

# CL1002 – Programming Fundamentals Lab



## Lab # 03

### Introduction to C Programming

Instructor: Engr. Muhammad Usman

Email: [usman.rafiq@nu.edu.pk](mailto:usman.rafiq@nu.edu.pk)

Department of Computer Science,  
National University of Computer and Emerging Sciences FAST Peshawar

# Programming

Programming is the process of creating a set of instructions that tell a computer how to perform a task.

## Introducing C

C is a general-purpose programming language developed in 1972 by Dennis Ritchie.

C has been used to write everything from operating systems (including Windows and many others) to complex programs like the Python interpreter, Git, Oracle database, and more.

C programming is considered as the base for other programming languages, most of the compilers, Kernels, etc. are written in C language, and most of the programming languages follow C syntax, for example, C++, Java, C#, etc. That is why it is known as mother language.

## GCC

GCC is a Linux-based C compiler released by the Free Software Foundation which is usually operated via the command line. It often comes distributed freely with a Linux installation, so if you are running UNIX or a Linux variant you will probably have it on your system. You can invoke GCC on a source code file simply by typing:-

```
gcc filename
```

The default executable output of GCC is "a.out", which can be run by typing “ ./a.out ”. It is also possible to specify a name for the executable file at the command line by using the syntax “ -o outputfile ”, as shown in the following example :-

```
gcc filename -o outputfile
```

Again, you can run your program with "./outputfile". (The ./ is there to ensure you run the program for the current working directory.)

## Beginning with C programming:

```
#include <stdio.h>
int main()
{
    printf("Hello World");
    return 0;
}
```

## Structure of a C program

We can formally assess the structure of a C program. By structure, it is meant that any program can be written in this structure only. Writing a C program in any other structure will hence lead to a Compilation Error.

The structure of a C program is as follows:

Structure of C Program	
<i>Header</i>	<code>#include &lt;stdio.h&gt;</code>
<i>main()</i>	<code>int main() {</code>
<i>Variable declaration</i>	<code>int a = 10;</code>
<i>Body</i>	<code>printf( "%d ", a );</code>
<i>Return</i>	<code>return 0; }</code>

The components of the above structure are:

**#include <stdio.h>** includes the standard input output library functions. The `printf()` function is defined in `stdio.h`.

**int main()** The `main()` function is the entry point of every program in c language.

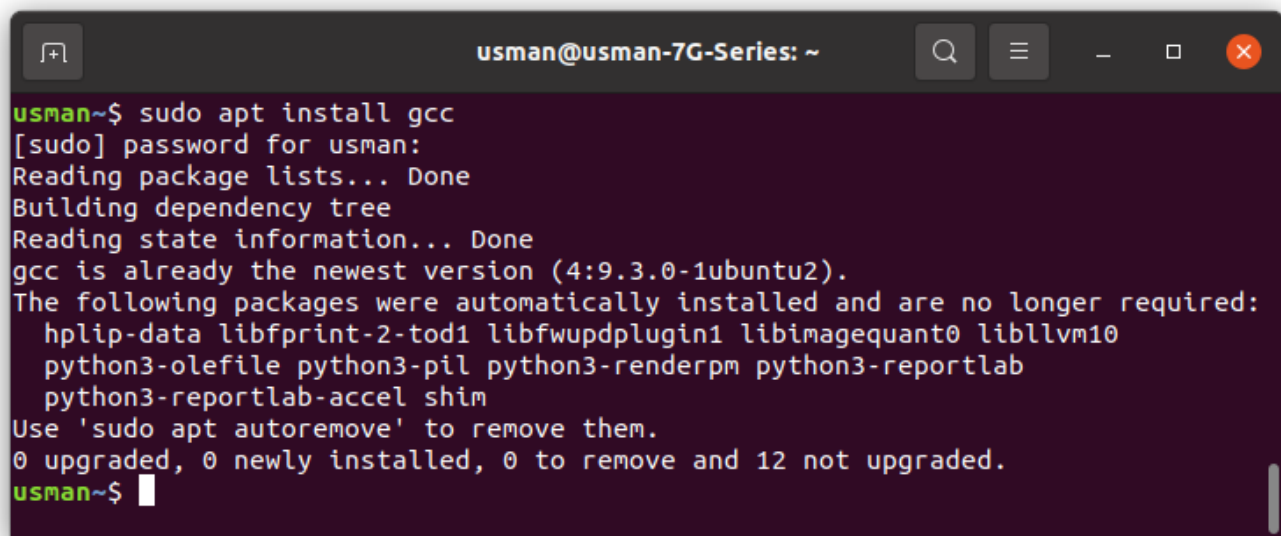
**printf()** The `printf()` function is used to print data on the console.

