CISC 2010 Lab: Marathon Times

Marathon.cpp

Introduction

We've been learning about structures and how to use them to describe more complicated things than a single value can describe. Now we are going to use them in a program. The program will define a Timer array that holds a Timer for each runner in a marathon. It will print the shortest and longest times for the marathoners.

Define struct Timer

Define a structure for a Timer. It should have integers for the hours, minutes, and seconds. It should be defined at the top of your program so that you can declare several functions that use it (and its variables).

Declare and define the following functions:

compareTimes

Function should take as parameters two Timers. It should compare the hours, then minutes and then seconds. If the hours from both Timers are not equal, then return the difference of the hours. If they are equal, move on to the minutes and compare them. Do the same for seconds. It should have similar overall behavior like we saw for function strcmp.

Pseudo code:

```
if (timer1 < timer2)
    return a negative value
else if (timer1 == timer2)
    return zero
else if (timer1 > timer2)
   return a positive value
```

findMinMaxTimer

Function to find the minimum and the maximum Timer in an array of Timers. It should not print the result but should return the value in the given parameters. It should use the compareTimer function to determine if a Timer has a lower value or higher value than the current minimum and maximum. As a hint, start with the first Timer in the array as both the minimum and maximum.

createTimerArray

Function to create a dynamically allocated array of Timers and return a pointer to the array. It should be similar to the ReadSequenceNumber function from the Dynamic Array Minilab. It should call `readTimer` (defined below) to assign values to the Timers in the array.

readTimer

Function to read a Timer in the format of hours:minutes:seconds (Eg. 1:15:20 is 1 hour, 15 minutes and 20 seconds) using `cin`.

Important:
cin will stop reading whenever it hits a space or a non-numerical number.

Read in a char in between the 3 ints
Like this:

cin >> hour >> aChar >> min >> aChar >> second;

aChar is declared as a char, which we will use to eat (& throw away) the **':'** char

There are other ways to read this, such as reading character by character and then converting to int, or using string functions.

Main Program

In the main program, do the following:

Declare the necessary variables for the dynamically allocated Timer array and the lowest Timer and highest `Timer`.

Call your functions `createTimerArray` and `findMinMaxTimer` and print the results.

Don't forget to free the Timer array at the end.

Sample Output

```
How many Timers do you need? -5
Enter a positive value!
How many Timers do you need? 5
Enter a Timer as hours:minutes:seconds(h:m:s)? 1:15:20
Enter a Timer as hours:minutes:seconds(h:m:s)? 0:59:46
Enter a Timer as hours:minutes:seconds(h:m:s)? 0:60:46
Invalid Timer!
Enter a Timer as hours:minutes:seconds(h:m:s)? 1:31:28
Enter a Timer as hours:minutes:seconds(h:m:s)? 1:09:09
Enter a Timer as hours:minutes:seconds(h:m:s)? 1:21:56
```

```
The minimum Timer is: 0:59:46
The maximum Timer is: 1:31:28
```