Eduardo Ramirez Torres

UID: 205-317-677

CS31 Lecture 1 Discussion 1D

**Report for Project 3**

1. Project 3 has been the most difficult task so far due to how many obstacles I ran into. As with most coding assignments, I had to keep re-reading the specs in order to fully grasp the concept of what to do. There were some portions of the warm-up and FAQs that I did not read, which lead to some headaches and frustrations. An example of this was when I began to build the function plotline. I was trying to see that, if the mode was background and the current position contains a space, then place the character there (otherwise, skip it and go to the next grid position, either horizontal or vertical depending on the mode). However, when I implemented the sample code that made sure that plotline works, it would not draw the # characters, which were in background mode. After going to tutoring, it was pointed out that I incorrectly compared an int to a char (if ((columns + k) == ‘ ‘)). Instead, I should have used getChar to covert the int to a char variable type. Then came along the entirety of Phase 3. The amount of obstacles I ran into in this phase was enormous. The main challenge I had was figuring out how to parse the command, check its syntax and logic errors, and plot the command afterwards. It took a long time, and a lot of help from office hours, but I got through Phase 3. Afterwards, I had to fix the small errors in Phase 3, such as errors in setting background characters and plotting limits. Overall, Project 3 was tough but fun when it worked.
2. I created multiple helper functions to detect inputs for H, V, B, F, and C, along with their lowercase counterparts. For plotline, I detect all possible syntax errors in case it gets tested without performCommands. After detecting errors, I check if we want a horizontal or vertical line, and plot that respective line. That line’s properties change depending on foreground or background mode, starting point, and distance. performCommands is the function that calls helper functions to check syntax and logic errors. If there are no errors, then the function will parse the command string into smaller, executable sets of commands that will result in what the user originally wanted to draw. How it will draw it depends on the way it runs through all of my if-else statements.
3. Test Cases:
   1. h12V3H-1B@v-3: check if it works with the default test case
   2. v2b h12fHh1fih0: check the default test case more
   3. CV14: goes further into the default test case
   4. H25,H-10: comma not allowed between plotting commands
   5. H25 H-10: space not allowed between plotting commands
   6. H+25H-10: ‘+’ not allowed in H command
   7. h30: test if the line will draw all the way through the grid
   8. ccccccccccCCCcCC: see how many times it can run the same command
   9. B&f\*b^f$B+f/h2: check how the program prioritizes setting modes
   10. H29v19h-28v-18h27v17h-26v-16h25v15: creating patterns