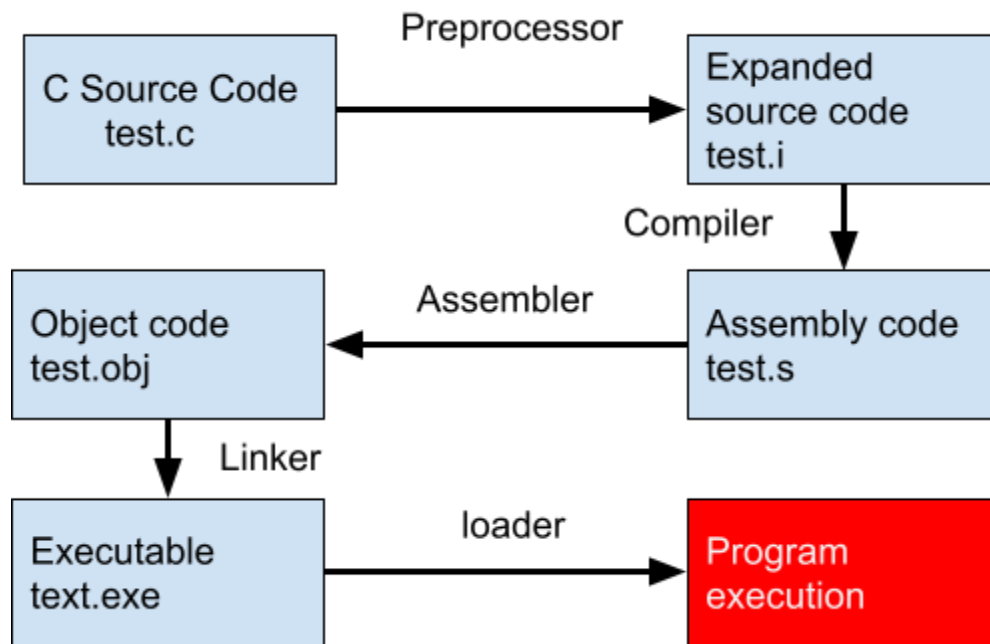


Lab 2

Overall flow of a program:



1) Writing a C program.

2) Compile a C program.

3) Executing a C program.

1) Writing a C program.

Structure of a C program:

Documentation Section // Begin with the comment about the file contents.

Preprocessor Directives // Insert #include statements and preprocessor definitions.

Global Declaration Section // Function prototypes and variable declarations.

Main () function section // Define the main function

{

Lab 2

Declarations : local variables to function main;

Executable statements : statements associated with function main;

// Body of the function

}

f1()

{

Declarations : local variables to function 1;

Executable statements : statements associated with function 1;

}

So on

fn()

{

Declarations : local variables to function n;

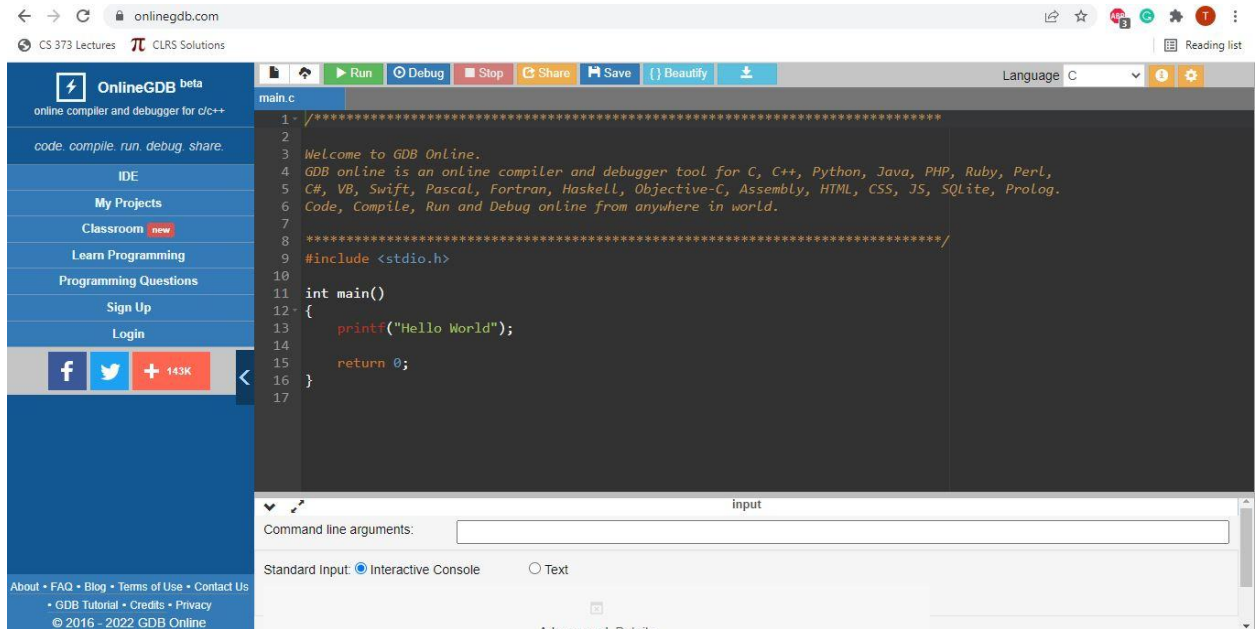
Executable statements : statements associated with function n;

