



# SACRAMENTO STATE

MARIO PALACIOCS  
LAB COURSE CpE 185 – SECTION 03  
MONDAY (6:30PM – 9:10PM)  
LAB 03: RASPBERRY Pi LAB  
INSTRUCTOR: SEAN KENNEDY

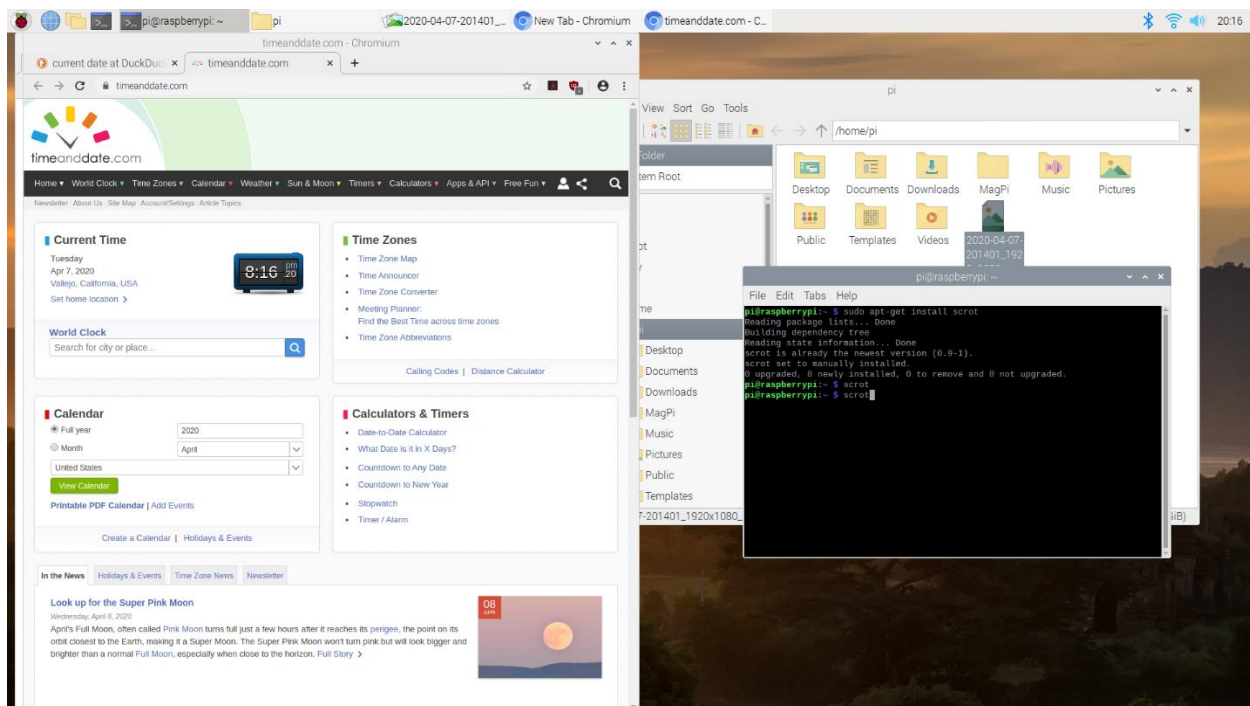
## **Introduction:**

In this lab I will be demoing parts 1 – 6. The goal of the lab was to introduce us to another microprocessor, the Raspberry Pi. I had bought the Raspberry Pi Model 3 B+, and installed the Raspbian Operating System. Throughout the completion of this lab I learned more on how to navigate a microprocessor, through its UNIX shell to connecting it to our breadboards, all useful knowledge that can be applied to industry.

