CSCI-1200 Computer Science II — Fall 2007
Homework 2 — Golf Classes

In this assignment you will parse and compute statistics from the results of the recent golf majors (Masters, US Open, British Open, and PGA Championships). Scoring and determining the winner of a golf tournament is a bit involved, but we have summarized everything you need to know to complete this assignment. Please read the entire handout before starting to code the assignment.

Golf Scores

Here's a crash course in golf scoring: Each player entering a *tournament* plays 4 *rounds* of golf, each round consisting of 18 *holes*. Each hole is rated by the difficulty or *par*, the expected number of shots necessary to complete the hole. Thus the entire round and the entire tournament also have a par. If a player has a good day, they will score *under par* and have a negative score for the hole, round, and/or tournament. If the player has a bad day they will be over par. The player with the lowest sum of scores over the 4 rounds is the winner of the tournament. Not all players who begin the tournament complete the tournament. Players with poor scores from the first two rounds will be *cut*, or sometimes players will *withdraw* or be *disqualified*.

As an example of the input you will be parsing, this is the tournament result for the winning player, Tiger Woods, at the 2007 PGA Championships:

```
Tiger Woods          -8       71 63 69 69       272       $1,260,000
```

Tiger's final summed par score for the tournament was -8. His four round scores were 71, 63, 69, and 69. This was a tournament total of 272 (par for this 18 hole course was set at 70). And finally as the winning player his prize money for this tournament was $1,260,000.

File I/O and Command Line Arguments

Your program will run with two command-line arguments, one being the name of the input file containing the raw tournament data and the other being the name of the output file where you will write the computed statistics. Example input and output files are posted on the course website. For example, here is a valid command line to your program:

```
golf_statistics.exe top_scores.txt top_results.txt
```

We have provided you with a few golf score datasets. The original data was taken from an online resource: `http://sports.espn.go.com/golf/schedule?year=2007`. The format has been modified slightly to ease parsing. The results from a tournament are grouped together and begin with the keyword "`TOURNAMENT`", the year, and the tournament name. For example:

```
TOURNAMENT 2007 PGA_CHAMPIONSHIPS
```

The players who entered the tournament and their results follow (in the format shown above & below). Here's an example of a player who only played the first 2 rounds at the 2007 US Open because he did not make the cut.

```
Phil Mickelson     CUT       74 77  -  -       151       $0
```

Instead of a valid par score the string "`CUT`" appears and uncompleted rounds are recorded with the "`-`". If a player's total score for the tournament is equal to par, the string "`E`" appears. The string "`WD`" or "`DQ`" appears if the player withdrew or was disqualified. The list of players who entered the tournament ends when the next "`TOURNAMENT`" keyword is encountered or when the end of the file is reached. *Note: white space is not significant in the input file format. Do not make any unnecessary assumptions about formatting.*

Statistics Collected and Output

The output will be in *three parts*. First is a table with the players sorted alphabetically (by last name, then by first name). Each row of the table should include the player, the number of tournaments entered, and the total prize money won. For example, given an input file with these player results:

```
TOURNAMENT 2007 US_OPEN
Angel Cabrera              +5      69      71      76      69      285     $1,260,000
Tiger Woods                +6      71      74      69      72      286     $611,336
Jim Furyk                  +6      71      75      70      70      286     $611,336
TOURNAMENT 2007 PGA_CHAMPIONSHIPS
Tiger Woods                -8      71      63      69      69      272     $1,260,000
Woody Austin               -6      68      70      69      67      274     $756,000
Ernie Els                  -5      72      68      69      66      275     $476,000
TOURNAMENT 2006 US_OPEN
Geoff Ogilvy               +5      71      70      72      72      285     $1,225,000
Jim Furyk                  +6      70      72      74      70      286     $501,249
Phil Mickelson             +6      70      73      69      74      286     $501,249
TOURNAMENT 2006 PGA_CHAMPIONSHIPS
Tiger Woods               -18      69      68      65      68      270     $1,224,000
Shaun Micheel             -13      69      70      67      69      275     $734,400
Luke Donald               -12      68      68      66      74      276     $353,600
```