}

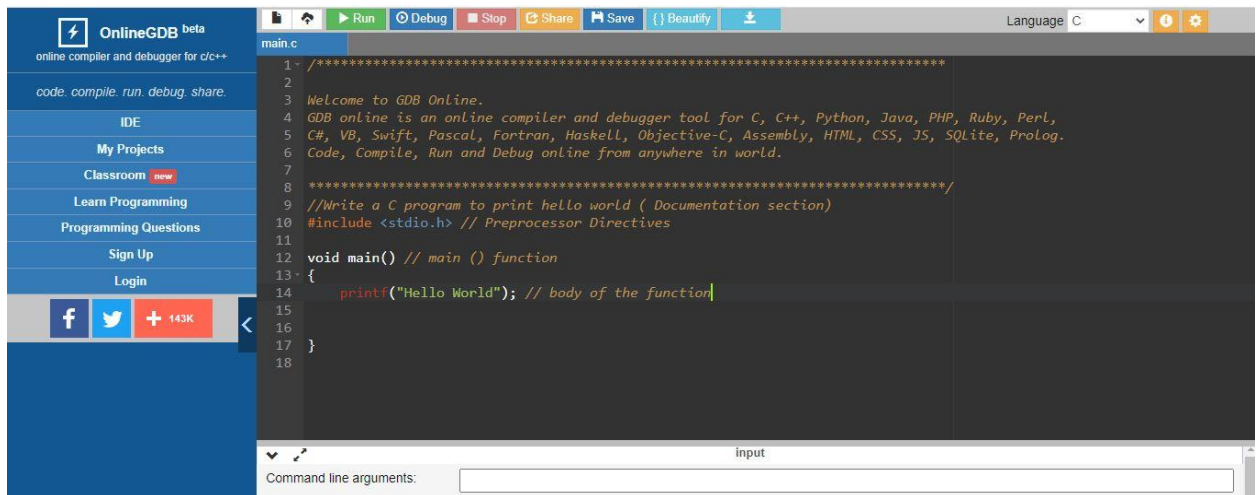
2) Compile and Execute a C program. (Using online compiler gdb)

Step 1: Click on <https://www.onlinegdb.com/> and select the c language option

Lab 2

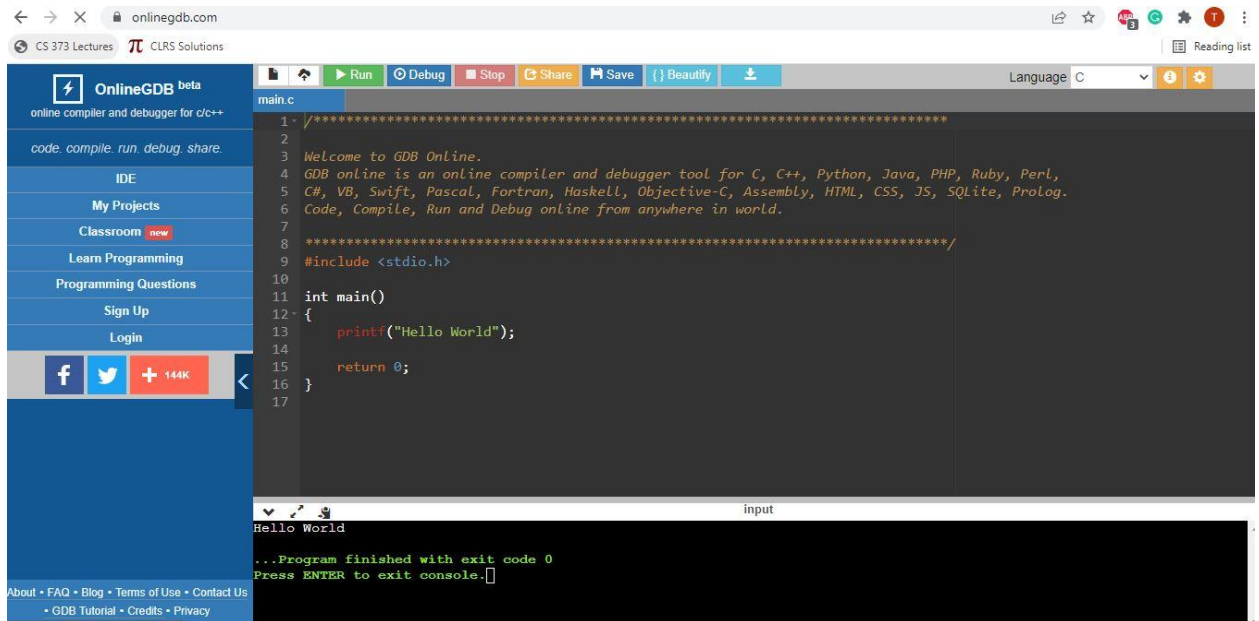


Step 2 : Understand the different sections mentioned above while writing C program.



Step 3: Click on the run button to execute your program. The output is displayed on the screen.

