

Functions Details

Lecture 1

Function

Introduction to functions: A function is **a block of code that performs some operation.**

- A program may need to repeat the same piece of code at various places.
- Repetition of same code may result in a very large program.
- Reason to use function is to remove this repetition and divide program in different parts.

cslearnerr.com

Advantages of Functions

- Easy to code
- Easy to modify
- Easier to maintain
- Reusability
- Less Programming Time
- Easier to understand

cslearnerr.com

Types of Functions

cslearnerr.com

- **Built-in Function:** Function defined as a part of the language and can be used in any program . e.g. main(). These function are grouped into specialized libraries e.g. iostream , math, stdlib.....
- **User-Defined Function:** The user-defined function makes programmer build their own functions. The most important thing behind these functions is the programmer can create applications with reusable code. These functions are written as a part of a program to perform a specific task.

User-Defined Function

cslearnerr.com

Parts of user defined function.

- Function Declaration.
- Function Definition
- Function Calling.

Structure of Function

Function declaration **cslearnerr.com**

- Tells compiler about a function name and how to call function .
- It gives details about the number and datatype of parameters.
- Return type of function.

Calling a Function

- Takes the program control to the called function.

Defining a function / Function body

- Contain all the commands that makeup the function.

Structure of Function

Syntax :

//function declaration

returntype Function-Name(parameter1, parameter 2,)

{ **cslearnerr.com**

Statements (function body) //Function definition

}

int main()

{

//calling a function

Function-Name(); //Function Call

}

Function Types

Function with no argument(input) and no return value(output)

- void function-name (void)

Function with input but no output

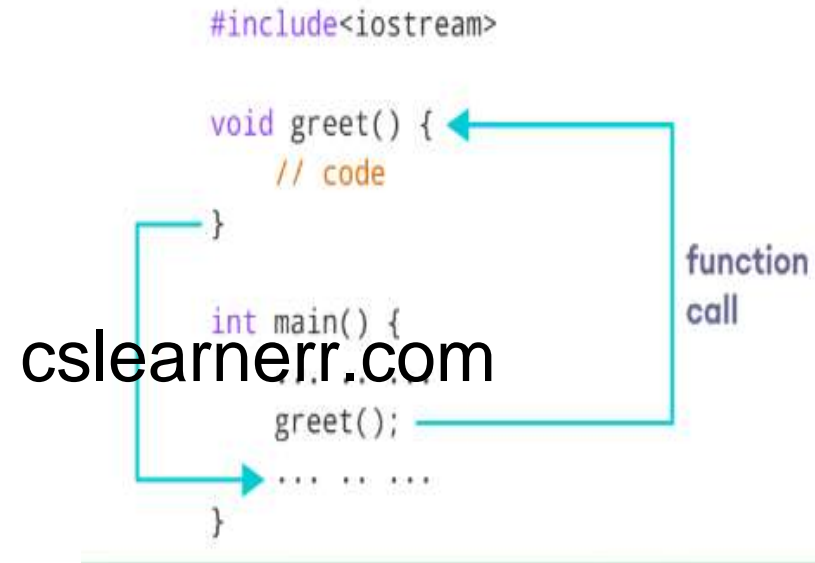
- void function-name (int)

Function with no input but output

- int function-name (void)

Function with both input and output

- int Function-name (int)



Function with no argument(input) and no return Value(output)

PROGRAM CODE:

program to display error message if any other no except 7 is input.

```
func1.cpp
1  #include<iostream>
2  using namespace std;
3  void message(void)
4  {
5      cout<<"wrong entry";
6  }
7  int main()
8  {
9      int x, a=0;
10     while(a<10)
11     {
12         cout<<"\nEnter any no. key: ";
13         cin>>x;
14         if(x!=7)
15             message();
16         a++;
17     }
18     return 0;
19 }
```

OUTPUT:

```
E:\Programming-Lectures\func1.exe
Enter any no. key: 4
wrong entry
Enter any no. key: 3
wrong entry
Enter any no. key: 7
Enter any no. key: 7
Enter any no. key: 7
cslearnerr.com
```

Function with argument(input) and no return value(output)

PROGRAM CODE

```
func1.cpp  func2.cpp
1  //program to find square of two no.
2
3  #include<iostream>
4  using namespace std;
5  void square(int a)
6  {
7      int b;
8      b=a*a;
9      cout<<"\nThe square is= "<<b;
10 }
11
12 int main()
13 {
14     int y=5;
15     cout<<"Finding the square of "<<y;
16     square(y);
17
18     return 0;
19 }
```

OUTPUT

```
Select 1: Programming-Lectures\func2.exe
Finding the square of 5
The square is= 25
-----
Process exited after 0.07992 seconds with return value 0
Press any key to continue . . .
```

cslearner.com

Function with no argument(input) but returnValue(output)

PROGRAM CODE

```
func1.cpp func2.cpp func3.cpp
1 //program to find um of two number.
2 #include<iostream>
3 using namespace std;
4
5 int sum(void)
6 {
7     int a,b,ans;
8     cout<<"\nEnter two integers: \n";
9     cin>>a>>b;
10    ans=a+b;
11    return ans;
12 }
13 int main()
14 {
15     int add;
16     add=sum();
17     cout<<"\nThe sum is = "<<add;
18     return 0;
19 }
20 }
```

