CSCI 1100 — Computer Science 1 Homework 5
Lists, Loops and Files

Overview

This homework is worth **90 points** toward your overall homework grade, and is due Thursday, October 26, 2017 at 11:59:59 pm. It has two parts, each worth 45 points. Please download `hw5_files.zip`. and unzip it into the directory for your HW5. You will find multiple data files and a utility module, `hw5util.py` that will help with Part 1. We also provide you with a simple test program `hw5part1_test.py` that demonstrates how to use the utility module to read in the Part 1 data files.

The goal of this assignment is to work further with loops, lists and files. You are already familiar with loops and lists. Files will be covered on Monday.

In Part 1 of the homework, you will use the utility module to read data from a file as shown in the test program and you are required to use a double (i.e. nested) loop. In Part 2 of the homework, you will need to read and process the data yourself, but cannot use a double loop. You may, of course, use as many single loops as you like. Failure to follow these guidelines will lose you points.

Test your programs with smaller data files first – use `predictions_short.txt` for Part 1 and `temp_data_short.txt` for Part 2. This will allow you to more easily track if your programs are working correctly or not before you try the full data sets files. In particular, the full temperature file is quite large as you will find out.

As always, make sure you follow the program structure guidelines. You will be graded on program correctness as well as good program structure.

Remember as well that we will be continuing to test homeworks for similarity. So, follow our guidelines for the acceptable levels of collaboration. You can download the guidelines from Piazza in the resources section if you need a refresher.

Can you predict the future ... of soccer?

Note that this part is intended as an exercise for double loops. You must use double loops for full credit on this part!

A good friend of mine runs a soccer league before all major tournaments. Instructions for the league are given at the end of this section for your entertainment. All participants have to guess all games before the tournament starts and then when it ends, players are scored based on how well they guessed. The scoring is simple:

- Guessing the outcome of the game correctly (win/lose/draw) is 2 points.
- Guessing the correct number of goals for either team is worth 1 point each.
- The score for each player is the total score she gets for her guesses for all the games.
- The player who has the highest score wins the league.

Your job in this homework is to ask the user for the name of a prediction file containing the guesses by the league participants for a particular soccer tournament. Use the utility module we give you to read the file and retrieve the data for the different players. As **Dragnet** would say, "The

predictions you are about to read are real. The names have been changed to protect the innocent." (Are any of you familiar with Dragnet?) Your program will then:

- Compute and print the scores for each player
- Print the winner (or winners if there is a tie)
- Compute and print the total score for each game across all the players
- Print the hardest game or games to predict. This is the game or games with the lowest total score.

Note that the first two and the last two requests are almost identical, but iterate differently.

Here is some more detail. The utility module we give you will read the prediction data into three lists. For example, consider the following test program:

```
import hw5_util

if __name__ == "__main__":
    games, players, predictions = hw5_util.read_predictions('predictions_short.txt)
    print ("Games:")
    print (games)
    print ("Players:")
    print (players)
    print ("Predictions:")
    print (predictions)
```

The `read_predictions` function returns three lists `games`, `players`, and `predictions`:

`games` is a list of games, where each game is given by the tuple:

`(game_no, game_date, group, team1, team2, team1_score, team2_score))`.

`players` is the name of the players.

`predictions` is a list of lists where each sub-list contains the predicted scores for the games above from one of the players. The scores are all given by tuples of `(team1_score, team2_score)`.

The above program will print for this file the following output:

```
Games:
[(1, '10-Jun', 'A', 'France', 'Romania', 2, 1), (2, '11-Jun', 'A', 'Albania',
    'Switzerland', 0, 1), (3, '11-Jun', 'B', 'Wales', 'Slovakia', 2, 1), (4, '11-Jun',
    'B', 'England', 'Russia', 1, 1)]
Players:
['Eleven', 'Dustin']
Predictions:
[[(2, 1), (0, 1), (1, 1), (1, 1)], [(1, 0), (0, 1), (0, 1), (0, 1)]]
```

According to this data, for example, the first game was `France vs Romania`. Here is the real result and the predictions (the real result is given by `games[0][-2],games[0][-1]`):

	Real	Eleven	Dustin
France vs Romania	(2, 1)	(2, 1)	(1, 0)

Remember in soccer, a team wins if it has higher score (so France won the above game). If the number of goals are equal, then this is a draw.

