./

Learning Report – C-PROGRAMMING



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Ver. Rel. No.** | **Release Date** | **Prepared. By** | **Reviewed By** | **To be approved By** | **Remarks/Revision Details** |
| 1 |  | Name/PS No | Name/PS No | Module Owner Name | Comments |
| 2 | 20/02/21 | 99003779 |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**Document History**

Table of Contents

ACTIVITY 1……………………………………………………………………………………………………………………………………………………………………4

REVISION OF BASIC OF C PROGRAMMING……………..…………………………………………………………………………………………………4

LEARNING CERIFICATE ON SOLOLEARN…………………………………………………………………………………………………………………….7

ACTIVITY 2……….…………………………………………………………………………………………………………………………………………………………8

PROGRAM\_1…………………………………………………………………………………………………………………………………………………………..8

PROGRAM\_2………………………………………………………………………………………………………………………….………………………………..9

GITHUB……………………………………………………………………………………………………………………………………….…………………………10

[Table of Figures 3](#_Toc64289204)

[Table of Tables](#_Toc64289205) 3

## Table of Figures

[Figure 1 C Programming Sololearn Certificate](#_Toc52177314) 7

[Figure 2 Git Repository](#_Toc52177315) 10

## Table of Tables

[Table 1 Basic Commands](#_Toc52177304) 4

[Table 2 Section Description](#_Toc52177305) 6

**ACTIVITY 1**

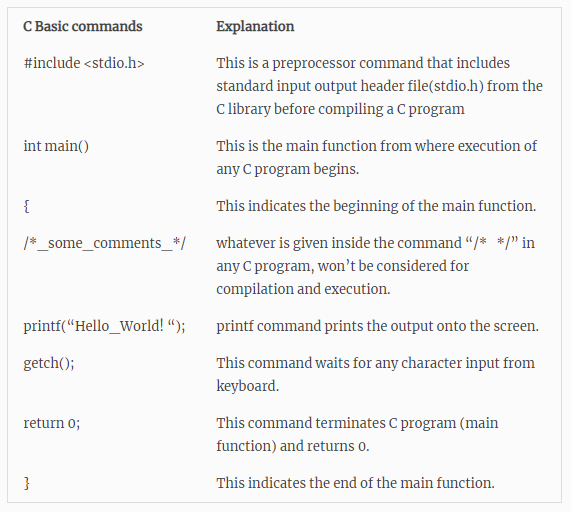
**LEARNING OF C-PROGRAMMING**

**Basics of C Programming :**

C language is a general-purpose programming language. It is used to develop any software like operating systems, compilers, databases etc. C is a low level programming language, best for beginners

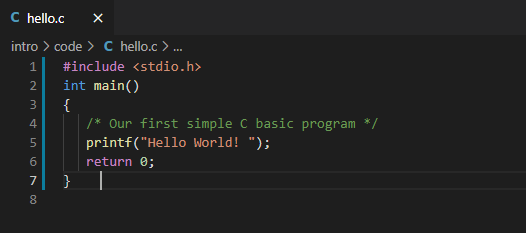
Below is a list of syntax and few command used in C programming to write simple c programs:

**Table 1: C Basic commands**

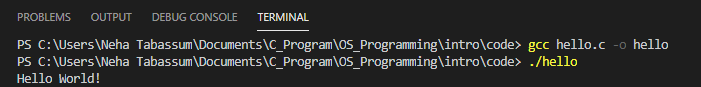


**C program for beginner:**

The following code shows the Hello world program which displays “Hello World” in the terminal . C programming is a case sensitive language. Semicolon is required to end each statement and is known as statement terminator.



Output:



**STEPS FOR WRITING THE C PROGRAM AND GETTING THE OUTPUT:**

**Following are the steps required to create and get the output of any C program. The steps are common to all the programming language with no exception. No matter whether the code is large or small.**

**Write the Code**

**Compile**

**Run and Execute**

**Generate the output**

# ****CREATION, COMPILATION AND EXECUTION OF A C PROGRAM:****

1. Prerequisite: In order to create, compile and execute the C program, one should have a compiler. Here we used GCC compiler on Visual Studio code platform.
2. Once the compiler is installed, the written is ready to execute and run.

**BASIC STRUCTURE OF A PROGRAM**

Protocols are the set of rules which is used to define the structure of C program. These protocols are used by the programmers while writing the C program. All programs have different sections/parts which are described below:

