README

Name: Chris Bair

Email: Chrisbair5@gmail.com

**Intro**: This readme is intended to provide understanding to important aspects of the program and to guide intention for future features to be added. I worked hard to make this program easy to understand but realize I could have made mistakes or parts of the program may be difficult to understand. If clarification is needed I am willing to help as best I can. Questions can be sent to my email.

**Files:** Two files are used to run this program. mainMenu.py and processingcontrol.py. Images are also used but are not necessary for proper operation. The file used for testing the light sensor is testing.py and is not yet implemented into the man program.

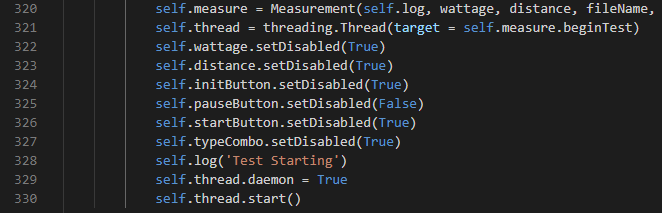
**GUI:** The file mainMenu.py contains the setup and execution for the GUI. The program should be easy to understand based on the naming convention used. However, one aspect of this program that requires further explanation is multithreading. A secondary thread was necessary to allow the program to pause or else the entire program would lock up. Below are screenshots and details on creating and using the second thread.

**(mainMenu.py)**



Import the threading library (standard python library included with python)

Import secondary file classes from processingcontrol.py



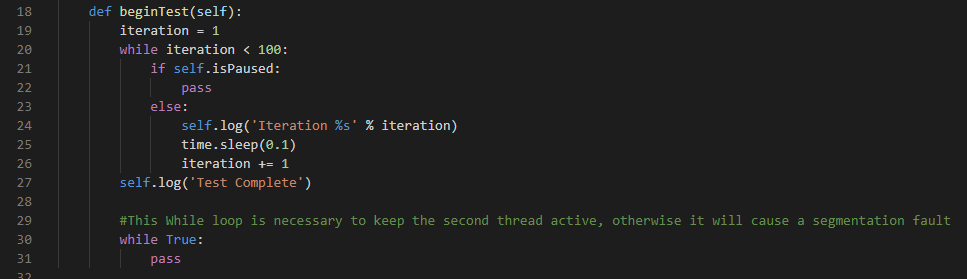
Line 320 is creating a variable to pass in information from the user input fields to pass into the Measurement class in processingcontrol.py

Line321 is creating a thread specific to the function, beginTest, in the class Measurement from processingcontrol.py

Line 329 is setting the thread to allow the main thread (GUI operation) to continue operation while the new thread (test operations) continues in the background.

Line 330 starts the thread. Recall from line 321, this is a specific function call beginTest.

**(processingcontrol.py)**



This function is the start of the second thread.

Most of this function is simply printing to test the that the secondary threading is working.

The important parts of note is the variable self.isPaused and the while loop on line 30.

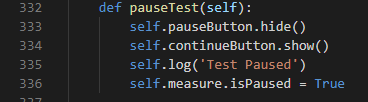
The **while loop** on line 30 is keeping the thread active after its purpose is complete.

**This is important!!**

Removing this while loop will cause a segmentation fault and crash the program. Further testing could be performed with terminating the thread after completion of test. However, keeping the thread active seems to be sufficient. There does not appear to be any downside to this method, but this is one aspect I would have considered looking into with more time.

**self.isPaused** is a variable set by the pause button from the GUI in mainMenu.py

**(mainMenu.py)**



**Motor control and sensor data**

These functions are not yet implemented yet. However, some code is available. The sensor data collection test program is labeled testing.py and some basic motor control is on the desktop of the Raspberry pi. The motor control programs are not ready for implementation with the main program and will likely need to be rewritten which is why they are not included in the github.