**return 0** The `return 0` statement, returns execution status to the OS. The 0 value is used for successful execution.

## Task: A Simple First Program

You need to perform the following to complete the task.

1. Open the Terminal (Ctrl + Alt + t)
2. Installation gcc



```
usman@usman-7G-Series: ~  
usman~$ sudo apt install gcc  
[sudo] password for usman:  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
gcc is already the newest version (4:9.3.0-1ubuntu2).  
The following packages were automatically installed and are no longer required:  
  hplip-data libfprint-2-tod1 libfwupdplugin1 libimagequant0 libllvm10  
  python3-olefile python3-pil python3-renderpm python3-reportlab  
  python3-reportlab-accel shim  
Use 'sudo apt autoremove' to remove them.  
0 upgraded, 0 newly installed, 0 to remove and 12 not upgraded.  
usman~$
```

**Note:** gcc is already installed on lab PCs

3. Create file of .c file extension using touch command  
~\$ **touch helloworld.c**
4. Now open the text editor using gedit command  
~\$ **gedit helloworld.c**
5. Write the following code in helloworld.c file.

```
#include <stdio.h>  
int main()  
{  
    printf("Hello World");  
    return 0;  
}
```

6. Save and close the file.

7. compile and execute it

```
~$ gcc helloworld.c -o helloworld.out
```

```
~$ ./helloworld.out
```

## printf() and scanf() in C

The printf() and scanf() functions are used for input and output in C language. Both functions are inbuilt library functions, defined in stdio.h (header file).

### printf() function

The printf() function is used for output. It prints the given statement to the console.

The syntax of printf() function is given below:

```
printf("format string",argument_list);
```

The format string can be %d (integer), %c (character), %s (string), %f (float) etc.

---

### scanf() function

The scanf() function is used for input. It reads the input data from the console.

```
scanf("format string",argument_list);
```

## Variables in C

- A named memory location where data is stored is called variable.
- A quantity whose value may change during execution of the program is called variable. It is represented by a symbol or name.
- Variable is name of reserved area allocated in memory. In other words, it is a name of memory location.
- It is a combination of "vary + able" that means its value can be changed.

```
int data=10 // Here data is variable
```

Let's see the syntax to declare a variable:

```
type variable_list;
```

The example of declaring the variable is given below:

```
int a;  
float b;  
char c;
```

Here, a, b, c are variables. The int, float, char are the data types.

We can also provide values while declaring the variables as given below:

```
int a=10,b=20; //declaring 2 variable of integer type  
float f=20.8;  
char c='A';
```

## Rules for defining variables

- A variable can have alphabets, digits, and underscore.
- A variable name can start with the alphabet, and underscore only. It can't start with a digit.
- No whitespace is allowed within the variable name.
- A variable name must not be any reserved word or keyword, e.g. int, float, etc.

