CSI PA08

Tracking the Greats of the NBA - Design Version

Paperwork due by Monday, April 7th at 11:00 AM

The purpose of this assignment is to give you more practice with creating top-down design documents and breaking large problems down into managable, small chunks.

Create a Top-Down Design Document

I want you to eventually create a program that parses through a very large CSV file full of real NBA player information and prints out a number of statistics. However, for this assignment, I ***just want you to create a heirarchical solution tree*** to be reviewed by your Program Manager (namely, me.) Your heirarchical solution tree should look similar to the trees in the presentation in Session Twenty Nine and drawn or graphed on a sheet of paper.

This assignment ***looks*** simple, but can actually be very hard. I don't want you to stop making modules in your tree until you have changed all your abstract steps into concrete steps. Remember, concrete steps can be translated ***directly***into Python code, where abstract steps need to be broken down and thought about. For example, the step *determine top 10 players* is still an abstract step, because we still don't know how to do that in Python. However, the step *read input file into list of lines* is a concrete step, because we can implement that step directly with the method fileObject.readLines().

The Program Problem Statement

Specifically, I'll want your program to print out the top 10 NBA **player names** based on the following:

- Top 10 players based on total points scored
- Top 10 players based on total rebounds
- Top 10 players based on total minutes played
- Top 10 players based on efficiency

The efficiency measurement 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

---

Input File

To see what kind of input file you'll be dealing with, please do the following:

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.

Hints

We have to somehow read the information from the file and sort it, depending on what we are sorting for (points scored, rebounds, etc.).

Remember that lists can be sorted using list methods. However, if we just put all the points scored in a list and sorted, we would have no corresponding player names -- we would just have a list of sorted numbers! Instead, we might want to create a tuple consisting of (points scored, first name, last name) for each line in the file and append that tuple to a big list. Putting the points scored and player names in tuples helps us keep data together that we want to stay together. When we have our giant list of tuples, we can call a sort on that giant list of tuples, and it will sort all the tuples based on the first field in each tuple, which was points scored! Finally, we can iterate through that each tuple in the list (up to 10) and print out the points scored and the player name.

Final Submission

This time, you won't be uploading your homework onto eLearning. Instead, I want you to turn in the following as stapled paperwork in class the day the assignment is due with your name on the top:

- hierarchical solution tree