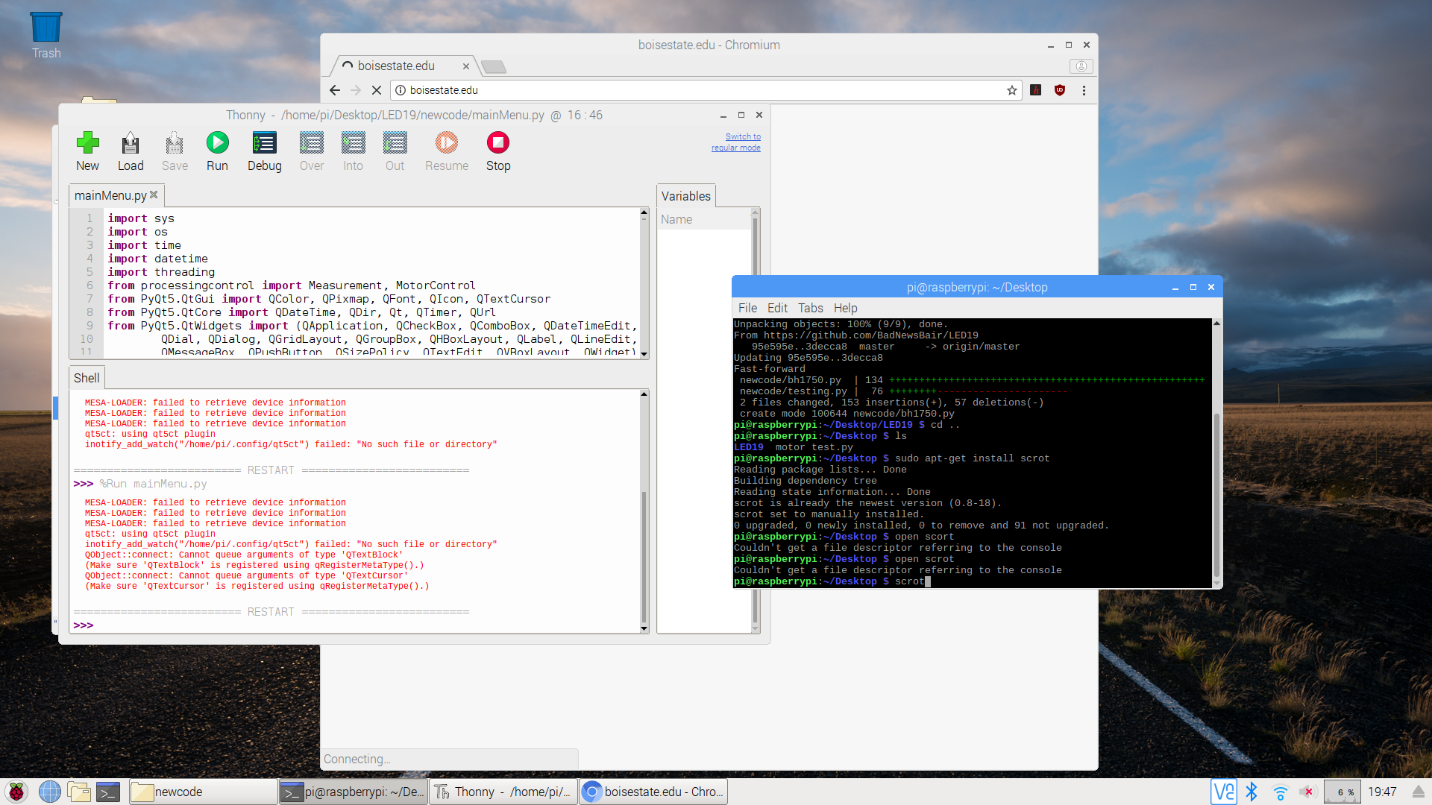
The beginTest function in processing control is intended to be where most of the test program is written. A loop for the pause function should be the outer while loop. Inside this loop should be additional loops for motor control and data collection. To keep the function small motor control functions are used. This is the idea I had, but changes could be made as needed.

**File management**

Saving the data was intended to automatic after the test is complete. This can still be accomplished but has not yet been implemented. Variables are available that contain user inputs which would be used for file naming. This part of the program should be straight forward and may require only 3-4 lines to implement.

**Additional Notes:**

**Errors**



The errors seen here, while annoying, do not adversely affect the program.

A fix for most of these errors is to remove the pyqt5 library and reinstall it. We had a new build on another SD card that didn’t have this issue, but we broke the SD card and had to revert to the current card at the last minute. This will fix the errors except the QOBJECT::connect:: error. This is due to the secondary thread calling the main thread to output to the log, which is not the correct way to do this. A fix for this issue is to use signals and slots, but I am unfamiliar with how to implement these topics. However, the error does not prevent the program from working properly.

**Single executable**

The idea at the end of this project was to create a single executable for ease of use. However, the program was not completed. Once the program is completed there are a few options to create an executable file. Pyinstaller is one option and is what I use most often.