CS 2401 – Spring 2022

Project 2 – Sequence Classes and Dynamic Arrays

Due: 11:59 PM Friday, February 18th

The idea of a *sequence* class is that the application programmer can choose where an item is stored in the list, and that sequence, or order, of the items remains the same, even when things are deleted. In this project we are going to pass that capability on to the user allowing them to order a playlist of their favorite songs in any way they choose. Some people have hundreds of favorite songs, while others have only a few, so we are going to implement this using a *dynamic array.*

Begin this project by downloading the `main.cc, date.h, date.cc, Song.h and Playlist.h` files that I have provided on Blackboard.

`Song.h` is a header file for a class called Song. This class has three private variables: the song's name, release date, and artist. The release date is of type Date which you also used for the last project. You are to write the implementation of this class (`Song.cc`), including overloaded insertion, extraction, = =, and != operators. Two songs are "equal" only if they have the same name, release date, and artist.

You can test this class by writing a main of your own that declares two songs, lets you put the information into both, outputs them to the screen and compares them for equality.

Now, in the main that I have given you, you will find that the application allows the user to:

- Add a new song to the beginning of the list
- View all the songs in the list
- Walk through the list, viewing one song at a time
- Remove a song from the list
- Insert a new song at some spot in the middle of the list, which includes at the back end
- Sort all the songs by their release date or artist
- Find a song using its name

There is also a file backup mechanism that uses the person's username for the name of the file.

I have given you the header file for the container class that makes all this possible, `Playlist.h`. **You** are to write the implementation of this (`Playlist.cc`). The private variables for this class consist of a pointer of type Song as well as variables for capacity, used, and current_index. The constructor will begin by allocating a dynamic array of type Song with the capacity to hold five songs (this will save on memory space for those users with fewer favorite songs). When the action of adding an additional song to the list happens you should check if used == capacity, and if it does, do a resize operation that increases the size of the array by five.

This container also has an internal iterator, as the author illustrated in section 3.2 of the text, which will require that you write the functions

- `start`
- `advance`
- `is_item`
- `current`
- `remove_current`
- `insert`
- `attach`

You will also need to implement `show_all`, `releaseDate_sort`, `artist_sort`, `find_song`, and `is_song`.

Because this is a dynamic array you will need to write a resize function and the Big 3 (`destructor`, `copy constructor`, and `assignment operator`). Also, because we are providing file backup, we will need to have functions for `load` and `save`.

Your submission should include a data file of at least four songs. I would suggest testing with a data file of at least six songs so that you are sure your resize() function works properly. Your data file should be formatted as follows:

// first song name

// first song artist

// first song release date

And so on for each song.

For grading purposes, I will be using my own file of songs, and then testing all the different branches of the menu. It would behoove you to do the same with your list of at least six songs.

Submit to Blackboard Song.cc and Playlist.cc. Also, be sure that you submit the individual files instead of a zipped version of the files.