**Programming Tools - Compilation and linking Guided Assignments**

**Getting started with gcc**

1. Log into the Linux server using your respective ids

A vi filename.c

2. Copy the C source file simple\_program.c. Compile the file with the following command:

gcc sourcefilename.c

Run : gcc filename.c

The a.out file will be executed

3. Observe the executable formed is called a.out. Execute the file using the command:

./a.out

Run the ./a.out

A screenshot of a computer screen

Description automatically generated

4. Once again compile the source code with the following command:

gcc –o outputfilename sourcefilename.c

A screen shot of a computer

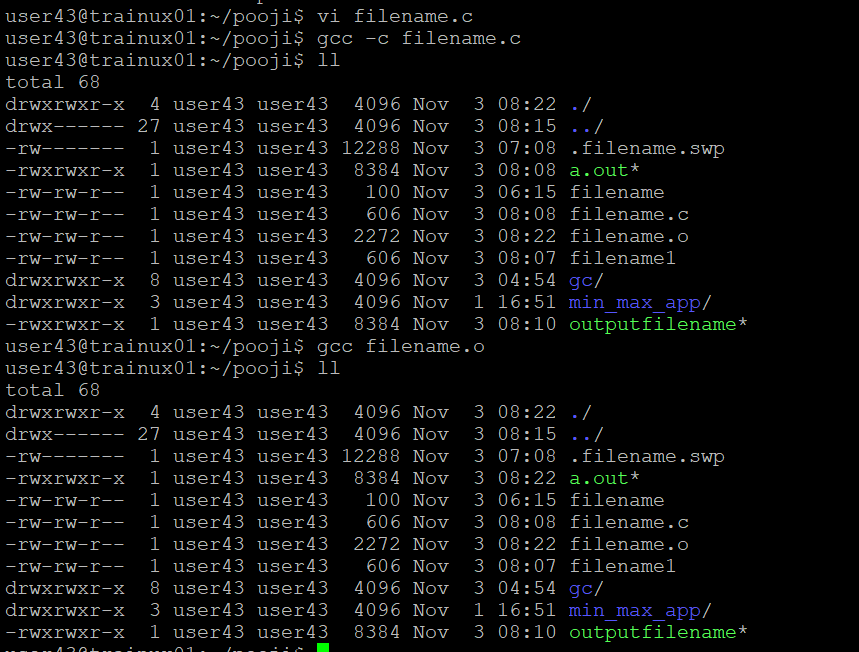
Description automatically generated

**Object files**

5. Once again compile the source code with the following command:

gcc –c sourcefilename.c

Observe using ls that an object file called sourcefilename.o is created in the directory

What is the difference between an executable and an object file? 

**Additional gcc switches**

6. Copy the program program\_warn.c as directed by the facilitator. Compile the program using gcc without any switch. Observe the result of compilation

A screen shot of a computer program

Description automatically generated

7. Now, compile the program with the –Wall switch. Observe the warnings that are now given by the compiler.

A screen shot of a computer program

Description automatically generated

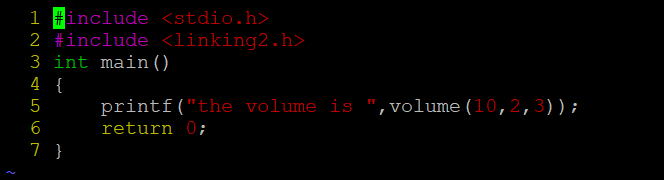
8. Now, compile the program with the –ansi switch. Observe the warnings that are now given by the compiler.