Lab 2



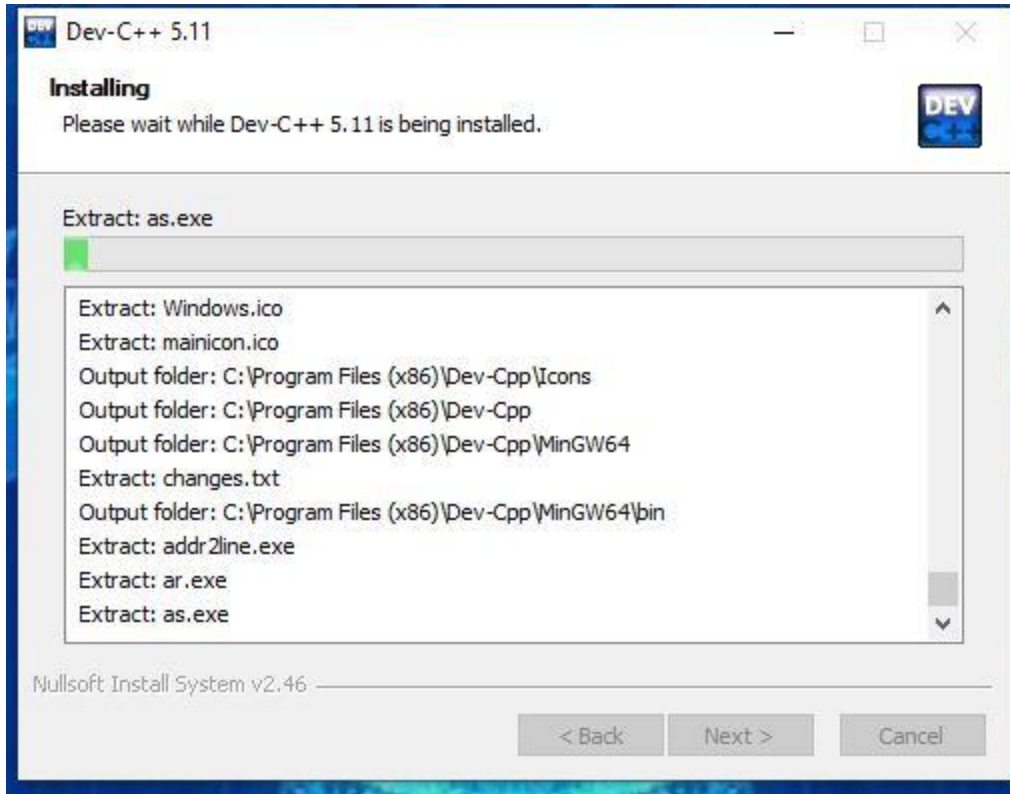
2) Compile a C program (Using Dev C++ IDE)

Step 1 : Click on the link <https://sourceforge.net/projects/orwelldvcpp/> and download Dev C++ IDE .

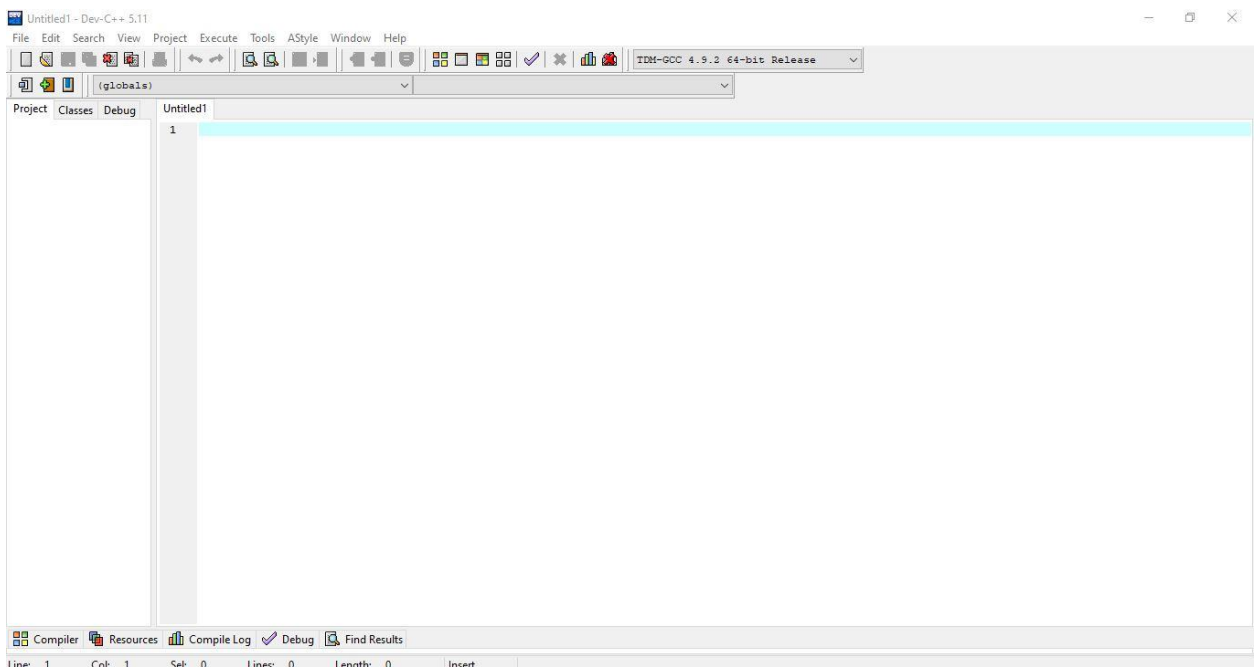


Step 2 : After downloading, click on setup and start installation. Click next, next and finish the installation.