## **Part 1. Raspberry Pi Setup**

**Description:** To setup Raspberry Pi and connect it according to the instructions on the Raspberry Pi webpage.

## **Engineering Data:**



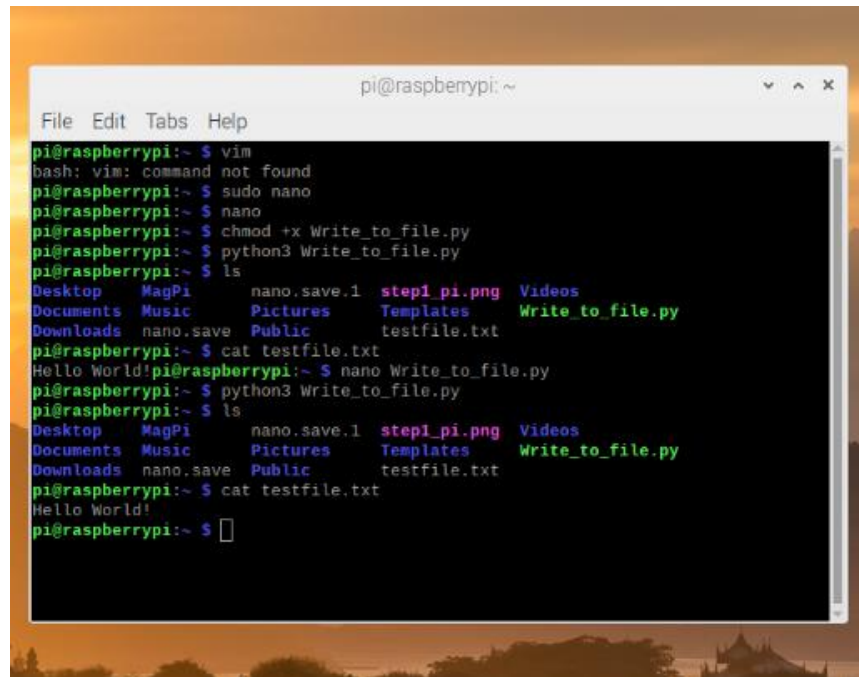
**Figure 1. The Desktop Page to Raspbian and a Webpage That Shows a Successful Connection to the Internet**

**Conclusion:** The first time setting up a Raspberry Pi was successful, where I learned how to properly connect a keyboard, mouse and HDMI cable to it, while also downloading the correct image. I had also taught myself how to take screenshots on it, through a program call *scrot*.

## Part 2. Python on Raspberry Pi

**Description:** To make sure Python is installed while also practicing writing to a file and writing to a CSV file all in Python.

### Engineering Data:



```

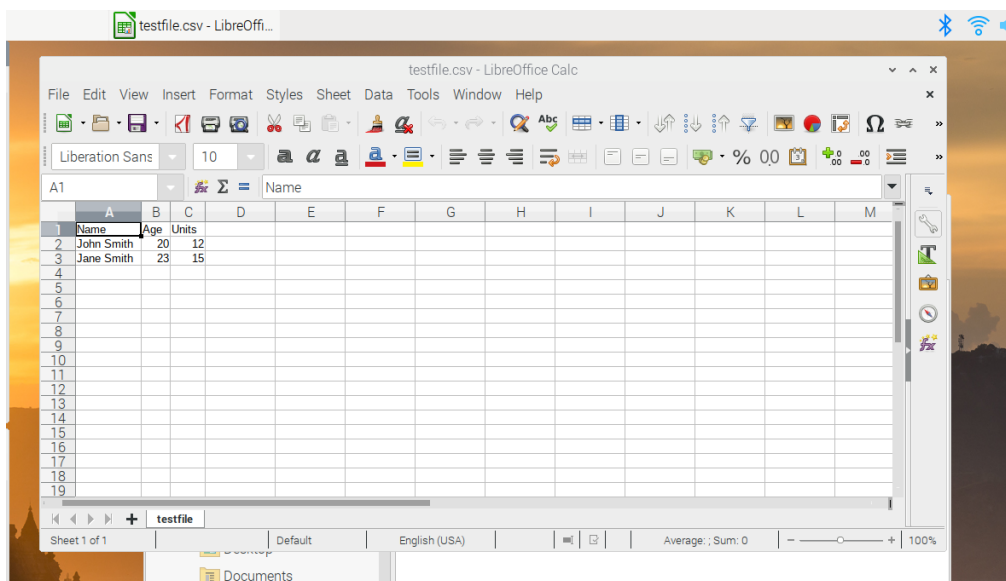
pi@raspberrypi:~
File Edit Tabs Help

pi@raspberrypi:~ $ vim
bash: vim: command not found
pi@raspberrypi:~ $ sudo nano
pi@raspberrypi:~ $ nano
pi@raspberrypi:~ $ chmod +x Write_to_file.py
pi@raspberrypi:~ $ python3 Write_to_file.py
pi@raspberrypi:~ $ ls
Desktop  MagPi      nano.save.1  step1_pi.png  Videos
Documents Music      Pictures    Templates     Write_to_file.py
Downloads nano.save  Public      testfile.txt

pi@raspberrypi:~ $ cat testfile.txt
Hello World!
pi@raspberrypi:~ $ nano Write_to_file.py
pi@raspberrypi:~ $ python3 Write_to_file.py
pi@raspberrypi:~ $ ls
Desktop  MagPi      nano.save.1  step1_pi.png  Videos
Documents Music      Pictures    Templates     Write_to_file.py
Downloads nano.save  Public      testfile.txt

pi@raspberrypi:~ $ cat testfile.txt
Hello World!
pi@raspberrypi:~ $
  