A black screen with white text

Description automatically generated

**Linking**

9. Write a program linking1.c, which has main function. Inside main there should be a call to another function which is defined in another file linking2.c



A computer screen with colorful text

Description automatically generated

10. Compile the file linking1.c using the following command:

gcc –c –Wall linking1.c

Observe the warnings received

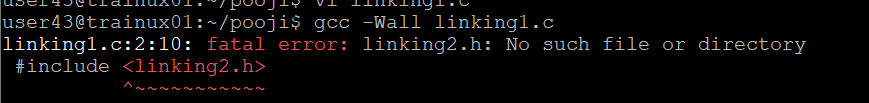
A screen shot of a computer screen

Description automatically generated

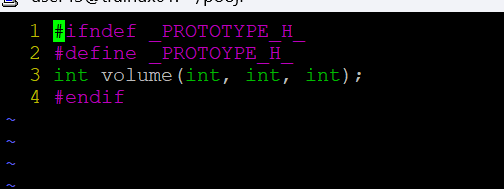
Again compile the file linking1.c using the following command:

gcc –Wall linking1.c

Observe the errors received



11. Now create a header file prototype.h, which contains the prototype of the function defined in linking2.c. Let the file content begin and end with lines as below, to avoid multiple inclusion of .h file.

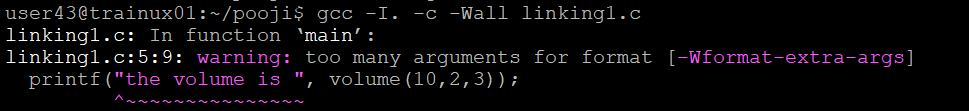


12. Include this file in linking1.c using the following statement:

#include <prototype.h>

and again compile linking1.c using the following command:

gcc –I. –c –Wall linking1.c



13. Similar effect can be achieved by including the file prototype.h using the following

command

#include “prototype.h”

and then compiling using the following command

gcc –c –Wall linking1.c

A black screen with white text

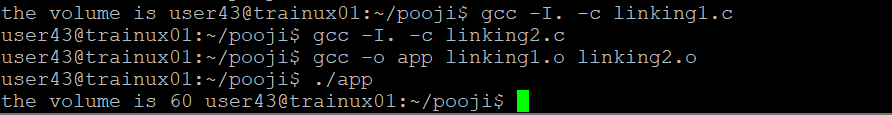
Description automatically generated

14. Now after all the compilation warnings have been removed, link the 2 files together:

gcc –c –Wall linking1.c

gcc –c –Wall linking2.c

gcc linking1.o linking2.o



15. Do not include the header file prototype.h containing the prototype of the function in the file linking1.c.

Instead of that include the file linking2.c in the file linking1.c using the following statement:

#include “linking2.c”

Now repeat the following three commands:

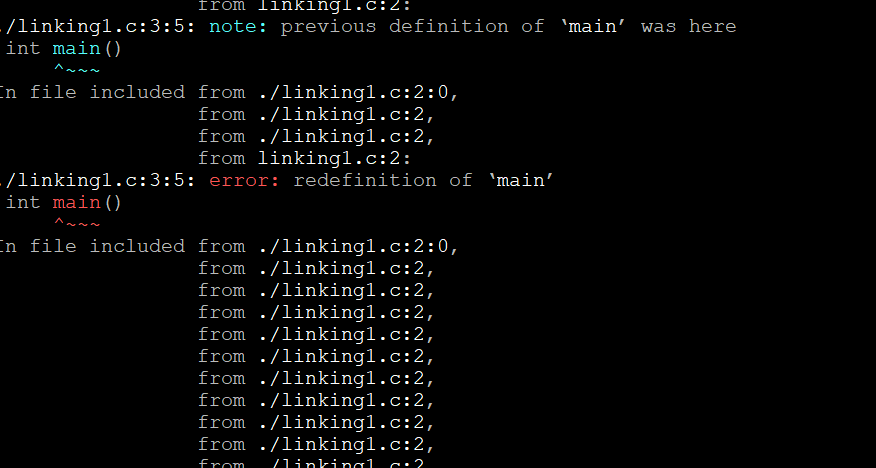
gcc –c –Wall linking1.c

gcc –c –Wall linking2.c

gcc linking1.o linking2.o

A computer screen with red and blue text

Description automatically generated



16. Do not include the file linking2.c in linking1.c. Instead of that include the file prototype.h and proceed as mentioned in the points 15 and 16.

A screen shot of a computer code

Description automatically generated