Lab 2

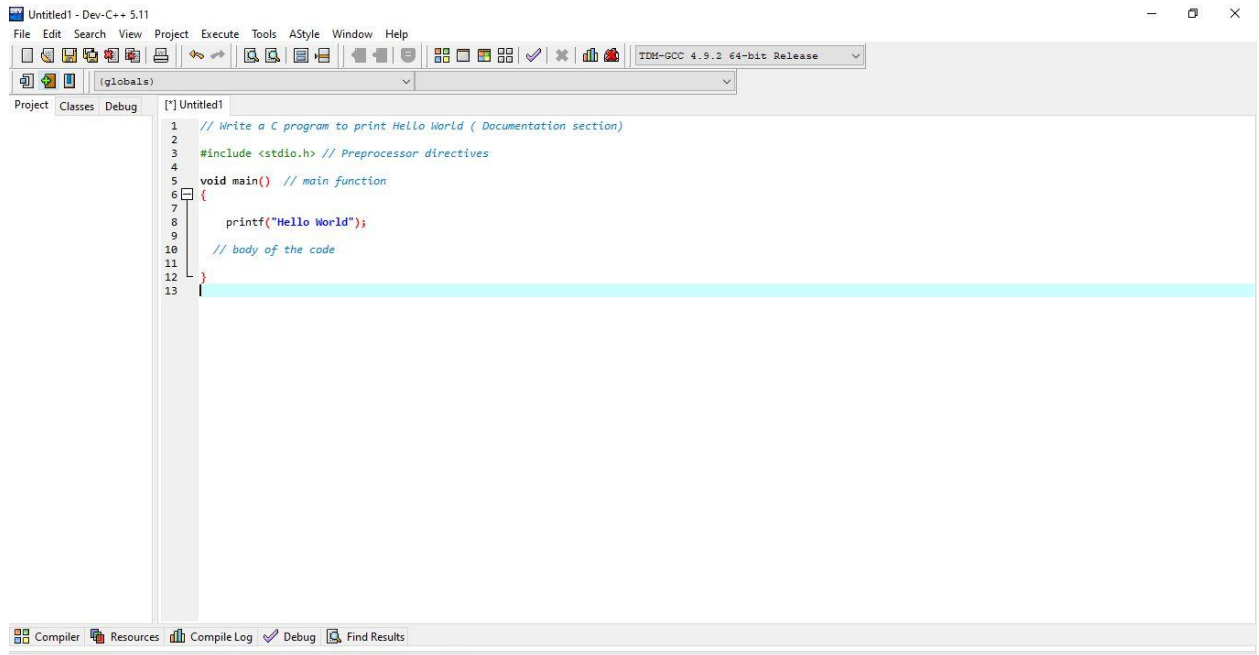


Step 3 : Open Dev c++ icon on your desktop. Open File -> New ->Source file

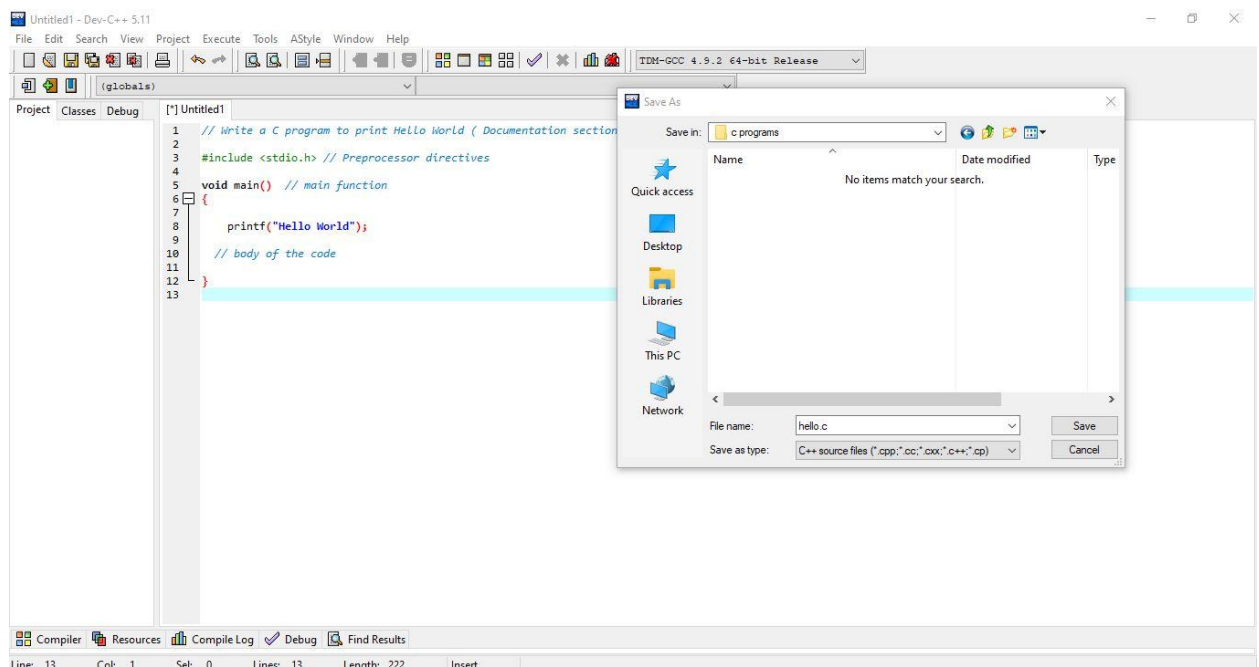


Step 4 : Write the C program in the source file.

Lab 2

