ECE250: Lab Project 1

Due Date: September 27th, 2019, 11:00 pm

1. Project Description

The goal of this project is to write a C++ program that manages a playlist of songs. The program allows the user to add songs to this list (at the end of the list), and select a song to be played. We will use the `vector` template class of the **C++** Standard Template Library (**STL**) to store the entries of the playlist. Each entry will contain the name of the song and the corresponding artist.

For this exercise, we ask you to use the object-oriented paradigm to implement your solution. You will create a design in which the playlist and the entries in the list are represented by classes that: (i) store their properties (data members) and (ii) provide services (function members).

2. The Vector Template Class

The `vector` template class is similar to a C++ array, with the added feature of performing bounds checking and the ability to expand. Since `vector` is a C++ class template, you can use this class to represent vectors of different data types (*e.g.*, `int`, `double`) or a vector of your own data type.

3. Program Design

Write a short description of your design. You will submit this document along with your C++ solution files for marking. This document must include your design decisions. Please refer to the course website for "Programming Guidelines" and the expected content for your design document.

4. Input/Output Requirements

Write a test program named **playlisttest.cpp** that reads commands from standard input and writes the output to standard output. This program will respond to the commands described in this section.

Command	**Parameters**	**Description**	**Output**
i	*s;a*	Adds a song (with the title *s* and the artist *a*) at the end of the playlist if the song is not already in the list	success or can not insert s;a
p	*n*	Plays the song at position *n*	played n s;a or could not play n
e	*n*	Erases the song at position *n*	success or could not erase n

- **Test Files**

The course website contains example input files with the corresponding output. The files are named *test01.in*, *test02.in* and so on with their corresponding output files named *test01.out*, *test02.out* and so on.

5. How to Submit Your Program

Once you have completed your solution and tested it comprehensively in your computer or on the lab computers, you have to transfer your files to the *eceUbuntu server* and test there since the automated testing is done using this environment. Once you finish testing in the *eceUbuntu server*, you will create a compressed file (tar.gz) that should contain:

- A document (maximum two pages) describing your design. This document must be typed for marking and should be submitted in PDF (*e.g.,* design.pdf).
- A test program (playlisttest.cpp*)* containing your *main()* function
- Required header files and classes (ending in *.h .cpp*)
- If your implementation requires more than one file (playlisttest.cpp) listed in g++ line for compilation, you will need to add to the design document the complete g++ command line, or provide a make file (named Makefile).

The name of your compressed file should be ***xxxxxxxx***_p***n***.tar.gz, where ***xxxxxxxx*** is your UW user id (e.g., jsmith) and ***n*** is the project number that is 1 (one) for this submission.