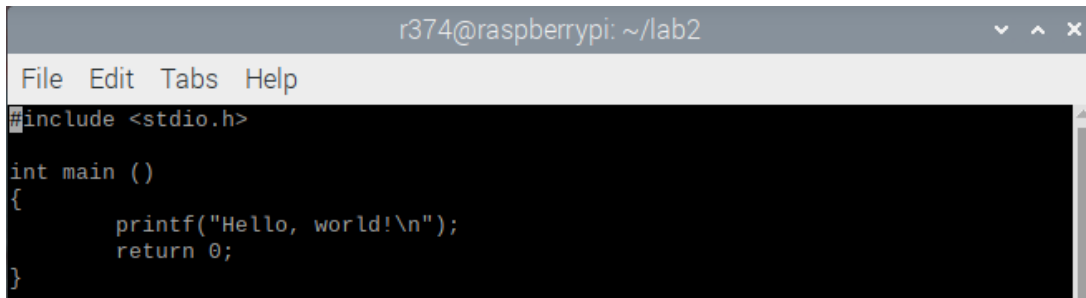


ENEE 340 Lab 2 Report

Procedure

First, I created a subdirectory in my home directory “/home/r374” called “lab2.” From here, I created a ‘hello.c’ file within the “lab2” subdirectory using the ‘vi’ command. Using the ‘i’ and ‘a’ command within the ‘hello.c’ file, I wrote the following code which can be seen in Figure 1.

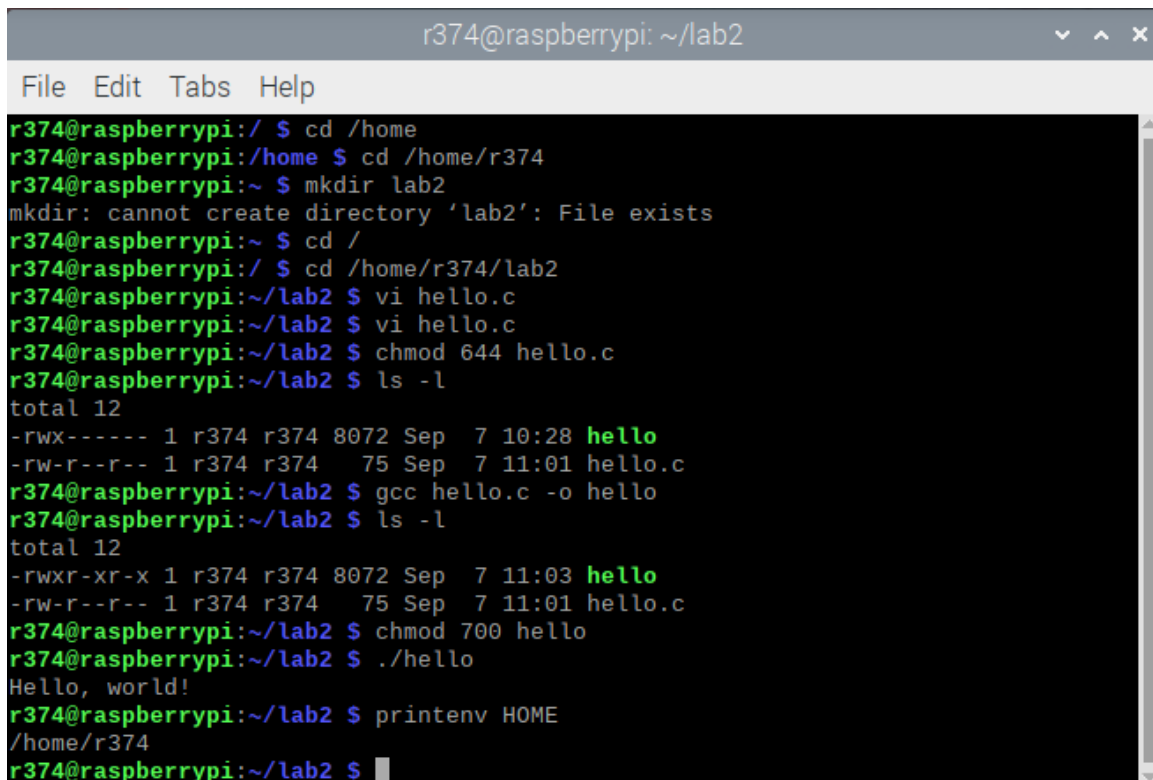
A screenshot of a text editor window titled 'r374@raspberrypi: ~/lab2'. The window has a menu bar with 'File', 'Edit', 'Tabs', and 'Help'. The code inside is as follows:

```
#include <stdio.h>

int main ()
{
    printf("Hello, world!\n");
    return 0;
}
```

Figure 1: hello.c source code (7 lines, 74 bytes)

After that, I compiled it and ran the resulting executable file in the command terminal. I also displayed the HOME variable using the ‘printenv’ command. The results of all of these commands are in Figure 2.

