Lab 01 Specification – Exploring C
Due (via your git repo) no later than 8 a.m., Friday, 7 September 2018.
50 points

Lab Goals

- Review basic lab tools (GitHub & Slack).
- Practice with C programming.

Summary

You will do a few exercises just to refresh your memory of git, write a short C program involving character i/o and post it to Slack for everyone to see and/or play with, then write several more C programs to post on your repository for grading.

Suggestions for Success

- Take a look at the suggestions for successfully completing the lab assignment, which is available at: `https://www.cs.allegheny.edu/sites/amohan/resources/suggestions.pdf`

Learning Assignment

If you have not done so already, please read all of the relevant "GitHub Guides", available at https://guides.github.com/, that explain how to use many of the features that GitHub provides. In particular, please make sure that you have read guides such as "Mastering Markdown" and "Documenting Your Projects on GitHub"; each of them will help you to understand how to use both GitHub and GitHub Classroom. To do well on this assignment, you should also read

- PH chapter 01 - section 1.3.
- KR chapter 01 - section 1.1 - 1.5.
- Class discussion notes, slides, and in-class coding files.

Assignment Details

1. **[GitHub Setup.]** Take a look at the detailed documentation for getting started with GitHub, which is available at: `https://www.cs.allegheny.edu/sites/amohan/resources/github.pdf`
2. **[Modify a C program and post it to Slack.]** For this problem only, you may work with a partner. However, you should each upload different programs () On page 17 of K&R there is a program to copy a file. To run it on a file named myfile.txt, you type:

```
./a.out < myfile.txt
```

Let's modify the program so that it prints out only every other character (other than the newline character, which will always be printed), i.e., the first, third, fifth, seventh,$\cdots$

We can do this by adding a counter and only printing when the counter has an even value. I have created such a program and included it in the "resources" directory inside the lab-01 repository that has been shared with you. I have also created a text file named "ritchie.txt" that you can use for test data, which is also available in the "resources" directory.

Read the comments to make sure you understand how the program works, then make the following changes and upload the resulting program to the general channel in the class Slack team page.

- Change the header comment so that it has your name and a description of the modified program.
- Change the behavior of the program in a nontrivial way, different from any of the examples given in the book. Here are some suggestions:
  - print only the vowels in the input file.
  - print only the non-vowels in the input file.
  - print only every other line of the input file.
  - print at most 20 characters per line (in other words, print a newline character after every 20th character that gets printed). Print all newlines in the file just as they are, but dont include them in the count.
  - **[Tricky!]** Using the information at the bottom of page 19 in K&R, print the file, changing all vowels to upper case (you may not use any built-in C functions for things like testing for upper-case or testing for letters).
  - **[Preferable!]** Come up with your own (simple!) variation of the file copying program. Remember, this is to be shared with the class, so keep it simple.

You may build on the programs of other in the class, but you must give credit to anyone whose ideas you use for your own program. Your program must be uploaded to the Slack general channel no later than 8 a.m., Tuesday, Sep. 4.

3. **[Write Some More C Programs.]**

   (a) Write a C program modeled after the ones in sections 1.5.3 and 1.5.4 of K&R that does the following: Given an input file, print it with leading line numbers, starting with line 1. Assume there are no more than 999 lines in the file. Line numbers should be right-justified in the first 3 columns.

   (b) Write a C program modeled after the ones in sections 1.5.3 and 1.5.4 that does the following: Given an input file, count the number of vowels and consonants and print out these counts, appropriately labeled. The vowels are a, e, i, o, and u (both upper and lower case); all other letters are consonants. You may not use any built-in C functions for checking for upper-case, etc.

4. **[Upload your C programs to your repository.]**
   Make sure your programs have header comments, including your name, explaining what the program does. ("Lab 1 assignment" is not an adequate comment. I want a description of what the program does, even if you just paraphrase my descriptions above.)

Submission Details

1. Push your programs into the repository you shared with me.
2. Your two programs must be uploaded to your repository no later than 8 a.m., Friday, Sep. 7.
3. Questions about the lab? Bring them to class on Tuesday morning!

Before you turn in this assignment, you also must ensure that the course instructor has read access to your git repository that is named according to the convention `cs200F2018-<your user name>`.

Grading Rubric

1. Task 1 in this assignment is a preliminary requirement to complete this lab assignment. There is no explicit points awarded for completion of this task as such. It is highly recommended to take this seriously and complete it as per the instructions provided, so that it can set you up nicely for the rest of the semester.
2. If you complete Task 2, it will help you do well in Task 3 and also improve your C coding skills. There is no explicit points awarded for completion of this task as such.
3. If you complete Task 3 fully, as per the requirement outlined above, you will receive a total of 50 points. Each sub question is worth 25 points.
4. If you do not follow the guidelines of Task 4, there will be 2 points deducted from your total points for the lab.
5. Failure to upload the lab assignment code to your git repo (Task 4 completion), will lead you to receive ”0” points given for the lab.
6. There will be a partial credit (30% of the points allocated for the task) given if your code compiles correctly but not able to produce the expected result.
7. There will be no partial credit awarded if your code doesn’t compile correctly. After the grade is released, you will still have an opportunity to interact with the instructor and your grade may be changed if deemed appropriate. It is highly recommended to validate if the correct version of the code is being submitted before due date and make sure to follow the honor code policy described in the syllabus.