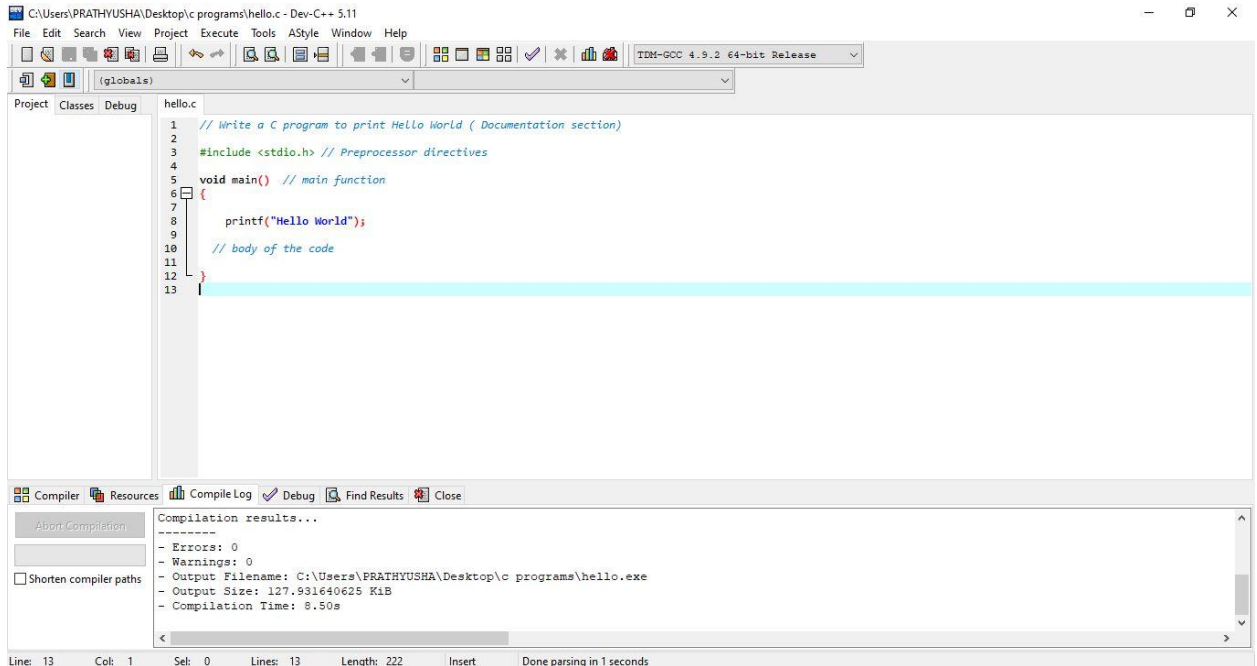


Step 5 : Save the C program using the .c extension. Click on File -> Save as -> Create one folder on desktop and name it as C programs and select the folder -> name the file as filename.c -> click save button.