In this case, both Eleven and Dustin guessed that France would win, so they will each get 2 points. However, Eleven also guessed the goals for France and for Romania correctly, so she will get an additional 2 points – 1 point each for the 2 correct scores. At this point of the competition, Eleven has 4 points and Dustin has 2.

First, calculate the scores for each player. Then print out the player(s) with the highest score.

Next, find the total score for each game and then print out the game(s) with the lowest scores.

Here are some example outputs of this program (Note that in the `Player points` and `Game points` lists, players are formated to 10 characters, scores to 4 and games to 30 characters):

```
Enter the filename => predictions_short.csv
predictions_short.csv
Player points:
Eleven    :  13
Dustin    :   8

Winner(s): (max points: 13)
Eleven

Game points:
France vs Romania             :   6
Albania vs Switzerland        :   8
Wales vs Slovakia             :   2
England vs Russia             :   5

Hardest games(s) (min points: 2)
Wales vs Slovakia
```

```
Enter the filename => predictions.csv
predictions.csv
Player points:
Eleven    :  59
Dustin    :  57
Lucas     :  57
Nancy     :  52
Mike      :  59

Winner(s): (max points: 59)
Eleven
Mike

Game points:
France vs Romania             :  17
Albania vs Switzerland        :  19
Wales vs Slovakia             :   8
England vs Russia             :  12
Turkey vs Croatia             :  13
Poland vs Northern Ireland    :  16
Germany vs Ukraine            :  18
Spain vs Czech Rep            :  15
```

```
Rep. Ireland vs Sweden        :   3
Belgium vs Italy              :   0
Austria vs Hungary            :   3
Portugal vs Iceland           :   2
Russia vs Slovakia            :   4
Romania vs Switzerland        :  13
France vs Albania             :  17
England vs Wales              :  11
Ukraine vs Northern Ireland   :   2
Germany vs Poland             :   2
Italy vs Sweden               :  13
Czech Rep vs Croatia          :   3
Spain vs Turkey               :  14
Belgium vs Rep. Ireland       :  15
Iceland vs Hungary            :   6
Portugal vs Austria           :   4
Switzerland vs France         :   7
Romania vs Albania            :   3
Slovakia vs England           :  10
Russia vs Wales               :   0
Northern Ireland vs Germany   :  13
Ukraine vs Poland             :   3
Croatia vs Spain              :   2
Czech Rep vs Turkey           :   0
Iceland vs Austria            :   5
Hungary vs Portugal           :   0
Sweden vs Belgium             :  10
Italy vs Rep. Ireland         :   1

Hardest games(s) (min points: 0)
Belgium vs Italy
Russia vs Wales
Czech Rep vs Turkey
Hungary vs Portugal
```

Both of the test files are given to you. We will test on Submitty with a different input file.

When you have tested your code, please submit it as `hw5Part1.py`. You must use this filename, or Submitty will not be able to run and grade your code.

To end this section, here are the official rules of entry for this league (for fun):

The participants are allowed to use any conceivable method, including, but not limited to: gut feelings, expert opinions, insider information, computer simulation, Monte Carlo simulation, match fixing (for Juventus fans), clairvoyance, Jedi mind trick, time machine, divine revelation, and divine intervention. However, methods that involve tampering with, harassing, and intimidating the organizers are strongly discouraged.

The organizers reserve the right to change the rules for noble causes, such as universal justice, survival of mankind, etc.

GRAND PRIZE: Although last year we doubled the prize, we inexplicably continue receiving complaints from past winners who did not receive anything for winning the league. This year, we prove our continual commitment to improving this league by tripling the value of the prize.

Temperatures in Troy are still quite warm....