Your program will produce a table similar to this:

```
Player         Num Tournaments    Total Earnings
-----------------------------------------------
Woody Austin                 1          $756,000
Angel Cabrera                1        $1,260,000
Luke Donald                  1          $353,600
Ernie Els                    1          $476,000
Jim Furyk                    2        $1,112,585
Shaun Micheel                1          $734,400
Phil Mickelson               1          $501,249
Geoff Ogilvy                 1        $1,225,000
Tiger Woods                  3        $3,095,336
```

The formatting shown above is an example. Your output may be formatted differently, as long as it satisfies the description above. The second part of the output is a table where the players are sorted by their average par score per hole. This is computed by summing the total par of all *completed tournaments* and dividing it by the total number of holes in their completed tournaments. Remember, the number of holes in a tournament is $18 * 4 = 72$.

```
Player                    Par Sum   Total Holes   Par Average
-----------------------------------------------------------
Shaun Micheel                 -13            72        -0.181
Luke Donald                   -12            72        -0.167
Tiger Woods                   -20           216        -0.093
Woody Austin                   -6            72        -0.083
Ernie Els                      -5            72        -0.069
Angel Cabrera                  +5            72         0.069
Geoff Ogilvy                   +5            72         0.069
Jim Furyk                     +12           144         0.083
Phil Mickelson                 +6            72         0.083
```

Again, the formatting shown above is an example of how to present this information. Players who did not complete at least one tournament will not appear in the table.

The third and final part of the output is a chance for you to be creative. Brainstorm a new interesting statistic that can be calculated from this data. Try to come up with something that makes use of the year and/or tournament name information. A simple example would be to output the players ordered by their total winnings from all US Open tournaments. A more complex example would be to compute the standard deviation of each player's round scores per year and output the players whose consistency is improving (the standard deviation of the scores decreases each subsequent year).

Extra credit will be awarded to particularly interesting statistics that require clever programming. The most important task for this part of the assignment is to write a concise description (< 100 words) of your new statistic. Put this description in your *plaintext* file named `README.txt` along with any other notes for the grader. Be sure to tell the grader which dataset best demonstrates your new statistic. Feel free to create your own dataset and include it with your submission.

Useful Code

To control the formatting of your tables, you'll want to read up on the various iomanipulators: `std::setw(int)`, `std::setprecision(int)`, `std::fixed`, `std::left`, etc. And don't forget about the `sort` function that can be used to order the contents of a `vector`.

To help you parse the strings that represent the prize money, we provide the function `parse_money` that takes in the string "`$12,345`" and returns the integer `12345`. The function `print_money` does the opposite. Similarly, the functions `parse_rounds`, `parse_par`, and `print_par` will be handy. You do not need to fully understand the details of these functions in order to use them as *black boxex*. With this code and the `>>` stream function for strings used in lecture you can implement all of the parsing necessary for this assignment.

Program Requirements & Submission Details

Your program should involve the definition of *at least one class* that has its own .h and .cpp files, named appropriately. Initially you should focus on the small dataset shown in this handout which does not include example data for players who were cut, withdrawn, or disqualified from a tournament. Once that is working you can extend your solution to handle the big test case.

Do all of your work in a folder named `hw2` inside of your CSII homeworks directory. Use good coding style when you design and implement your program. Be sure to make up new test cases and don't forget to comment your code! Please use the provided template `README.txt` file for any notes you want the grader to read. **You must do this assignment on your own, as described in the "Academic Integrity for Homework" handout. If you did discuss the problem or error messages, etc. with anyone, please list their names in your `README.txt` file.** When you've finished writing, testing, debugging, and commenting your code, prepare and submit your assignment as instructed on the course webpage. Please ask a TA if you need help preparing your assignment for submission or if you have difficulty writing portable code.