The goal of this lab is to help you get familiar with C++ **`struct`**s. It is also an opportunity to practice writing and debugging a program with multiple functions. Be sure that you read this entire document before you being coding.

Due Date

You must *demonstrate* the solution to this lab exercise to the instructor during class by

Saturday, October 24, 2020,

in order to receive full credit for this work.

Programming Exercise

1. Define a **`struct`** named **`MovieData`** to store the following information about a movie:

Field Name	Data Type	Description
`title`	`string`	Movie Title
`director`	`string`	Movie Director
`yearReleased`	`int`	Year Released
`runningTime`	`double`	Running time in minutes

2. Write a program that uses dynamic memory allocation to create an array of **`MovieData`** structs, populates the array with data does some processing with that data:
   - Ask the user to enter *how many* movies they wish to process. The number specified by the user will be used for the dynamic memory allocation.
   - Allocate memory for an array of the **`MovieData`** structs, using the size value from the user input.
   - Populate the array with input from the user.
   - Display the array contents.
   - Find the LONGEST movie in the array (the movie with the longest **`runningTime`** value).
   - Output the complete data record for the movie with the longest running time (as shown in the **Sample Output** on the next page).
   - Before the program exits, it should **delete** the dynamically allocated array of **`MovieData`** structs.

Program Design

Your program **must** have several different functions. Start by considering the design of the solution for Lab09a: dynamically allocate the array in the "main" function, and then call other functions to perform the various tasks. The list of **function prototypes** shown below provides a hint about how you could organize your program.

```
void displayMovie(MovieData *ptr);
void populateMovieDataArray(MovieData *arrayPtr, int arraySize);
void displayMovieDataArray(MovieData *arrayPtr, int arraySize);
MovieData *findLongestMovie(MovieData *arrayPtr, int arraySize);
```

Sample Output

The table below contains sample output for the solution to this lab exercise. In this example, we have indicated which text the user types by showing it in a larger, bold font. In actuality, all text would appear in the same size font, with no bold characters

```
Enter desired array size: 3
arrayPtr = 01235014
Enter Title 0: 2001: A Space Odyssey
Enter Director 0: Stanley Kubrick
Enter Year Released 0: 1968
Enter running time (minutes) 0: 142
Enter Title 1: The Sound of Music
Enter Director 1: Robert Wise
Enter Year Released 1: 1965
Enter running time (minutes) 1: 174
Enter Title 2: Hidden Figures
Enter Director 2: Ted Melfi
Enter Year Released 2: 2016
Enter running time (minutes) 2: 127
01235014:  arrayPtr[0] =
        Title       : 2001: A Space Odyssey
        Director    : Stanley Kubrick
        Released    : 1968
        Running Time: 142 minutes

0123505C:  arrayPtr[1] =
        Title       : The Sound of Music
        Director    : Robert Wise
        Released    : 1965
        Running Time: 174 minutes

012350A4:  arrayPtr[2] =
        Title       : Hidden Figures
        Director    : Ted Melfi
        Released    : 2016
        Running Time: 127 minutes

Longest Movie in list:

        Title       : The Sound of Music
        Director    : Robert Wise
        Released    : 1965
        Running Time: 174 minutes
Longest Movie is:  174 minutes long
DELETING array at arrayPtr = 01235014

C:\CSC237\Lab\Lab11a_Structures\Lab11a_SOLUTION\Debug\Lab11a_SOLUTION.exe (process
1864) exited with code 0.
```

Demonstrate the Working Program to the Instructor