CSI PA09

Tracking the Greats of the NBA

Code due by Monday, April 14th at 11:00 AM

Paperwork due the same day at the start of class

The purpose of this assignment is to give you more practice with functions, files, lists, and tuples while using a real data set populated with career data from NBA players!

Remember, before you begin this project, review the new design document and function commenting styles that I expect. Please create the design documents and function comments **before** beginning the actual coding.

Getting Started

1. Go to http://www.databasebasketball.com/stats_download.htm
2. Download databaseBasketball_2009_v1.zip
3. Unpack the zip file (most machines will do this when you double-click the file)
4. We will only do statistics based on "player_career.csv". So copy this file into the same directory where you will be writing your python program for this assignment

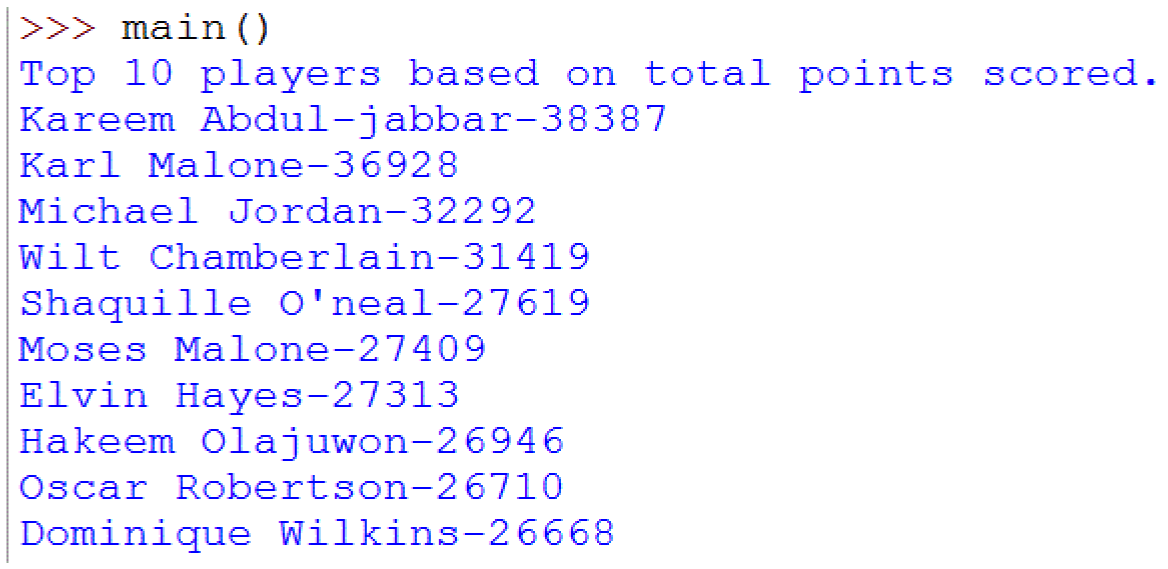
The format of this file is easy to understand. Open the file by right clicking on the file and selecting "Open With" and selecting a text editor of some kind like Notepad++ or Wordpad. The first line tells you the names of all the columns (To understand the meanings of each of the abbreviations, look at the page: http://www.databasebasketball.com/about/aboutstats.htm). After that, each line's data corresponds to one player's career statistics. Each field is separated by a comma.

Notice that, in addition to the very first line, which is the header information, there appears to be a blank line followed by a line of "garbage" at the bottom of the file. This will become an issue later in the assignment.

Your Assignment, Part 1

1. Begin by creating a file called pa09.py.
2. In this file create a function called readData(). This function should:
    - take no parameters. It will be specialized for the file that you are using.
    - open the file “player_career.csv”
    - each line in this data file should be read as a list, and then each line list should be then appended to a master data list. (Hint: Wasn't there a file method that would do this for you?)
    - you should ignore the "garbage" rows from the player_career.csv file and not add those lines to your list.
    - close the file "player_career.csv"
    - return the master data list, which is a list of line lists. (In other words, you should return a list of length 4051. Each element in the list is itself a list representing the career stats of a single player)
3. Write a function called points(). This function should:
    - take in one parameter - a list of player data (this will be the data you "loaded" and "cleaned" using readData())
    - For each player in the data list you should create a tuple consisting of the points the player earned and the player's name)
    - Each of these tuples should be appended to a separate list.
    - When you are done processing the input data you should return this new list of tuples.
4. Write a function called main(). This function should:
    - take no parameters
    - invoke readData() to get the data about all of the players from our stat file
    - send the list returned by readData() to points()
    - sort the list returned by points() to identify and print information about the ten players from this data file who scored the most points during their careers.