```

Figure 2. Writing to a File, Printing Out “Hello World” on Raspberry Pi Terminal



testfile.csv - LibreOffice Calc

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Name	Age	Units										
2	John Smith	20	12										
3	Jane Smith	23	15										
4													
5													
6													
7													
8													
9													
10													
11													
12													
13													
14													
15													
16													
17													
18													
19													

Sheet 1 of 1 | Default | English (USA) | Average: ; Sum: 0 | 100%

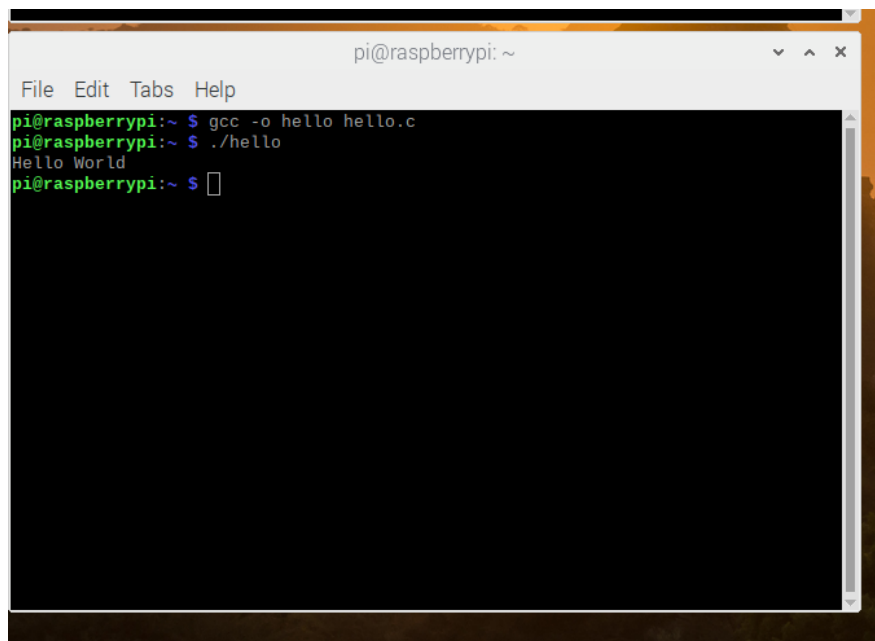
Figure 3. Writing to a CSV File, Displaying Values in a Spreadsheet

**Conclusion:** Even before writing any code in Python it is important to check if the most updated version is installed on the Raspberry Pi. When writing to a file it is important to classify the file type to “.txt” or else it will not run the script on the terminal, as seen in Figure 2. Writing to a CSV (*Comma Separated Values*) the file type will no longer be a “.txt” but a “.csv,” and it will be open through a program called LibreOffice Calc as seen in Figure 3. This is where the values will be imported to a spreadsheet. With each comma separating each value into its’ own cell.

### **Part 3. Writing C/C++ Program for Raspberry Pi**

**Description:** To explore the C programming side of the Raspberry Pi.

**Engineering Data:**

A screenshot of a terminal window on a Raspberry Pi. The window title is 'pi@raspberrypi: ~'. The terminal shows the following commands and output:

```
pi@raspberrypi:~ $ gcc -o hello hello.c
pi@raspberrypi:~ $ ./hello
Hello World
pi@raspberrypi:~ $
```

