EECS 281 Lab 01

Due Friday 9/21 at 11:59pm

Before you attend your first lab section (the second week of classes in Fall/Winter, the end of the first week in Spring), be sure to complete the Lab01-Prelab document! Be prepared before you come to Lab, and read this document so that you know what you're going to do. You can even start it ahead of time, and get as much of it done as you're able, then attend your lab section for help completing it.

Canvas - 5 points

1. You're writing a simple text-based game on the command line. When the user runs your executable (`./281quest`) the game starts at the first level. If the user wants to skip to a certain level, they can instead run the program with `./281quest --level 5` (where `'--level'` is followed by the level they want to skip to).

   Which of the following correctly specifies this flag for getopt?
   a. `{"level", no_argument,       nullptr, 'l'}`
   b. `{"level", optional_argument, nullptr, 'l'}`
   c. `{"level", required_argument, nullptr, 'l'}`

2. We're writing a program that will take 5 command line options: **-k** --katie, **-n** --noah, **-o** -oliver, **-d** --drew, **-s** --sam. Options **n**, **d**, and **s** will have required arguments; the rest take no arguments. Which of the following strings is a correct short options string?
   a. kn:ods:
   b. k:n:od:s
   c. kn:ods
   d. kn:od:s:
   e. k:no:d:s

3. Assuming the same options as above, is the following a valid longOpts array?

```
struct option longOpts[] = {{"katie", no_argument, nullptr, 'k'},
                            {"noah", required_argument, nullptr, 'n'},
                            {"oliver", no_argument, nullptr, 'o'},
                            {"drew", required_argument, nullptr, 'p'},
                            {"sam", required_argument, nullptr, 's'}};
```

   a. Yes
   b. No

Coding Assignment - 10 points

For this lab you will be familiarizing yourself with the `getopt_long()` function, as well as our Makefile and the autograder. To accomplish this task, there are two files in the Canvas Lab 1 folder called “Starter Files”: `lab1.cpp` and `sorting.h`. There is nothing for you to do in `sorting.h`, but you need to have it in the same directory as `lab1.cpp` for the code to compile. Step through `lab1.cpp` to find and complete all the `TODO` statements. There is one test file for you to test your code on in the folder named `test.txt`, and there’s a `Makefile` that you can use (more on this below).

`lab1.cpp` contains a definition for a `MusicLibrary` object. This object will read a CSV file, put the entries into a vector, and then sort the vector and print the top `n` songs based on several command line options that will be processed with `getopt_long()`. Carefully read through the comments and observe what is happening in the code; there are a lot of concepts contained in the functionality that will help you later in the course.

`sorting.h` contains the `Song` definition, as well as an overloaded `operator<<` for printing the songs. Please look around the file, but there’s nothing that you have to do in the code.

Once you have completed the coding portion, take a look at the `Makefile`. There are a few `TODO` statements in the `Makefile` as well. If the `Makefile` is confusing, don’t worry; we will go over it in lab. Once you have a grasp of how to use it, compile your code and use the command `make fullsubmit`. Make sure that you change the name of the executable to `lab1` (should be on line 27). Also make sure that the editor that you use doesn’t modify the `Makefile` and change tabs into spaces! For instance, line 79 should start with a <Tab> character, not spaces.

Finally, submit your code to the autograder. When you run the `make fullsubmit` command in the Makefile, there will be a tarball in the directory that contains all of your files. Go to the autograder, click ‘choose file’ under Lab 1, and find the tarball in your file system. Then upload your submission and watch as test cases run!

To help you get to know the autograder system a little better, **you will be refunded all of your late days that you use on Lab 1.** We recommend you try using a late day so that you can see how it works! These will be refunded 3 days after Lab 1 is due (past the time when anyone could still use late days to extend it).

For this command: `./lab1 -p 2 < test.txt`

You should get this output:

```
29 #Strafford APTS by Bon Iver has 25977731 plays.
Cold Canary Gaslight by Marty O'Reilley has 1262106 plays.
```