User defined function definition section

Documentation Section

Link Section

Definition section

Main Function

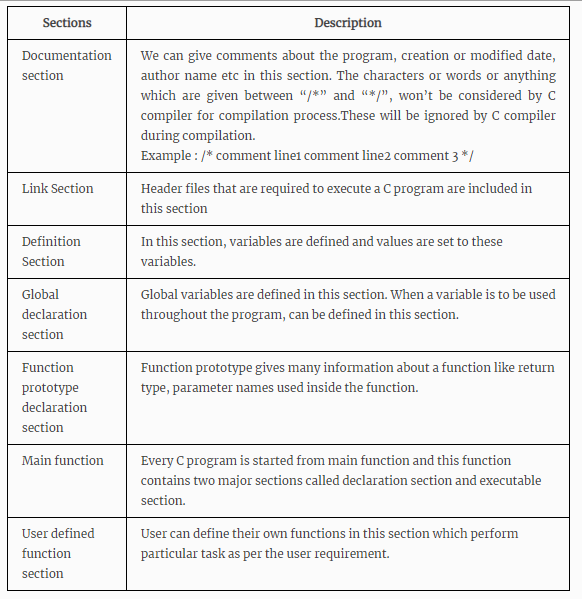
Global Declaration section

Function prototype declaration section

Main function

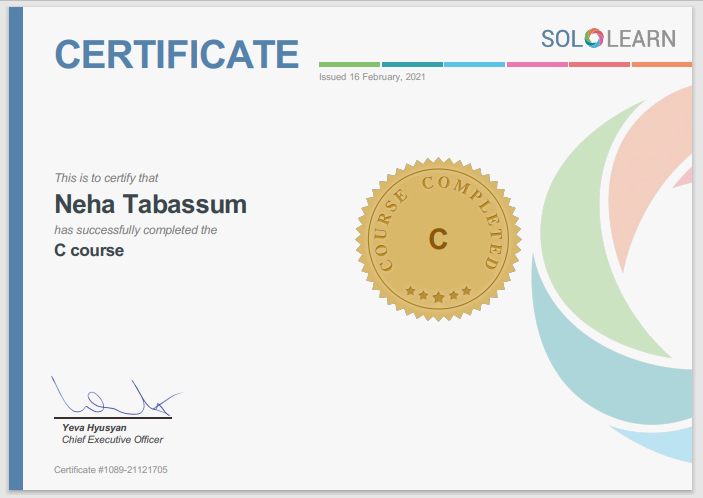
Description are shown below in the table:

**Table 2: Section Description**



**Learning Certificate on SOLOLEARN:**

Following is the certificate on sololearn:



**Figure 1: C Program Learning Certificate**

**ACTIVITY 2**

**C-PROGRAMS**

**Program 1:- Write a program to break the given string and return the correct string between ‘$’ and ‘;’**

**Ex 1: Input “Garbage$Welcome to LTTS;ignore”**

**Output: “&Welcome to LTTS;”**

**Ex2: Input “Hello$This is $$an example #2string;And filled with words”**

**Output: “$an example #2string;”**

**Code:**

#include<stdio.h>

#include<stdlib.h>

char\* splitString(char inputStr[1000]) {

    int i, dollarStart = 0, semiColon = 0;

    for(i = 0; inputStr[i]!='\0'; i++) {

        if(inputStr[i]=='$') {

            dollarStart = i;

        }

        if(inputStr[i]==';') {

            semiColon = i;

            break;

        }

    }

    char \*outputStr = malloc(1001);

    int j = 0;

    for(i = dollarStart; i <= semiColon; i++) {

        outputStr[j] = inputStr[i];

        j++;

    }

    outputStr[semiColon] = '\0';

    return outputStr;

}

int main()

{

    char inputStr[1000];

    gets(inputStr);

    printf("%s\n", splitString(inputStr));

    char line[256];

    {

        printf("%s\n", splitString(line));

    }

    return 0;

}

**Program 2: In program 1 pass the input string after reading from the file.**

#include<stdio.h>

#include<stdlib.h>

char\* splitString(char inputStr[1000]) {

    int i, dollarStart = 0, semiColon = 0;

    for(i = 0; inputStr[i]!='\0'; i++) {

        if(inputStr[i]=='$') {

            dollarStart = i;

        }

        if(inputStr[i]==';') {

            semiColon = i;

            break;

        }

    }

    char \*outputStr = malloc(1001);

    int j = 0;

    for(i = dollarStart+1; i <= semiColon; i++) {

        outputStr[j] = inputStr[i];

        j++;

    }

    outputStr[semiColon] = '\0';

    return outputStr;

}

int main()

{

    char inputStr[1000];

    FILE \*fp = fopen("input.txt", "r");

    if(fp == NULL) {

        printf("Unable to open file!");

        return 0;

    }

    char line[256];

    while (fgets(line, sizeof(line), fp)) {

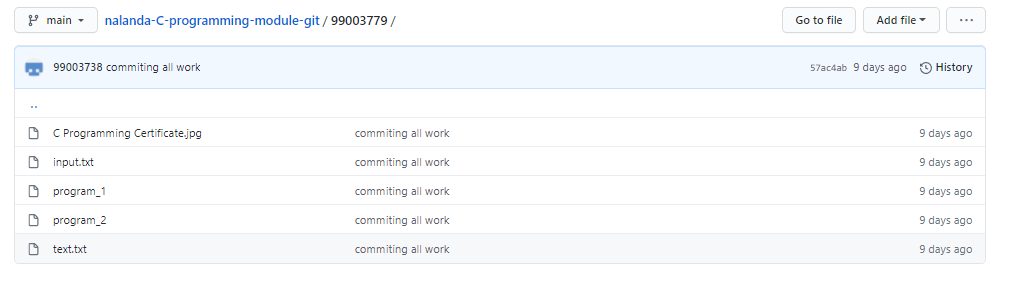
        printf("%s\n", splitString(line));

    }

    return 0;

}

**GitHub :**



**Figure 2: Git Repository**