### Valid variable names:

```
int a;  
  
int _ab;  
  
int a30;
```

### Invalid variable names:

```
int 2;
```

```
int a b;
```

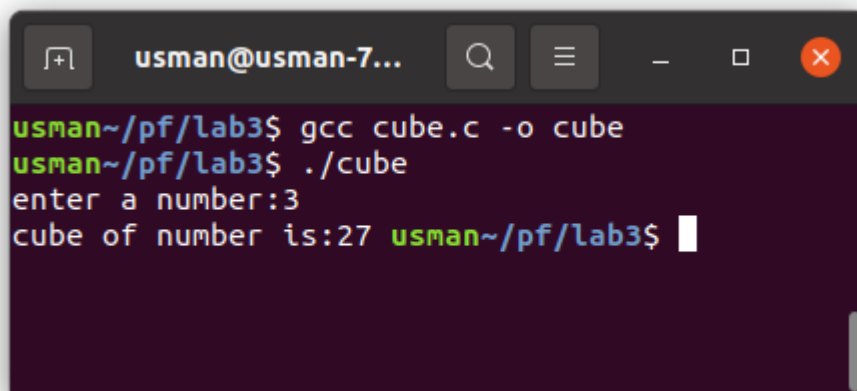
```
int long;
```

## Program to print cube of given number

Let's see a simple example of c language that gets input from the user and prints the cube of the given number.

```
1#include<stdio.h>
2int main(){
3
4  int number;
5  printf("enter a number:");
6  scanf("%d",&number);
7  printf("cube of number is:%d ",number*number*number);
8
9  return 0;
10 }
```

### Output

A terminal window with a dark background and light-colored text. The window title is "usman@usman-7...". The terminal shows the following commands and output:   
1. `usman~/pf/lab3$ gcc cube.c -o cube`   
2. `usman~/pf/lab3$ ./cube`   
3. `enter a number:3`   
4. `cube of number is:27 usman~/pf/lab3$`   
The cursor is at the end of the last line.

```
usman@usman-7...  
usman~/pf/lab3$ gcc cube.c -o cube  
usman~/pf/lab3$ ./cube  
enter a number:3  
cube of number is:27 usman~/pf/lab3$
```

The `scanf("%d",&number)` statement reads integer number from the console and stores the given value in number variable.

The `printf("cube of number is:%d ",number*number*number)` statement prints the cube of number on the console.

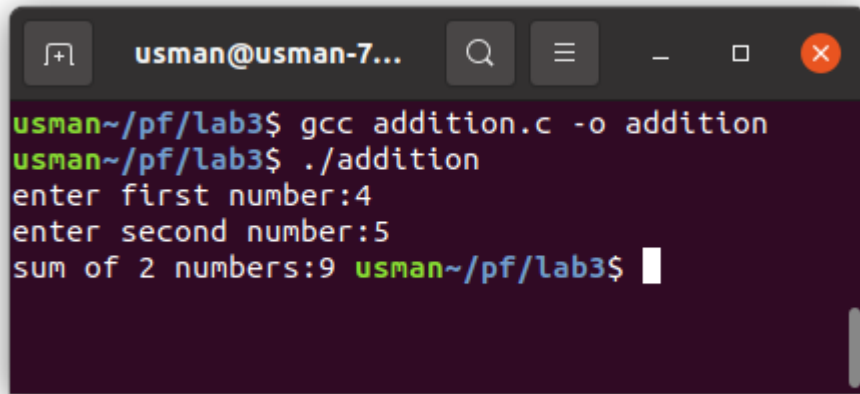
## Program to print sum of 2 numbers

Let's see a simple example of input and output in C language that prints addition of 2 numbers.

```
1 #include<stdio.h>
2 int main(){
3
4 int x=0,y=0,result=0;
5
6 printf("enter first number:");
7 scanf("%d",&x);
8 printf("enter second number:");
9 scanf("%d",&y);
10
11 result=x+y;
12 printf("sum of 2 numbers:%d ",result);
13
14 return 0;
15 }
```

**Output**



A terminal window with a dark background and light-colored text. The window title is "usman@usman-7...". The prompt is "usman~/pf/lab3\$". The user enters "gcc addition.c -o addition", followed by "./addition". The program prompts "enter first number:" and the user enters "4". Then it prompts "enter second number:" and the user enters "5". Finally, it outputs "sum of 2 numbers:9" and returns to the prompt "usman~/pf/lab3\$".

```
usman~/pf/lab3$ gcc addition.c -o addition
usman~/pf/lab3$ ./addition
enter first number:4
enter second number:5
sum of 2 numbers:9 usman~/pf/lab3$
```

## Exercises:

1. Write a C program to find the difference between the two numbers.
2. Write a C program to find the sum and average of three numbers.

## References

<https://www.geeksforgeeks.org/c-language-set-1-introduction/>

<https://www.javatpoint.com/first-c-program>