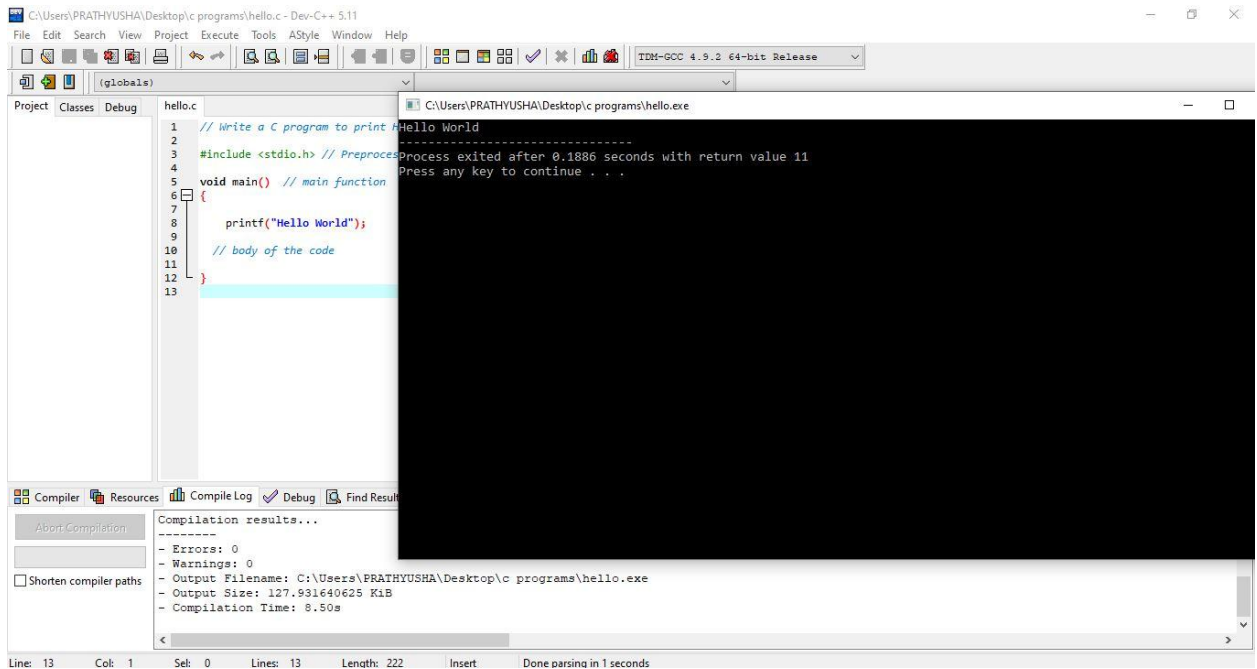


Lab 2

Step 6 : Compile the program. Click on Execute -> Compile (F9 is shortcut key). You can see the below screenshot showing Errors as zero and warnings and Zero. Indicates your program is ready to execute.



Step 7 : Executing the program. Click on Execute -> Run (F10 is the shortcut key). See the below Screenshot displaying the output.



Lab 2

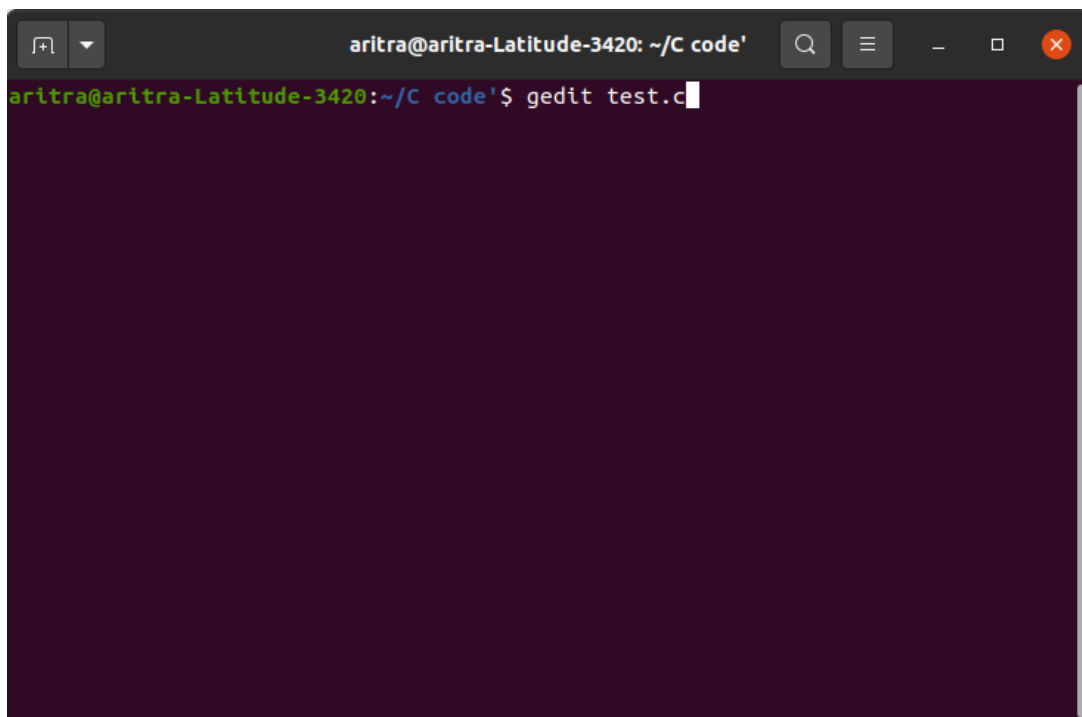
Compiling C code with Linux, gcc compiler (Ubuntu screenshots shown)

The GCC (GNU Compiler Collection) comes bundled with linux distribution so one need not install anything extra. Just an update to the system will configure it.

One can use an IDE like VS Code to compile and run a C code but better control is there if it is compiled manually as many other flags can be included as required. A detailed description of the various flags are beyond the scope of this manual and for compiling complicated sets of code one needs to use a cmake so that step by step compiling and building can be executed in an automated manner.

The most primary and common flag one needs to know is the -o whose job is to rename the output file from the default a.out

First open a terminal in any folder by right click and “open in terminal” or you can just open a terminal by ctrl + alt + t and then ‘cd’ to the folder where you want to create the code. In Ubuntu you have the gedit text editor which can be used for writing code (Though Vi editor is barebones and more powerful, it is time consuming to learn and not necessary in 99% of scenarios). Just give the following command:

A screenshot of a terminal window. The window title bar shows 'aritra@aritra-Latitude-3420: ~/C code' with search, menu, and window control icons. The terminal text shows the prompt 'aritra@aritra-Latitude-3420:~/C code'\$ followed by the command 'gedit test.c' being entered, with a cursor at the end of the command. The terminal background is dark purple.