OUTPUT

cslearner.com

```
F:\Programming-Lectures\func3.exe
Enter two integers:
2
3

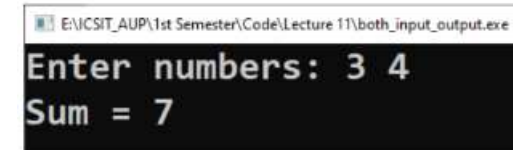
The sum is = 5
-----
Process exited after 2.11 seconds with return value 0
Press any key to continue . . .
```

Function with argument(input) and returnValue(output)

PROGRAM CODE

```
both_input_output.cpp
1 // Functions with both input and output
2 // C++ program to find sum of two numbers
3
4 #include <iostream>
5 using namespace std;
6
7 int sum(int x, int y)
8 {
9     int ans;
10    ans = x + y;
11    return ans;
12 }
13
14 int main()
15 {
16    int num1, num2, add;
17    cout << "Enter numbers: ";
18    cin >> num1 >> num2;
19    add = sum(num1, num2); // function call
20    cout << "Sum = " << add;
21
22    return 0;
23 }
```

OUTPUT



```
E:\VCSIT_AUP\1st Semester\Code\Lecture 11\both_input_output.exe
Enter numbers: 3 4
Sum = 7
```

cslearnerr.com

C++ Function Overloading

In C++, two functions can have the same name if the number and/or type of arguments passed is different.

These functions having the same name but different arguments are known as overloaded functions.

Forexample:

```
int test()
```

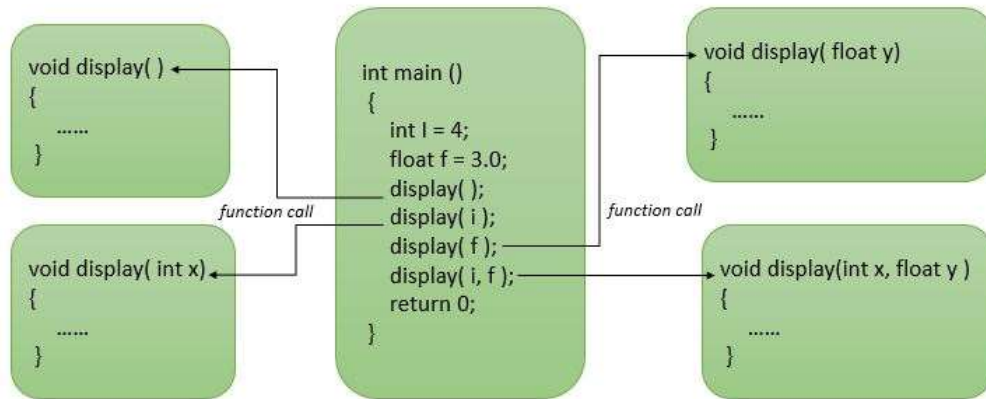
```
int test (int a)
```

```
float test(double a)
```

```
int test(int a, double b)
```

```
//Error code  
int test (int a)  
Float test(int b)  
because both functions have the  
same name, the same type, and  
the same number of arguments.
```

Example



The screenshot shows a C++ IDE with the following code in the editor:

```
15 y = 9.8;  
16 z = 12;  
17  
18 //automatically call  
19 sum(x,y);  
20 /* This will output  
21 Sum using int sum(int  
22 */  
23 sum(x,z);  
24 /* This will output  
25 Sum using int sum(int x, int y): 22  
26 */  
27 sum(x,y,z);  
28 /* This will output  
29 Sum using int sum(int x, int y, int z): 31  
30 */
```

The output window on the right shows the following results:

```
Sum using int sum(int x, float y): 19.8  
Sum using int sum(int x, int y): 22  
Sum using int sum(int x, int y, int z): 31  
-----  
Process exited after 0.1502 seconds with re  
turn value 0  
Press any key to continue . . .
```

Red arrows point from the function calls in the code to the corresponding output lines. The text "cslearner.com" is overlaid at the bottom of the screenshot.

Recursion function

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

int factorial(int n);

int main() {
    int n;

    cout << "Enter a positive integer: ";
    cin >> n;

    cout << "Factorial of " << n << " = " << factorial(n);

    return 0;
}

int factorial(int n) {
    if(n > 1)
        return n * factorial(n - 1);
    else
        return 1;
}
```

A function that calls itself is known as a recursive function. And, this technique is known as recursion.

Example: As shown in example function factorial() is calling itself in its function body.

cslearner.com

Assignment no. 1

- Write a program to print message “This is my first program” on screen using Functions.
- Write a program that takes two parameters x and y as input and returns max of the two numbers.
- Write a function that calculates and returns area of square. Function parameter is length of each side of square.

Discover comprehensive notes for BSCS, BSIT and BSAI courses conveniently gathered on our user-friendly website, www.cslearnerr.com.