Programming Fundamental

Lab#08

Table of Contents

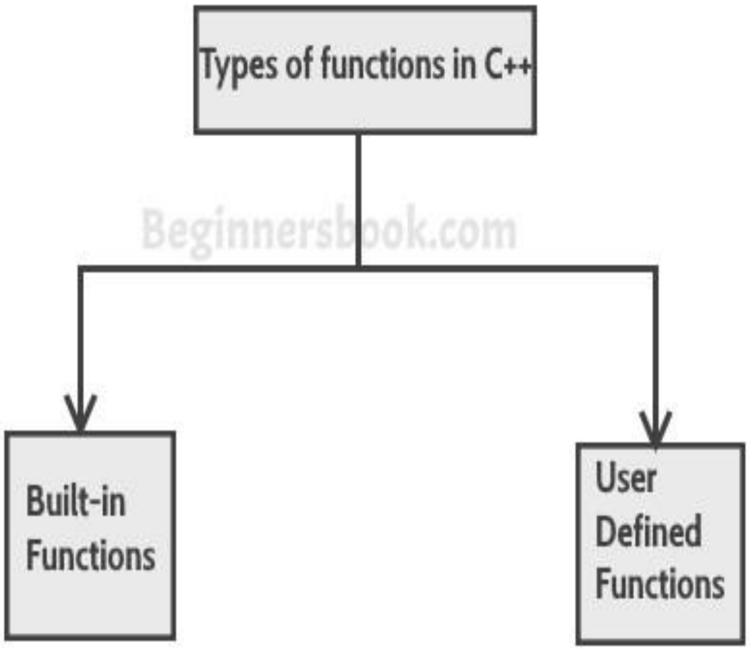
Functions:

A function is a group of statements that together perform a task. Every C++ program has at least one function, which is **main()**, and all the most trivial programs can define additional functions.

You can divide up your code into separate functions. How you divide up your code among different functions is up to you, but logically the division usually is such that each function performs a specific task.

A function is known with various names like a method or a sub-routine or a procedure etc.

Type of Functions



Build-in functions

Built-in functions are also known as library functions. We need not to declare and define these functions as they are already written in the C++ libraries such as iostream, cmath etc. We can directly call them when we need. For example pow(a,x), sqrt(x) etc.

User Define Functions

User define functions are the one that programmer write it by himself.

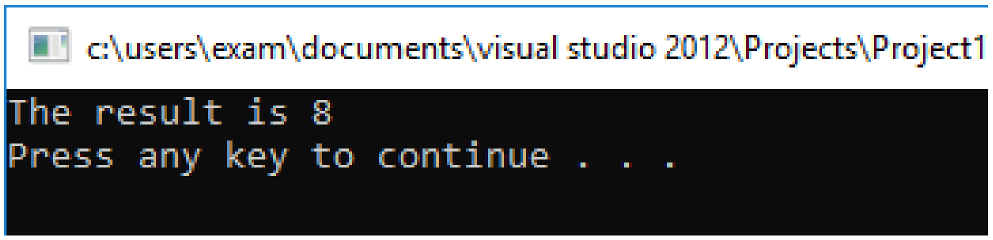
Defining a Function

```
return_type function_name (parameter(s))
{
   //C++ Statements
}
```

A C++ function definition consists of a function header and a function body. Here are all the parts of a function −

- **Return Type** − A function may return a value. The **return_type** is the data type of the value the function returns. Some functions perform the desired operations without returning a value. In this case, the return_type is the keyword **void**.
- **Function Name** − This is the actual name of the function. The function name and the parameter list together constitute the function signature.
- **Parameters** − A parameter is like a placeholder. When a function is invoked, you pass a value to the parameter. This value is referred to as actual parameter or argument. The parameter list refers to the type, order, and number of the parameters of a function. Parameters are optional; that is, a function may contain no parameters.
- **Function Body** − The function body contains a collection of statements that define what the function does.

```
// function example
#include <iostream>
using namespace std;

int addition (int a, int b)
{
  int r;
  r=a+b;
  return r;
}

int main ()
{
  int z;
  z = addition (5,3);
  cout << "The result is " << z<<endl;

  system("pause");
}
```


c:\users\exam\documents\visual studio 2012\Projects\Project1

```
The result is 8
Press any key to continue . . .
```

Function Declarations

A function **declaration** tells the compiler about a function name and how to call the function. The actual body of the function can be defined separately.

A function declaration has the following parts −

```
return_type function_name( parameter list );
```

For the above defined function add(a,b), following is the function declaration −

```
int add(int num1, int num2);
```

Parameter names are not important in function declaration only their type is required, so following is also valid declaration −

```
int add(int, int);
```

Function declaration is required when you define a function in one source file and you call that function in another file. In such case, you should declare the function at the top of the file calling the function.

Calling a Function

While creating a C++ function, you give a definition of what the function has to do. To use a function, you will have to call or invoke that function.

When a program calls a function, program control is transferred to the called function. A called function performs defined task and when it's return statement is executed or when its function-ending closing brace is reached, it returns program control back to the main program.

To call a function, you simply need to pass the required parameters along with function name, and if function returns a value, then you can store returned value. For example

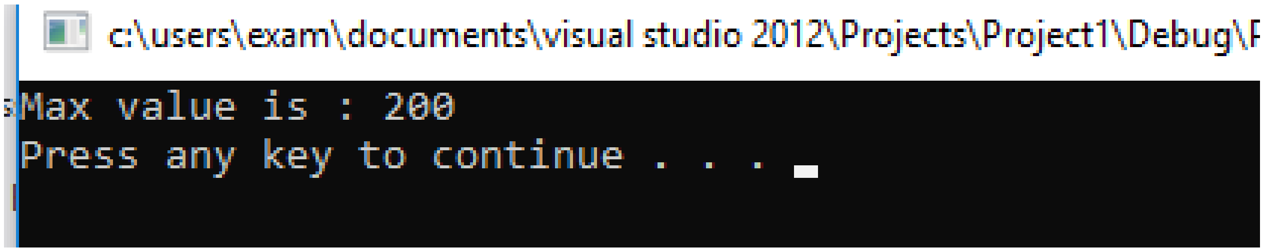
While creating a C++ function, you give a definition of what the function has to do. To use a function, you will have to call or invoke that function.

When a program calls a function, program control is transferred to the called function. A called function performs defined task and when it's return statement is executed or when its function-ending closing brace is reached, it returns program control back to the main program.

To call a function, you simply need to pass the required parameters along with function name, and if function returns a value, then you can store returned value. For example −

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

// function declaration and
int max(int num1, int num2) {
   // local variable declaration
   int result;

   if (num1 > num2)
```

```
      result = num1;
   else
      result = num2;

   return result;
}

int main () {
   // local variable declaration:
   int a = 100;
   int b = 200;
   int ret;

   // calling a function to get max value.
   ret = max(a, b);
   cout << "Max value is : " << ret << endl;
 system("pause");
   return 0;
}
```



c:\users\exam\documents\visual studio 2012\Projects\Project1\Debug\P

```
Max value is : 200
Press any key to continue . . .
```