Just in time for these extremely warm October days, we have a climate based homework. The data for this homework comes from NOAA (National Oceanic and Atmospheric Administration). It is temperature data from Troy, NY from July 1956 till September 2017, measured at the Hudson River Lock and Dam (you can see it for yourself here: `https://www.google.com/maps/place/4245'07.1"N+7341'15.5"W`).

The data is given to you in a file called `temp_data.txt` along with a smaller test file named `temp_data_short.txt`. Each row is temperature data for a specific month. The items in the rows are separated by commas. Note that some fields are empty, because there was not enough data for this field in that month. Whenever there is a missing value, you must not use it in your calculations (i.e. do not assume it is zero). For example, the row:

```
TROY LOCK AND DAM NY,2015-04,,,,,1.95,,,,
```

is missing lots of data fields. If you were to split this line on a comma, you would get:

```
['TROY LOCK AND DAM NY', '2015-04', '', '', '', '', '', '1.95', '', '', '', '']
```

which is missing values for fields 2,3,4,5,6,8,9,10,11. You can check a string is empty with the length function or by comparing to `""`. Here are the field definitions for this file:

Id	Field	Description
0	STATION NAME	Name of weather station (this is the same for all of our data)
1	DATE	Year and month, ex. 2017-09
2	DX32	Number of days with minimum temperature <= 32 degrees (F)
3	DX90	Number of days with maximum temperature >= 90 degrees (F)
4	EMNT	Lowest daily minimum temperature for the month/year (F)
5	EMSN	Highest daily snowfall in the month/year (in)
6	EMXT	Highest daily maximum temperature for the month/year (F)
7	PRCP	Total Monthly Precipitation (in)
8	SNOW	Total Monthly Snowfall (in)
9	TAVG	Average Monthly Temperature (F)
10	TMAX	Monthly Maximum Temperature (F)
11	TMIN	Monthly Minimum Temperature (F)

Write a program that asks the user for the file name and the index of a month. Your program should then output the following information:

- The earliest and latest recorded average monthly temperature for this month in the given data file.
- The average temperature for this month over all the data (average of `TAVG`) for months with non-empty data
- The lowest monthly min (min of `TMIN`) and highest month max (max of `TMAX`) ever recorded for this month and which year (or years if multiple years had the same value) each one was recorded
- Print a histogram of average temperature values over ten year intervals for that month, print the year range and a star for each degree.

Note that the first line of the file contains header information defining the fields. You need to read and discard that line before beginning to process the data.

Here is a sample output of your program (using the shorter data file):

```
Filename => temp_data_short.txt
temp_data_short.txt
Month => 4
4
Earliest recorded average 47.90 in 2014
Latest recorded average 52.80 in 2017
Average temperature: 49.20
Lowest min value recorded: 35.20 in year(s): 2016
Highest max value recorded: 63.10 in  year(s): 2017
Histogram of average temperature
2014-2017: ************************************************
```

Here is another potential output (using the much larger data file):

```
Filename => temp_data.txt
temp_data.txt
Month => 2
2
Earliest recorded average 29.40 in 1957
Latest recorded average 33.30 in 2017
Average temperature: 24.94
Lowest min value recorded: -0.40 in year(s): 2015
Highest max value recorded: 44.10 in  year(s): 2016
Histogram of average temperature
1957-1966: **********************
1967-1976: **********************
1977-1987: ************************
1988-1997: ************************
1998-2007: *************************
2008-2017: *************************
```

You are welcome to use any method you want for this problem. One particularly useful thing to learn for this homework is how sorting of lists works in Python. If you have a list of lists or list of tuples, then Python will first sort by the first element of each sublist or tuple and then by the second element, etc. Here is an example:

```
>>> x = [(2,'d'), (1,'c'), (3, 'a'), (1,'b')]
>>> x.sort()
>>> x
[(1, 'b'), (1, 'c'), (2, 'd'), (3, 'a')]
>>> x.sort(reverse=True)
>>> x
[(3, 'a'), (2, 'd'), (1, 'c'), (1, 'b')]
```

We do not require you to use this method, but it can help.

When you have tested your code, please submit it as `hw5Part2.py`. You must use this filename, or Submitty will not be able to run and grade your code.