Lab 2

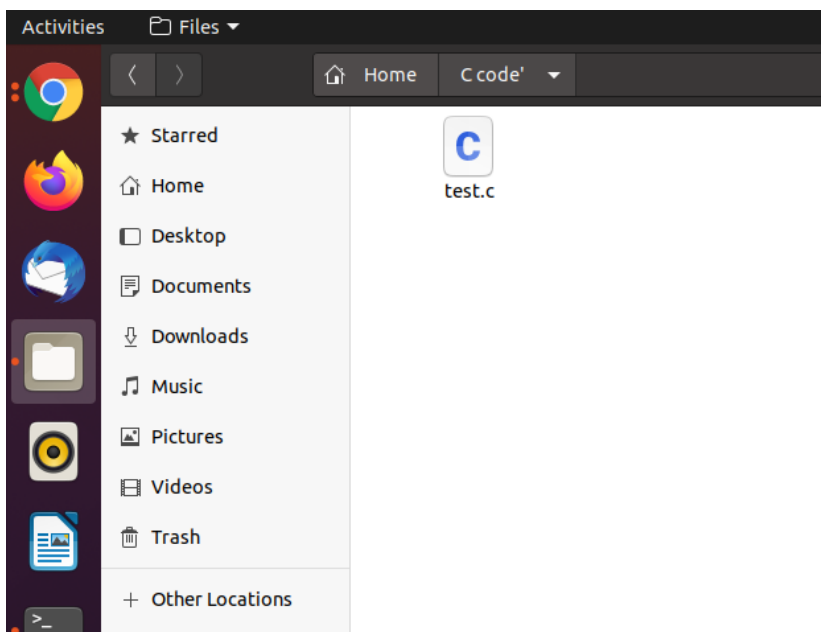
After that press enter. A window will open up where you can write the code. Write a simple code and save it by ctrl + s or simply the GUI option in the file on the menu above.



```
1 #include <stdio.h>
2
3 int main(){
4     printf("Hello World!");
5 }
6
```

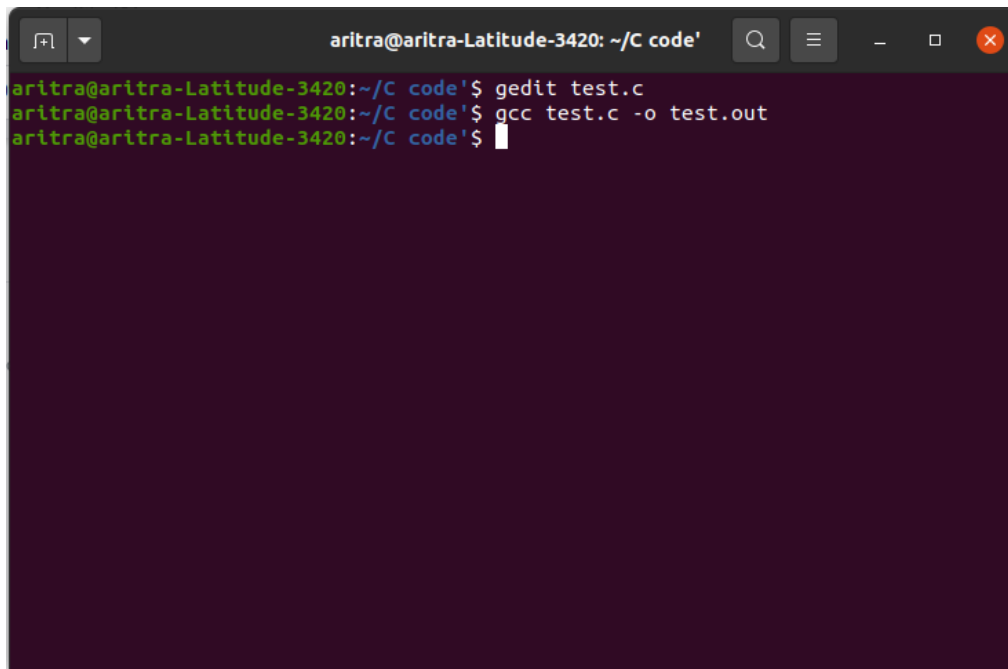
The screenshot shows a code editor window with a dark theme. The title bar indicates the file is named '*test.c' and is located in the directory '~/C code'. The code is written in C and prints "Hello World!". The status bar at the bottom shows the current cursor position as 'Ln 6, Col 1' and the input mode as 'INS'.