When you are all done you should be able to load and invoke main() from the shell and get a response that looks like the following:



```
>>> main()
Top 10 players based on total points scored.
Kareem Abdul-jabbar-38387
Karl Malone-36928
Michael Jordan-32292
Wilt Chamberlain-31419
Shaquille O'neal-27619
Moses Malone-27409
Elvin Hayes-27313
Hakeem Olajuwon-26946
Oscar Robertson-26710
Dominique Wilkins-26668
```

Your Assignment, Part 2

1. Write additional functions called minutes() and rebounds(). These functions should work just like the points() function does. That is, they should take in the main data list, process each player in the list to make appropriate tuples, and return a list of these tuples.
2. You should also modify your main() method so that it prints the ten players with the largest number of **each** of these items. When I run this function I should see a long screen dump of the top 10 players in each of these three statistic categories:

```
>>> main()
Top 10 players based on total points scored.
Kareem Abdul-jabbar-38387
Karl Malone-36928
Michael Jordan-32292
Wilt Chamberlain-31419
Shaquille O'neal-27619
Moses Malone-27409
Elvin Hayes-27313
Hakeem Olajuwon-26946
Oscar Robertson-26710
Dominique Wilkins-26668

Top 10 players based on total minutes played.
Kareem Abdul-jabbar-57446
Karl Malone-54852
Elvin Hayes-50000
Wilt Chamberlain-47859
John Stockton-47765
Reggie Miller-47622
Gary Payton-47123
John Havlicek-46471
Robert Parish-45712
Moses Malone-45071

Top 10 players based on total rebounds.
Wilt Chamberlain-23924
Bill Russell-21620
Kareem Abdul-jabbar-17440
Elvin Hayes-16279
Moses Malone-16212
Karl Malone-14967
Robert Parish-14715
Nate Thurmond-14464
Walt Bellamy-14241
Wes Unseld-13769
```

Your Assignment, Part 3 - The Efficiency Statistic

Each of the above statistics is interesting, but it only tells us how good a player is at one specific statistic. How do many NBA coaches quickly evaluate a player's overall game performance? They check his efficiency. This statistic is something like the QB passer rating we calculated earlier in the course. It is a calculation that tries to assign a number to how "well" a player played the game. Higher numbers mean a better performance from that player.

NBA.com evaluates all players based on the efficiency formula indicated below (and shown on the aboutstats.htm page). In this project, we will follow this efficiency formula. Since we are not evaluating a player based on one game, we need to divide the total efficiency by the number of games the player played. So the formula is:

$$Efficiency = \frac{(pts + reb + asts + stl + blk) - ((fga - fgm) + (fta - ftm) + turnover)}{gp}$$

The abbreviations on the right hand side of the equation correspond to the fields in the statistics file. Again, you can check out the the meanings of each of the abbreviations at: http://www.databasebasketball.com/about/aboutstats.htm

1. Create a function called efficiency(). This function should behave very similarly to the functions you wrote in parts 1 and 2 in that it should:
   - take in one parameter - a list of player data (this will be the data you "loaded" and "cleaned" using readData())
   - For each player in the data list you should create a tuple consisting of the player's career long efficiency and the player's name)
   - Each of these tuples should be appended to a separate list.
   - When you are done processing the input data you should return this new list of tuples.
   - Now modify main() so that it uses this as it has the prior functions.

```
Top 10 players based on total efficiency.
Wilt Chamberlain-41.4976076555O239
Spencer Haywood-37.8095238095238l
Artis Gilmore-33.126l9047619047
Julius Erving-32.26044226044226
Bill Russell-31.7061266874351
Oscar Robertson-31.614423076923078
Bob Pettit-31.108585858585858
Kareem Abdul-jabbar-30.92756410256 41
Connie Hawkins-30.299145299145298
Larry Bird-29.767001114827202
```

Bonus

1. Write a function called recordTopHundred(). This function should:
    - take no parameters
    - invoke readData() to get the data about all of the players from our stat file
    - send the list returned by readData() to efficiency()
    - sort the list returned by efficiency()
    - select the top 100 players from this sorted list and **write these players** to a file entitled "top100.txt"

Helpful Hints:

1. Remember the split() function, which takes as an argument the character to split on, and returns a LIST of STRINGS
2. Pay attention to the type of data you are working with. Don't forget to convert Strings to numbers or vice versa as needed:
3. To create a tuple, remember you just need () and a comma, so a 2-item tuple would be:

    ```
    myTuple = (x,y)
    ```

    To append this tuple to a list you can just say myList.append(myTuple). Then to access the different items in the tuple you index into the list twice, so for example if you appended the above tuple as the first item in a list:
    myList[0][0] would return x
    myList[0][1] would return y

4. Since there are so many fields, do some testing (E.g. output some parsed data) to make sure that you get the correct data.
5. List’s sort function and reverse function should be useful.
    - `myList = [ [3,2], [1,2], [2,5]]`
    - `myList.sort() # myList will be [ [1,2], [2,5], [3,2]]`
    - `myList.reverse() # myList will be [ [3,2], [2,5], [1,2]]`

Final Submission

To upload your homework for grading, log on to eLearning, select this class, and navigate to the "Assignment Submissions" area. Click on the "Programming Assignment 9" folder and upload the python file in its designated location.

In addition to this, you should print paper copies of your code and design document. Please submit these stapled printouts in the following order in class the day the assignment is due:

- design document
- pa09.py