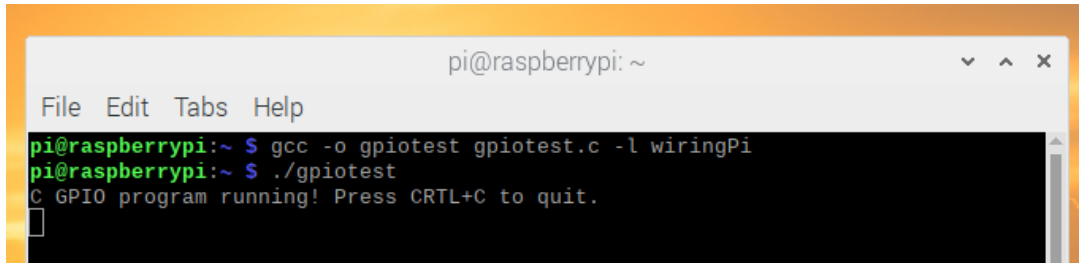
**Figure 4. Printing Out ‘Hello World’ Into the Raspberry Pi Terminal**

**Conclusion:** The Raspberry Pi OS had a gcc compiler already installed so practicing writing a code in C was fairly easy and did not take me a while to finish.

## **Part 4. Use GPIO for Raspberry Pi**

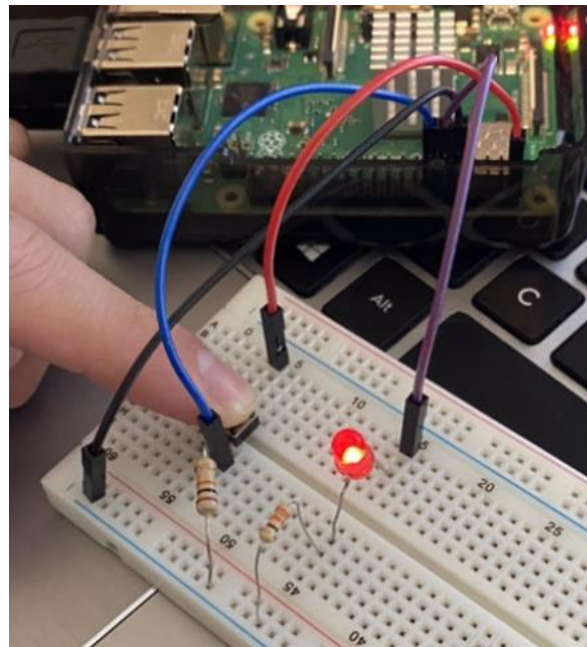
**Description:** We were tasked to create a simple LED circuit and push button circuit and connecting them using the Pi's GPIO pins, while also using the wiringPi library.

### **Engineering Data:**



```
pi@raspberrypi: ~  
File Edit Tabs Help  
pi@raspberrypi:~ $ gcc -o gpiotest gpiotest.c -l wiringPi  
pi@raspberrypi:~ $ ./gpiotest  
C GPIO program running! Press CTRL+C to quit.  
█
```

**Figure 5. WiringPi GPIOtest Program Running**



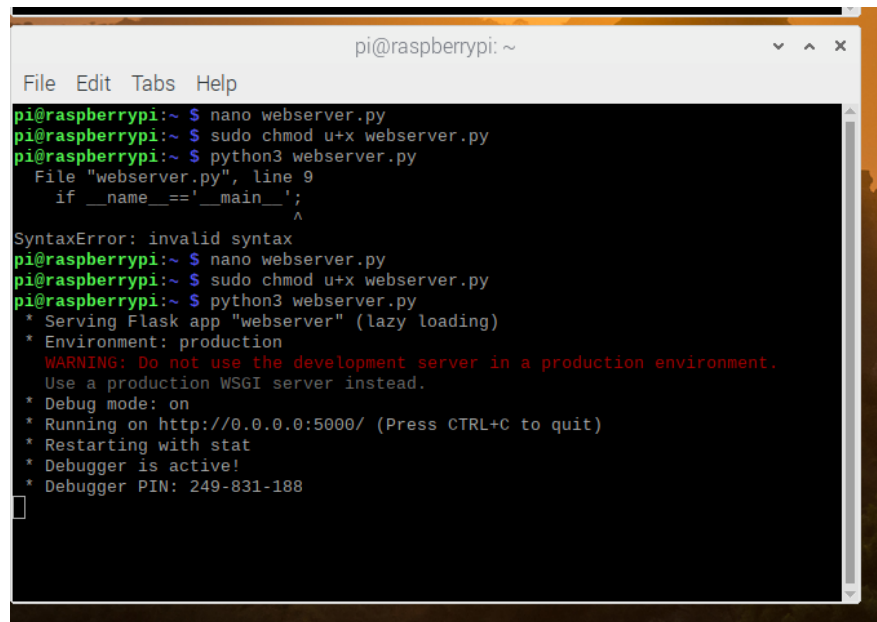
**Figure 6. Pressing Pushbutton on Breadboard to Display LED Light**

**Conclusion:** Wiring the GPIO pin from the Raspberry Pi to the breadboard was an easy task however one must be careful and make sure the correct GPIO pins are connected to its designated location. Each GPIO has their own designated variable in the C code, so everything needs to be wired up correctly. I had an issue where my GPIO 27 was connected to the positive lead of the LED and it caused my circuit not to work. However after careful examination I corrected my mistake.