Now close the window. You can see a file is saved with the name in the folder



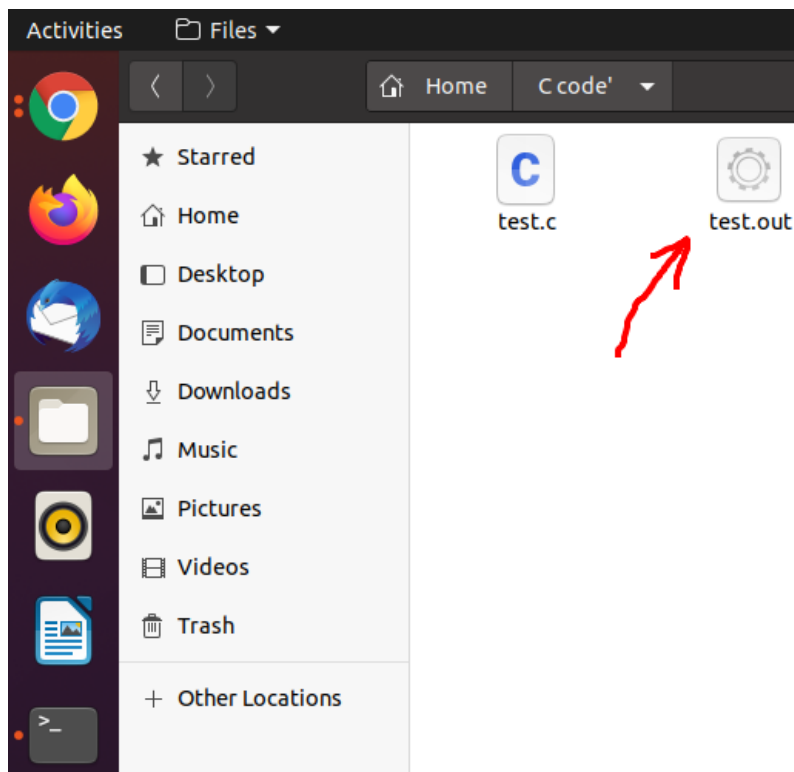
Lab 2

Now go back to the terminal and compile it with the command `gcc <filename.c> <-o outputname.out>`



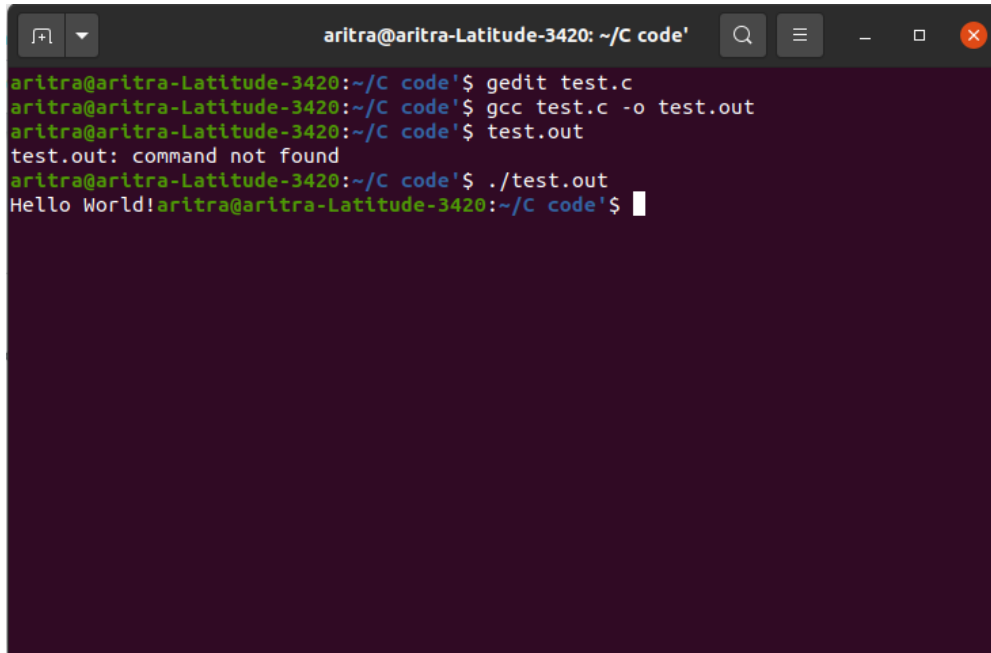
```
aritra@aritra-Latitude-3420: ~/C code'$ gedit test.c
aritra@aritra-Latitude-3420:~/C code'$ gcc test.c -o test.out
aritra@aritra-Latitude-3420:~/C code'$
```

You will observe one new file being added to the directory/folder.



Lab 2

You can see the file name is the same as given after the flag -o. You can give other names too. Now time to execute! Go the terminal again and just type `./<outputname.out>` and press enter to run the code. In some linux distributions the `./` prefix is not necessary if your terminal is open in the same directory but not in ubuntu, as you can see below.

A terminal window titled 'aritra@aritra-Latitude-3420: ~/C code' with standard window controls. The terminal shows the following commands and output:

```
aritra@aritra-Latitude-3420:~/C code'$ gedit test.c
aritra@aritra-Latitude-3420:~/C code'$ gcc test.c -o test.out
aritra@aritra-Latitude-3420:~/C code'$ test.out
test.out: command not found
aritra@aritra-Latitude-3420:~/C code'$ ./test.out
Hello World!aritra@aritra-Latitude-3420:~/C code'$
```

As there was no `\n` or carriage return so after printing the “hello world” there was no newline.

In this way you can compile c codes in linux. For a manual regarding gcc compilers see: <https://man7.org/linux/man-pages/man1/gcc.1.html> (warning: only for the geeks!)