Programming Assignment #2

Music Library

CS 2308.255 Spring 2019
Instructor: Jill Seaman

Due: Monday, 2/18/2019: upload electronic copy by 11:55pm.

Problem:

Write a C++ program that will allow a user to query a digital library of music.

The music library will contain the following information for each song:
id (sequence of numbers and/or letters, **without** spaces)
title (may contain spaces in it)
artist (may contain spaces in it)
size (in Megabytes, could have fractional parts)
plays (number of times the song has been played, a whole number)
rating (a value between 1 and 5, the number of stars in the rating, a whole number)

Note: Your program should be able to store up to 100 different songs. You may assume that the ids will be unique (you do not need to check for this).

The program should first read the data from a text file named "library.txt". This file will contain data for each song in the library in the order listed above. The id, title, and artist will each be on a separate line. The size, plays, and rating will be on the same line. See the sample file on the class website.

Then, it should offer the user a menu with the following options:

1. Display Songs sorted by title
2. Display Songs sorted by rating
3. Lookup title and artist by ID
4. Lookup ID by title and artist
5. Quit the Program

The program should perform the selected operation and then re-display the menu.

For the **Display** operations, display the information for each song on a separate line. The values should line up in columns (use setw). The title and artist should be left justified, the remaining data should be right justified. Headers for the table are optional.

For the **Lookup** operations, label the output values. If the product is not found, display an appropriate message.

Do not change the menu numbers associated with the operations.

Additional Requirements:

- This program must be done in a **Linux or Unix** environment, using a command line compiler like g++. Do not use codeblocks, eclipse, or Xcode to compile.
- Your program **must compile** and run, otherwise you will receive a score of 0.
- The program must be **modular** (use top-down design), with significant work done by **functions**. Each function should perform a single, well-defined task.
- Use incremental development!! Implement one feature at a time!!
- Use a **partially filled array** of structures to store the library**:**
  Use a counter variable to count the number of products that are read in from the file, and use this value as the size of the array for the search and sort functions. See Pr5-22.cpp in the Assign2 provided code on TRACS Resources.
- Your program should work for an input file with any number of songs up to 100.
- Use `fin>>ws; getline(fin, str);` to input a string on a given line, including spaces, from an input file fin into a string variable str. This also works for cin.
- For option 2. Display Songs sorted by rating, if multiple songs have the same rating, it does not matter which one is listed first (or second…) within that group. Also sort in descending order: from 5 down to 1.
- You MUST use **binary search** for option 3. Lookup title and artist by ID. I recommend linear search for option 4.
- You may use (and modify) the code from the book. See the Resources tool in TRACS. Please look at this code before you start implementing your program.
- I will put a sample input file on the class website (library.txt) and the console output from running my solution on that file (output2.txt).

Logistics:

Name your file **assign2_xxxxx.cpp** where xxxxx is your TX State NetID (your txstate.edu email id).

There are two steps to the turn-in process:

1. Submit an electronic copy using the Assignments tool on the TRACS website for this class.

2. Submit a printout of the source file at the beginning of class, the day the assignment is due. Please **print your name on top of the front page**, and staple if there is more than one page.

See the assignment turn-in policy on the course website (cs.txstate.edu/~js236/cs2308) for more details, including deadlines, penalties, and where to submit printouts outside of class.