## **Part 5. Web Server**

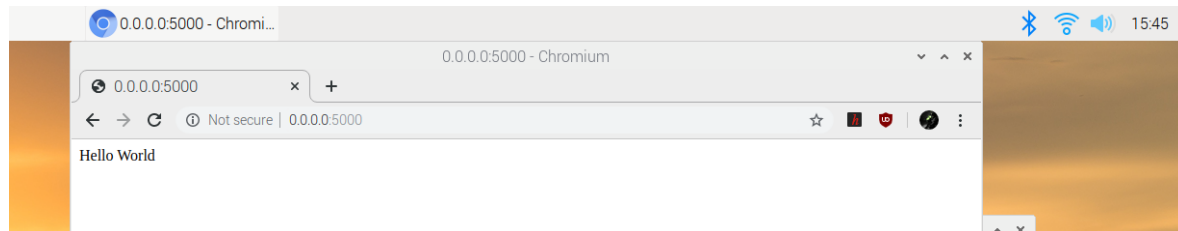
**Description:** In this part we are going to use Python with Flask framework to create a simple web server that is hosted on the Raspberry Pi.

### **Engineering Data:**



```
pi@raspberrypi: ~  
File Edit Tabs Help  
pi@raspberrypi:~ $ nano webserver.py  
pi@raspberrypi:~ $ sudo chmod u+x webserver.py  
pi@raspberrypi:~ $ python3 webserver.py  
File "webserver.py", line 9  
    if __name__ == '__main__':  
        ^  
SyntaxError: invalid syntax  
pi@raspberrypi:~ $ nano webserver.py  
pi@raspberrypi:~ $ sudo chmod u+x webserver.py  
pi@raspberrypi:~ $ python3 webserver.py  
* Serving Flask app "webserver" (lazy loading)  
* Environment: production  
  WARNING: Do not use the development server in a production environment.  
  Use a production WSGI server instead.  
* Debug mode: on  
* Running on http://0.0.0.0:5000/ (Press CTRL+C to quit)  
* Restarting with stat  
* Debugger is active!  
* Debugger PIN: 249-831-188
```

