Ishva Patel
Prof. Salazar
CS 4348.502
10/3/2022

Project 1 Write Up

Approach to the Project:

My approach to the project was first to get the driver and the logger file to connect and work together. I did this because to it felt easy to go back and add the encryption file later in wherever the logger file is taking in text. After I got the logger file and the driver file to work together, then I began integrating the encryption text. Within the encryption text, I first started writing all the methods necessary for encrypting and decrypting the text that was input. I made sure that the encryption program would work by itself by manually inserting it into the encryption program. After the encryption, worked by itself, I went back to connecting the encryption file to the driver file. I did this by sending the user input, to the encryption file, this information would have the command, then the encryption program would go to the specific if statement and do the task for that command. After I got the encryption and the driver program connected, I worked on returning results from the encryption program and sending it to the driver program, and sending the result to the logger program, to get the result logged into the logger text file.

Organization of the Project:

The project is organized into three individual programs, driver.py, logger.py, and encryption.py. The driver program has a while loop, that goes until the quit command is pressed. Inside the while, there are if-elif-else statements that correspond to each of the commands that are called. This organization of the program, allows the menu to keep showing up until the user decides to quit the program. The if structure allows having different options that can handle the commands. The encryption program has a similar organization to the driver file. It has a while loop and it goes until the user quits. Followed by a series of if-else statements that check the command necessary. In encryption there are also methods at the beginning of the program for generating the key, encrypting, and decrypting. This allows for calls to the method instead of placing the code into the if statements. The logger file has a similar organization to the encryption file. The program has a while that runs until the quit command is going followed by a series of if structures that have the specific commands that log and write to the file.

Problems Encountered:

The problems that I encountered, were mainly related to the pipes that were created in the driver program. Specifically, I had an issue trying to figure out how to send variables from the encryption file to the driver program. In addition, another problem that I ran into was debugging the logger and encryption program. This was an issue because the stdout was not to the terminal but instead to the Pipe. By doing that I had a hard time trying to find errors in the logger and encryption file.

Problems Solved:

I was able to figure out the issue of sending variables from the encryption file to the driver file. I figured it out by experimenting with the flush statements, print statements, stdin statements, and

stdout.write statement. Eventually, I found a combination of the stdout.write and a flush that would send the variables from the encryption file. The other issue of debugging, I was able to solve by creating a dummy file, that would send print messages to those places, so I could know where the program was stopping. This made it so I could easily debug, by looking at the text file that was created.

What I learned from the Project:
I learned how to use multiprocess in python. I learned how to create pipes in python using the submodules process. In addition to that, I was able to learn how to send a variable from the child process to the parent process, using the stdout.write messages. In addition to that, I also learned how to write complex programs in python. Learning this was significant because I can now apply the concepts of multiprocessing in a real-life project which can be helpful in visualizing multiprocessing and multithreading in a real-life application.