A screenshot of a terminal window titled 'r374@raspberrypi: ~/lab2'. The terminal shows the following commands and their outputs:

```
r374@raspberrypi:/ $ cd /home
r374@raspberrypi:/home $ cd /home/r374
r374@raspberrypi:~ $ mkdir lab2
mkdir: cannot create directory 'lab2': File exists
r374@raspberrypi:~ $ cd /
r374@raspberrypi:/ $ cd /home/r374/lab2
r374@raspberrypi:~/lab2 $ vi hello.c
r374@raspberrypi:~/lab2 $ vi hello.c
r374@raspberrypi:~/lab2 $ chmod 644 hello.c
r374@raspberrypi:~/lab2 $ ls -l
total 12
-rwx----- 1 r374 r374 8072 Sep  7 10:28 hello
-rw-r--r-- 1 r374 r374  75 Sep  7 11:01 hello.c
r374@raspberrypi:~/lab2 $ gcc hello.c -o hello
r374@raspberrypi:~/lab2 $ ls -l
total 12
-rwxr-xr-x 1 r374 r374 8072 Sep  7 11:03 hello
-rw-r--r-- 1 r374 r374  75 Sep  7 11:01 hello.c
r374@raspberrypi:~/lab2 $ chmod 700 hello
r374@raspberrypi:~/lab2 $ ./hello
Hello, world!
r374@raspberrypi:~/lab2 $ printenv HOME
/home/r374
r374@raspberrypi:~/lab2 $
```

Figure 2: Results of Linux commands

Afterwards, I created a “lab2.c” source code file within the same “lab2” directory using the ‘vi’ command. Using the C programming language, I created a program to convert Celsius temperature to Fahrenheit temperature. The contents of the lab2.c source file is below.

```
r374@raspberrypi: ~/lab2
File Edit Tabs Help
#include <stdio.h>

int main (void)
{
    float F, C;

    printf("\nWelcome to the Celsius-2-Fahrenheit Converter!\n\n");
    printf("Please enter a Celsius Temperature: ");
    scanf("%f", &C);

    F = (C*(9/5)) + 32;

    printf("Fahrenheit: %.2f\n\n", F);
    return 0;
}
```

Figure 3: lab2.c source code (17 lines, 286 bytes)

Finally, I compiled the code using the ‘gcc’ command and executed the output file. The results of these commands are in Figure 4.

```
r374@raspberrypi: ~/lab2
File Edit Tabs Help
r374@raspberrypi:~/lab2 $ vi lab2.c
r374@raspberrypi:~/lab2 $ gcc lab2.c -o lab2.exe
r374@raspberrypi:~/lab2 $ ./lab2.exe

Welcome to the Celsius-2-Fahrenheit Converter!

Please enter a Celsius Temperature: 30
Fahrenheit: 62.00

r374@raspberrypi:~/lab2 $
```

Figure 4: lab2.c compilation and execution

After executing the “lab2.exe” file, I was able to input the test value, 30 degrees, and receive the correct corresponding Fahrenheit value of 62 degrees.

Conclusion

After completing this experiment, I have learned and applied various details about directory manipulation, file creation, compilation and execution. Throughout this lab, I encountered obstacles and lessons about syntax that I was previously unaware of. Learning how to edit code without accidentally pressing incorrect keys that would cause tedious errors was very irritating. Nevertheless, I was able to practice and perfect my technique as well as learn new shortcuts like “a” and “o” in editing mode.

I also learned how to formulate a basic C program using what I learned about fundamental coding structure and pseudo code from class. In addition, I learned how to debug issues concerning ‘.swp’ files that formed from incorrectly saving my source code files.

In the future, I would like to apply what I learned to bigger projects such as creating a database of inventory that could be modified externally using commands. I would suggest creating advanced programs like calculators or multifarious unit converters in future labs.