**Lab 4 - Arrays, Structs, and Pointers**

*Due Date: 5:00 p.m., October 16, 2015*

*Remember: Formatting matters. Format the output nicely using format specifiers and extra printf statements.*

Part 1: Time Difference

# Part A

* + Create a function that takes number of hours, minutes, and seconds as integers, and returns the total number of seconds.
    - The function interface should look like this:  
       int numSeconds(int hours, int minutes, int seconds);
    - The function should be declared after your main
  + Create a time struct that contains the following attributes:
    - Hours
    - Seconds
    - Minutes
  + Write a function that calculates the difference between two time structures by calculating the total seconds each time structure represents by subtracting the total number of seconds. The function should return a new time structure that represents the elapsed time (in hours, minutes, and seconds) between the two times.
    - Where time1 represents 3:45:15 and time2 represents 9:44:03, the function should create and return a time structure that represents 5 hours, 58 minutes, and 48 seconds.
    - All times are in military time (24 hour clock)
  + Do not use scanf for input. Just hardcode the following values for testing:
    - 1:30:45 and 16:30:45
    - 1:23:01 and 12:11:12
    - 00:01:01 and 23:59:59
    - 12:00:00 and 12:00:00

# Part B

* + Create another struct called DateTime that contains both the date and the time (using the time struct from part 1). The struct must contain the following:
    - time struct
    - enum type representing months *(the enum should be defined outside of the struct)*
    - day
    - year
  + Create structs with the following datetimes, and print them to the console:
    - January 19 1809 12:01
    - November 11 1922 6:00
    - January 6 2000 8:22
      * *You must print these datetimes using the struct*

Part 2 -- Tournament (Extra Credit)

In this part of the lab you are going to implement your own tournament program. You can name your teams whatever you like.

# Part A

* + You will need to create a struct called Team that contains a string buffer for the team name. After you've defined 8 teams, you will place pointers to all 8 into an array called league[], defined the following way: Team \* league[8]. This must be an array of pointers to teams, not an array of Teams.
  + Write a function called game() that takes pointers to two teams, then randomly and numerically determines a winner using the rand() function (you do not have to store the score in the struct). Your game() function should
    - take pointers to two teams as parameters
    - randomly determine a winner using rand() (don't forget to seed with srand() )
    - print out the teams names, scores, and the winner of the game
    - Return a pointer to the winner.
  + Make sure this works correctly before moving on to the next part.

# Part B

* + Once you have your league, create a function called tournament() that takes an array of 8 pointers to Team structs, then returns a pointer to a single winning team from the array. Use your game function and a loop for each round (there should be 3 rounds) to determine the winner. Because this is an elimination style tournament, each team should lose only once, while the winner goes on to the next round. The easiest way to accomplish this is to use a different array of pointers for all 3 rounds.
  + You should print the round number and the winners of each round to the console.
  + Once you have determined that the tournament() function works correctly, then go on to part C.

# Part C

* + Add a handicap (your choice of how to implement) to the Team struct, and assign a value to each team by hand or randomly (again, up to you). Alter your game function to use the handicap to weight the tournament in favor of some teams, and run your results again.

Part 3 - Submission

* Create a tar archive with the command ”tar -cvzf lab4.tar.gz .”, and then upload the archive to Blackboard before the deadline. Make sure you do not include the executable in your archive (make clean before creating the archive).
* Demo your lab by the beginning of class the on the due date by compiling your code with your makefile, then running running your code. Afterwards, show your source to me, and answer any questions I may have.

Grading Guidelines

## Part 1 (9 points):

* + Part A: 6 points
    - num seconds declared after the main with a declaration before the main and returns the total number of seconds
    - time structs holds hours, minutes and seconds
    - Contains a function that calculates the difference between two time structs and returns a structs containing that difference
  + Part B: 3 points
    - Contains an enum representing months
    - Contains a Time struct from part A
    - Contains day and year
    - Dates are stored in and printed from the struct

## Part 2 (5 points):

* + Part A: 1 points
  + Part B: 3 points
  + Part C: 1 points

## Style Guidelines

* + Follows Style Guidelines: 1 point