Comp 202 - Spring 2022
Homework #1
Due date - 23:59 13/03/2022

Homework Description

In this homework, you will implement a song playlist. “Playlist” will be a list of “Song” elements. You have to implement the “Playlist” class as a **Doubly Circular Linked List (DCLL).** Song class and Main class are already provided for you. You do not need to implement anything for those classes. They are straightforward classes, but before you start writing your code, make sure that you understand them correctly. You will implement several methods in the Playlist class. Then we will test your methods one by one, as you can see from the main class.If all of your methods are working fine you should see an output match from Github’s Autograding.

Methods that you will implement are insertToEnd(), insertToIndex(), removeSong(), move() and reverseSequence() . They are all explained below. Node class and displayList method are already provided for you, do not make changes for these two classes. You can define additional helper methods in the Playlist class if you want.

insertToEnd(Song new_song): Add a new song element to the end of the DCLL. Notice that if the DCLL is empty, this new_song will be our first song. If new_song is null, throw a NullPointerException with the message “Null song entry!”.

insertToIndex(Song new_song, int index): Add a new song element to the given index of the DCLL. Our DCLL has 1-based indexing, which means the first Node has an index of 1. When we call this method with one index we will add a song to the very first position. Index values need to be in the range of 1 to the length of the DCLL plus one. Otherwise, index value is not a valid value, and we will not perform any operations on the list. When the index value is the length of the DCLL plus one, our method will operate just like insertToEnd. Notice that if our DCLL is empty, we will only perform this operation when the index is one. If new_song is null, throw a NullPointerException with the message “Null song entry!”.

removeSong(String song_name): Given the name of the song, remove it from the Playlist. If there is no such song, do not do anything.

move(String song_name, int move_num): Given the name of the song, move it to the front by a given amount. We will not be moving in the other direction so that move_num cannot be negative. We also will not be passing beyond the head Node with this operation so that move_num cannot be greater than the distance of the given song to the head node. For those cases do not perform any operations on the DCLL. To sum up, with this method, our node will either get closer to the head node or become the head node itself. For an invalid move_num or a song_name that does not exist in the DCLL, we will not perform any operations.

reverseSequence(int first_ind, int second_ind): Given two indices revert the sequence of nodes between them. For any of the following cases do not perform any operations. If first_ind

is greater than or equal to second_ind, first_ind is smaller than one, second_ind is greater than the index of the tail and DCLL is empty. Remember that our DCLL has 1-based indexing. Figure 1 below is a simple demonstration of this method with given indices as one and four.

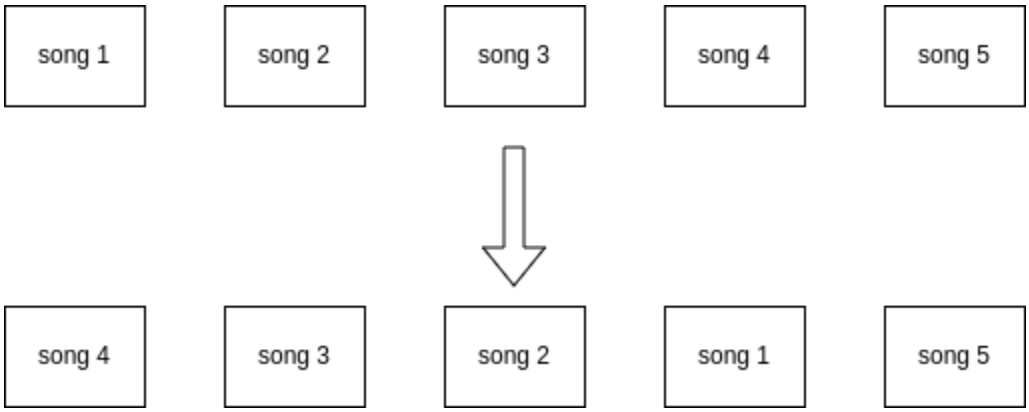

Figure 1

You can examine the example output and Main Class to understand methods better.

Keep in mind that your implementation must have **head and tail pointers**, and they should be connected to hold circular property. For all these methods, be careful where they are pointing. Do not try to implement them as separate nodes without song data; they are pointers to the first and the last nodes.

Submission Details

Usage of github classroom and submission details:
1) Accept the invitation using this link (https://classroom.github.com/a/KmyLVSfm)
a) If you have no github account, create one.
2) Choose your student ID from the list, it will be linked to your github account from now on.
3) A personal repository with the starter code will be created. You can directly edit the code on github's page, clone it to your local storage and edit it there. **Ensure that your repositories are private.**
4) Make sure your changes are committed and pushed to the repository.
5) Check if you have successfully passed the tests in the "Actions" tab of your repository. Note that these automatic tests do help you understand whether or not your solution works up to some level, but passing them does not guarantee that you will receive a full Grade.
6) To make sure your code is received and avoid any potential problems on github's side, submit a copy of java files on Blackboard as well. Submit all files as a single .zip file, named as: "ID_NAME_SURNAME.zip"

Grading Criteria

For each method you are going to receive 20% of your final grade for this homework.

For your further questions about the homework please use the Discussion Board created for this homework on Blackboard. Do not post your codes in this Discussion Board since others can also see it.

Code of Conduct

Same rules that are mentioned in the course syllabus also apply to this assignment. Please refer to that course syllabus for the code of conduct.