdots + 2 + 1$$

$$S_n = n + S_{n-1}$$

Показати результат

Показати програмний код