CS162 Project 5

Goals for This Project:

- More about class and OOP
- Use pointers and dynamic memory: focusing on linear linked list.
- Use memory leak detection tool

Problem You Need to Solve for This Project (Same as Project 2)

You are asked to write an app to keep track of a relatively small music library. The app should load song information from a data file once the app is started. It should allow user to view, add, remove, and search for songs. The app should save the data back to the same data file when the program exits.

What Your Program Should Do (Same as Project 2)

Write an interactive text based menu interface (using a loop) that will allow the user to

- Enter information for a new song
- Display information for all the songs in the database with index for each song
- Remove a song by index
- Search for songs by a certain artist
- Search for songs by a certain album
- Quit

For each song, you need to keep track of:

- title
- artist
- duration
- album

Allow the program to keep looping until user wants to quit. When the program starts, it should load the tasks from external file (**`"songs.txt"`**) into memory. When user enters information about the new song, the program needs to read them in, save them in memory and eventually write them to the external data file (**`"songs.txt"`**). The file format could look like:

```
Stereo Hearts;Gym Class Heroes;3;34;The Papercut Chronicles II
Counting Stars;OneRepulic;4;17;Native
```

The ';' is used as a delimiter or field separator. Each record ends with a new line character. Also the above sample data came from my teen son, not a reflection of your instructor's music taste ☺

Some Implementation Requirements: (Different from Project 2!!!)

1. Use class named `Song` to model task.
2. Use class named `SongList` to model the collection of tasks.
3. Use ***linear linked list*** to model `SongList`. Keep track of both ***head*** and ***tail*** of the linear linked list.
4. The linear linked list should be ***sorted*** by ***song title***. ***Please don't use sorting algorithms for this***. To make the list sorted, you simply insert the song at the correct position when you add it.
5. Use ***dynamic*** character array to model the strings in `Song,` such as artist and title. The character array should be the exact size as needed, e.g "CS162" should use an charcter array of size 6 including '\0'.
6. Use `destructor` to deallocate the dynamic memory for the object.
7. Make sure your program is "memory-leak-free" by using valgrind

   ```
   valgrind --tool=memcheck --leak-check=full executable-
   file
   ```

8. When using class, please make sure you encapsulate the data which means make all the instance data member `private` and provide accessor methods and mutator methods to access and manipulate the data.
9. **For submission, your data file should contain a sufficient set of test data. It should have test cases for same artist with multiple songs and same album with multiple songs in it.**
10. **You are required to have a makefile inside the project directory. The tar file should have the makefile in it. When I grade your project, after extracting from the tar file, all I will do is to type "make" to build your app and "make clean" to remove the executable and object code.**