**Figure 7. Python Webserver Script Properly Running**



**Figure 8. Webpage Displaying “Hello World”**

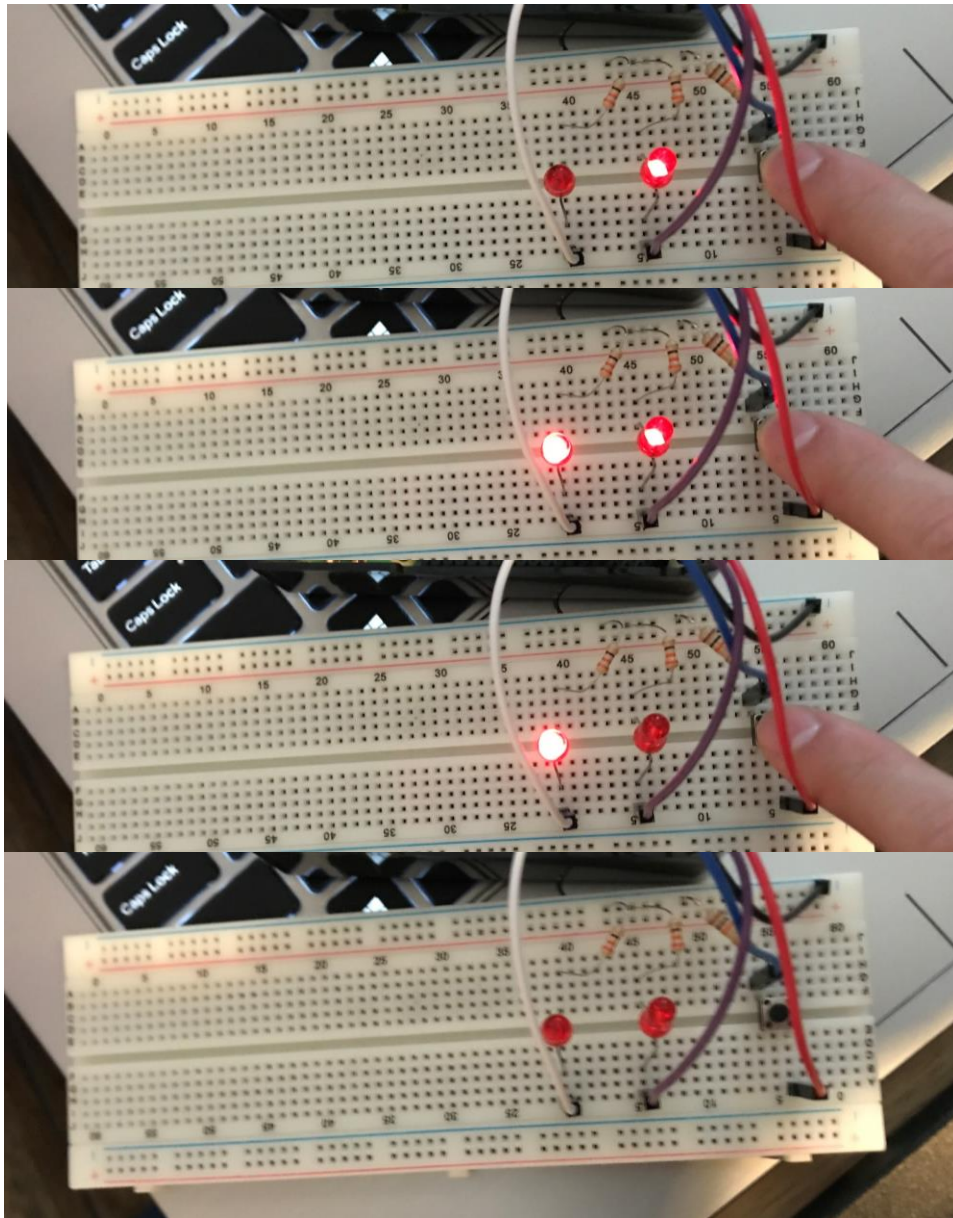
**Conclusion:** A common application of the Raspberry Pi is using the network protocols to display data or control aspects of a project through a Web Server. At the end we were tasked to display something else rather than Hello World and I had completely forgotten about that. However I did learn that by the host and port determines what to type into the web address in order to access your webserver. Flask looks like an important tool that can be used in further projects or testing.



## **Part 6. Project**

**Description:** To do a simple project that interacts with GPIO pins or some network programming.

### **Engineering Data:**



**Figure 9. Two Alternating LEDs That Share a Brief Active High and Active Low Stat**

```
#include <stdio.h>    //Used for printf()
#include <wiringPi.h> //Include Wiring Pi Library

//Pin number declarations. Using Broadcom chip pin numbers
const int ledPin = 17; //Regular LED - GPIO 17
const int ledPin2 = 22; //Regular LED - GPIO 22
const int buttonPin = 27; //Active-low button - GPIO 27

int main(void)
{
    //Setup stuff;
    wiringPiSetupGpio(); //Initialize wiring Pi

    pinMode(ledPin,OUTPUT); //Set regular LED as output
    pinMode(ledPin2,OUTPUT); //Regular LED2 as output
    pinMode(buttonPin,INPUT); //Set button as INPUT

    printf("C GPIO program running! Press CTRL+C to quit.\n");

    while(1)
    {
        if(digitalRead(buttonPin))
        {
            digitalWrite(ledPin,HIGH);
            delay(1000);
            digitalWrite(ledPin2,LOW);
        } else {
            digitalWrite(ledPin,LOW);
            delay(1000);
            digitalWrite(ledPin2,HIGH);
        }
    }

    return 0;
}
```

**Figure 10. Source Code for Part 6**

**Conclusion:** In the end of this part of the lab I was able to apply some programming knowledge learned in the previous lab to complete this part. Using GPIO pins will prove useful in my final project.



**Final Conclusion:**

Overall I feel more confident in my skills on programming microprocessors to do certain functions. Knowing how to create a Web Server will provide useful in my final project. The possibilities with a Raspberry Pi are vast and allowed us to practice our technique.