COT 4400: Analysis of Algorithms
Football Sorting (modified)
Due: February 25, 2014

This problem requires you to solve a sorting problem[1] and write a program that implements your solution.

You are not allowed to use the Internet. You may only consult approved references[2].
This is an individual project.

1. *Problem Description:*

   Write a program, that given the fixtures of a football championship, outputs the corresponding classification following the format specified. Win, draw, and loss earn respectively three, one, and zero points.

   The criteria of classification are the number of points scored, followed by goal difference (goals scored minus goals suffered), and then scored goals. When more than one team have exactly the same number of points, goal difference, and scored goals, these are considered as having the same position in the classification.

   - The Input: (`input.txt`)
     The first line of the input is an integer, m, then a blank line followed by m datasets. There is a blank line between data sets.

     The first line of each dataset contains two positive integers, namely t and g. The first positive integer, t, denotes the number of teams in the dataset. The second positive integer, g, denotes the number of games in the dataset. This line is followed by t lines, each containing a string, which denotes the name of team. These lines are followed by g lines giving the score of each game.

     The team names will be strings consisting only contain letters from the English alphabet and dash characters.

     The format for the scores of each game are output with the following format: name of the home team, number of goals scored by the home team, a dash, number of goals scored by the away team, and the name of the away team.
   - The Output: (<`studentLastName`>`.txt`)
     For each dataset the output consists of $t + 1$ lines. The first line should contain the number t.

     Each the remaining t lines should contain the teams in order of classification. The statistics of each team are displayed on a single line: team position, team name, number of points, number of games played, number of scored goals, number of suffered goals, goal difference, and percentage of earned points (show exactly two decimal places for the percentage). Note that if several teams are in a draw, then each of the teams in the draw should have the same position number in the output. If multiple teams have the same position (i.e., they are in a draw) in their classification they should be listed in the order provided by the input file. There should be one space between each of the fields outputted.

     Print a blank line between consecutive test cases.

   A sample input/output file will be posted on canvas.

[1] The problem has been adapted from `http://uva.onlinejudge.org/`, "Football Sorting" - Programming Challenges 10698
[2] You must cite all references used.

2. *Deliverables:* Please submit a **hard copy**, in-class, of all of the items requested below. Please don't send this by e-mail as I would have to print it out anyways. Please also submit your code and your project report[3] to Canvas.

   (a) **Theory [50 pts]:**

   - **[20 pts]:** Describe an *efficient*[4] algorithm that solves this problem. Describe which data structures you use in your algorithm.
     For full credit your description must be clear enough that any competent programmer will understand how the algorithm works and can implement your algorithm in their preferred programming language.
   - **[15 pts]:** Prove that your algorithm will solve the problem for any valid input.
     For full credit you must present a formal proof that your algorithm will produce the correct solution for any valid input. Note that in a formal proof you must justify each step of the proof sequence.
   - **[15 pts]:** Derive the worst-case complexity of your algorithm in terms of t, g, and any other appropriate parameters. You must prove that the worst-case complexity that you derived is correct.
     For full credit you must present a formal proof justifying the worst-case time complexity of your algorithm. Note that in a formal proof you must justify each step of the proof sequence. The bounds on the worst-case complexity of your algorithm must be as tight as possible.

   (b) **Implementation [30 pts]:**

   - **[20 pts]:** Submit a print-out of your code. If you are unable to get your code to compile/run, please state this explicitly.
     Code Requirements[5]:
     - Code must compile and run on the C4 Linux Lab machines.
     - Read the input from a file in the described format.
     - Output the solution in the described format using the specified output file naming format.
     - README file describing how to compile and run your code. If your code requires more than a simple command to compile and run then you must also provide a Makefile and/or shell script.
       A simple command should only include the compiler to be used and the name of a single file. If your command requires any flags or directives then you must provide a Makefile and/or shell script. For example `g++ main.cpp` *is a simple command.*
     - *Optional: You may also submit an executable for the C4 Linux Lab machines*
   - **[10 pts]:** Demonstrate your algorithm works correctly by walking through a small example.
     For the walk-through you should select a small example and discuss step-by-step (line-by-line of the pseudocode) how your algorithm produces the output. The example used for your walk-through should be large enough to demonstrate all important aspects of your algorithm, but small enough to be understood by the grader.

   (c) **Results [20 pts]:**

   - **[20 pts]:** Submit your code to Canvas to be run on a set of inputs devised by the grader.
     Your grade is based on whether or not your code produces the correct answer for all inputs.

[3] Project report and code will be checked for plagiarism

[4] You will lose points for both your algorithm design grade (i.e., lose points for poor design) and your results grade (i.e., lose points because you are not able to process all inputs) if your algorithm is not efficient.

[5] Failure to follow all requirements will result in a 0 for the code portion of your grade AND a 0 in the results portion of your grade