Mini Project 2: Playlist

(100 pts, Due Friday, March 10)

You may work with a partner or you may work alone. If you choose to work with a partner, make sure:

- you both turn in the project
- you include both names on the project.
- Equally, note your partner's name in canvas.

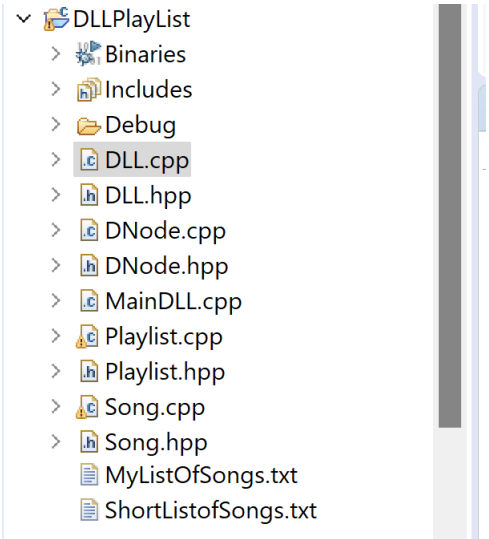
Please be aware that if your partner flakes on you, you are still responsible for completing the mini project and turning it in on time.

Time: This project may take about 12 hours, and should be worked on in a spread-out manner, as opposed to one sitting.

★★★½☆ In terms of difficulty, I consider this to be a 3.5 out of 5

This miniproject is closely tied to videos and ppts associated with these class videos

Note 2: I have included 2 songlist text files. It is a lot easier to work on this code if you are reading the songs in from a text file rather than having to type each one in every time. To make it so your code is able to read from the text files, place those text files in the project folder you've created. If you have a src or a binary folder inside the project folder, do not place the text files inside there. Place them inside the folder that holds your project (so, for instance, I created a project folder called DLLPlaylist. Inside are the following files:



Overview:

For this lab, you will be writing methods for a doubly linked list, to be used in a playlist class. For a playlist, we often add, remove, and rearrange our playlists, not to mention make new ones. So a doubly linked list is a good choice because of the sheer amount of resizing!

I am giving you:

For this project, I am giving you:

- the main function
- the class declaration and definition for a Song object (which will be your data type in the nodes in your linked list),
- The class declaration and definition for the playlist (which has as a field the doubly linked list)
- The class declaration for the DNode class
- The class declaration for the DLL (the doubly linked list class)
- The constructor definitions for the DLL class

Please download the zipped file from my website (under labs/hwks/projects) with this code in it!

Your job will be to:

(5 pt) Write the class constructor definitions for the DNode class

(this will be short), and

Write the following method definitions for the DLL class:

For writing and testing the DLL class***, please follow the instructions in Playlist.cpp.***

Note that most of what you will be doing is writing method code definitions in the DLL.cpp file, then uncommenting out code in the Playlist.cpp file and comparing your output to the output I gave you.

Once you have all the methods written and working, you can uncomment out the interface method in the Playlist.cpp and test your code on a larger playlist.

Points:

(5 pt) void push(string t, string a, int m, int s);

(6 pt) void printList();

(10 pt) int remove(string t);

(5 pt) Song *pop();

(10 pt) void moveUp(string t);

(10 pt) void moveDown(string t);

(20 pts) void makeRandom();

(7 pts) void listDuration(int *tm, int *ts);

(7 pts) Creating a brand new list (~DLL):

Once you have everything running and you've read in the new list, you should test your code by:

(15 pts)

1) add at least 2 songs (and print the list after each add)
2) remove at least 2 songs (and print the list after each remove)
3) Move up at least twice (and print)
4) Move down at least twice (and print)
5) Randomize twice (and print)
6) find the lists duration

7) play around, if you want :-)

That's it!

To turn in!

One Zipped file that includes:

1. Song.hpp
2. Song.cpp
3. MainDLL.cpp
4. Playlist.cpp
5. Playlist.hpp
6. DNode.hpp
7. DNode.cpp (with your code)
8. DLL.hpp
9. DLL.cpp(with your code)
10. A screenshot of your output
11. The two text files you used for testing.

Easiest way to do this – put all the above in a folder, zip the folder, and turn in the folder on Canvas!

```
Bye Now!
```