CMPS 455

Andrew Shullaw
c00161818

September 15, 2021

Project 1

1

As part of your report, answer the following questions about Task 1:

1. Why is the ability to check input so important?

1. Reading user input/files is what computing is all about really (in my mind.) Since day one at this university, I've been taught about user input. It has been the bane of my existence at times (backslashes), but it is absolutely necessary. What else do we honestly do!? Humans, files, reading, sharing, learning; its all through input.

2. Other than simply providing the wrong type of input, what other ways can you think of for bad input to cause an error? Consider situations other than typing input when prompted.

2. File input, but I guess that is still a user. The 1996 spaceship that blew up was because of bad input. I mean, we need to be really explicit here because the more I say input, the more I'm like "EVERYTHING IS INPUT!"; so yes I guess until someone corrects me–even the programmer himself is creating "input".

2

As part of your report on this project, answer the following questions about Task 2:

1. Remove the busy waiting loop used whenever a thread shouts and run the task with 6 shouters and 6 shouts per shouter. Then have each thread yield once after shouting and run another test with the same parameters. Note your results and explain your observations. Undo any changes made to accommodate this question before submitting your assignment.

1. Six threads are created (in the same order as they were with the loop.) Without the busy waiting loop they do everything in order. The output I have is about 24 lines, where as with the random busy waiting loop there is much more randomness (loops happening out of order).

2. Temporarily disable your input validation, run a minimum of 6 tests with garbage input, and note the results. How would an end user react to this? Undo any changes made

to accommodate this question before submitting your assignment.

2. I for the most part have valid output for garbage input. Explicit instructions are given before hand. They are notified when they have done something they were told not to–other than too many characters. I previously had it working, but textitc'est la vie. I think the end-user would understand what they had done wrong and be understanding unless they were trying to earn points on StackOverflow.

3

As part of your report, answer the following question about Task 3:

1. What other solutions can you think of to handle improper input on the command line?
1. Using a more robust system that allows for importation of more modern libaries and can handle the use of grep and the like.

If we aren't concerned about UI/UX, then accepting one character at a time seems to be the safest bet. Having all input come in on one line is also a safe bet. The fewer backslashes allowed, the better. Only give the user a number pad and give them extremely explicit instructions.

4

You must turn in a report on this assignment along with your code. In addition to the questions listed under each task, the report should answer the following:

1. In your own words, explain how you implemented each task. Did you encounter any bugs? If so, how did you fix them? If you failed to complete any tasks, list them here and briefly explain why.

1. First, read K&R, think you know exactly what you're doing, and then remember that you're a Python coder at heart and only dream of writing robust explicit code in C.

Next, I went with getchar() for task 1. It is what they did in K&R and it worked out well for me. Perhaps it was a bit overkill, but I can say without a doubt that I can create a C program without a connection to the internet.

Task 2, I should've read it before I did task 1. If I could go back, I would've preferred keeping the same function as task one. I just couldn't figure out how to overload the function to return a different value. On top of that, I wouldn't have created a struct, then a class, or put anything in thread.h. Then again, I learned from it.

Task 3, I had already kind of done this in Homework 1. I just had to fiddle with it a little more, and of course I got to use some of my getchar() logic (although I didn't use getchar()).
2. What did you learn from working on this assignment?

2. I learned a good bit about data structures, the C Programming language for sure, and all of the things that make my life so much easier as a non-C Programmer. I need

to stop for a second and note to myself and the reader that I keep talking about C rather than an OS. I really can't say this has taught me much about an OS. I guess we will leave that to the future projects. I certainly would love to learn more about C/C++, but this is definitely a strange environment to do it in (with all of the import errors and what not– probably my fault.) 3. What sort of data structures and algorithms did you use for each task?

3. I used ints, chars, and arrays for task one, created a class for task two (which may or may not have be necessary), and just some logic for task 3. Oh, well I suppose I did use external variables for this entire project. Those are useful, although I could not implement the one for task 3 as described in the Project 1 Info pdf. I would like to learn more about how to do that, but I think it probably just is going to require me to become more familiar with the entirety of the code.