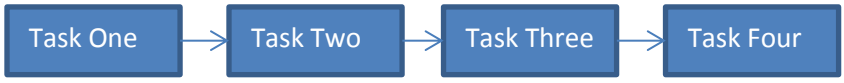
HW 6+7 Advanced: CS 110X C 2013

Note: This homework (and all remaining homework assignments) is a **partner homework** and must be completed by each partner pair. When you complete this assignment, you must not share your answers with any other student. Only one person from a partner pair needs to submit the assignment.

Only student teams confident with their Python skills should attempt this homework. Should you do so, it will become your grade for both Homeworks 6 and 7 (thus it is worth 10% of your grade). Only attempt this homework if both programming partners agree to tackle this assignment.

For this homework you will be tackling a more substantial problem that you are to break up in to a number of sub-steps. Lab4 was designed to get you thinking along these lines, but now it is time to put that theory into practice.

One of the most common structures in business applications is a linear "pipeline" of information.



Each task produces output for the subsequent task, and each one independently realizes a specific sub-step of the overall application. To complete HW6+7 you are to complete the implementation of a number of functions. Each one can be tested independently, and once you have them all done, the final larger application is realized. I have provided two functions to help "set the stage" for you.

Let me show you the code for main, which is the entry point into the program:

```
# All input data comes from this file.
dataSet = '1600_baseball_players_2008_thru_2012.csv'

# Data created from this program appears in the following CSV file
outputFile = 'outputSet.csv'

def main():
    """Allow user to experiment with a data set to uncover data and trends"""
    # Retrieve headers from the data set
    headerList = retrieveHeader()

    print ("Input file contains " + str(numRecords()) + " records.")

    # Find the index locations of the user fields of interest
    indices = selectColumns(headerList)
    if len(indices) == 0:
        return

    # Determine what multiple of std to use for threshold
    multiple = input ("How many stdevs is your threshold? (i.e., 1) ")
```

```
    # Compute the threshold values for the given columns.
    thresholds = computeThresholds(indices, multiple)

    count = identifyPlayers(headerList, indices, thresholds)
    print (str(count) + " player data written to " + outputFile)
```

This method retrieves the list of column headers from the `dataSet` file and tells the user how many records are in the file. Then the user is asked to select a number of these columns in which he would like to find outliers, namely, those records whose values that are higher than the average for that column plus some multiple of the standard deviation for that column. The program identifies the records which satisfy the user's criteria and then the values are written to `outputFile`.

Q1	`numRecords()`

This function determines the number of available records in the data set. It returns an `int` value.

Sample Output

```
>>> numRecords()
1600
```

The name of the file to use is stored in the module variable `dataSet`. Be sure to close the file after reading its contents.

Q2	`retrieveHeader()`

This Function retrieves the header row from `dataSet` and returns a list of strings representing the fields. No entry should have `'\n'` in its value. Be sure to close the file after reading its contents.

Sample Output

```
>>> retrieveHeader()
['Rk', 'Player', 'HR', 'Year', 'Age', 'Tm', 'Lg', 'G', 'PA',
'AB', 'R', 'H', '2B', '3B', 'RBI', 'BB', 'IBB', 'SO', 'HBP',
'SH', 'SF', 'GDP', 'SB', 'CS', 'BA', 'OBP', 'SLG', 'OPS', 'Pos']
```

Q3	`computeStdev(columnIndex)`

This function processes `dataSet` to compute the `float` standard deviation of the values found in the designated column. It returns a single `float` value.

In the existing `dataSet`, four of the columns – Player (1), Tm (5), Lg (6), Pos(28) – are string values, so you can assume that `columnIndex` will never be one of these values.

Sample Output

```
>>> computeStdev(2)
9.0378639060551897
>>> computeStdev(4)
4.1176053037536295
```

You will need to import numpy for this computation (as you did on an earlier homework). Be sure to close dataSet after reading its contents.

Q4	`computeAverage(columnIndex)`

This function processes `dataSet` to compute the `float` average of the values found in the designated column. It returns a single `float` value.

In the existing `dataSet`, four of the columns – Player (1), Tm (5), Lg (6), Pos(28) – are string values, so you can assume that `columnIndex` will never be one of these values.

Sample Output

```
>>> computeAverage(2)
13.988125
>>> computeAverage(4)
28.80875
```

You will need to import numpy for this computation (as you did on an earlier homework). Be sure to close dataSet after reading its contents.

Q5	`computeThresholds(indices, multiple)`

This function takes a list of `n` indices and a multiple `float`. It returns a list of `n` threshold float values. Each threshold value is computed by calculating the average and stdev for a given column index (drawn from `indices`); then the threshold = average + multiple * stdev.

Sample Output

```
>>> computeThresholds([2,4],1)
[23.02598890605519, 32.926355303753631]
>>> computeThresholds([2,4],1.5)
[27.544920859082787, 34.985157955630442]
```

Given a list of column indexes and the multiple of stdevs, compute threshold values for the assigned columns. Make use of the `computeAverage(columnIndex)` and `computeStdev(columnIndex)` functions that you defined earlier.

Q6	`selectColumns(headerList)`

This function allows the user to select a number of fields as contained in `headerList` and returns a list of their respective index locations in `headerList`.

Here, for example, the function was given a list of six fields (`a` through `f`). The user selected field `b` and `d`. However, when he typed `X`, he was told that field does not exist. Also, when he tried to select the same field again, he was told that he had already selected that field. When the user is done selecting fields, he pressed ENTER and the Python Shell shows that the indices of the selected fields are `3` and `1` (which is the order that the user had selected).

If the user presses ENTER (and by doing so selects no fields) then the empty list `[]` is returned.

Sample Output

```
>>> selectColumns(['a', 'b', 'c', 'd', 'e', 'f'])
There are 6 columns:
a     b     c     d     e     f
Select a column to include (or press ENTER to quit) d
Select a column to include (or press ENTER to quit) b
Select a column to include (or press ENTER to quit) X
  ** That column does not exist **
Select a field to include (or press ENTER to quit) d
  ** You have already selected that column **
Select a field to include (or press ENTER to quit)
[3, 1]
```

Note that this function does not need to open `dataSet` for reading.

Sample Output

Once you complete the assignment, here is a transcript of a sample run. What if you want to find whether any player has hit more than 3 stdevs above the average of homeruns for the data set.

```
>>> main()
Input file contains 1600 records.
There are 29 columns:
Rk    Player      HR    Year  Age   Tm    Lg    G     PA    AB    R     H
2B    3B    RBI   BB    IBB   SO    HBP   SH    SF    GDP   SB    CS    BA
OBP   SLG   OPS   Pos

Select a column to include (or press ENTER to quit) HR
Select a column to include (or press ENTER to quit)
How many standard deviations is your threshold? (i.e., 1) 3
12 player data written to outputSet.csv
```

And the output file contains the following information for twelve players.

Name	HR
Jose Bautista	54
Ryan Howard	48
Albert Pujols	47
Prince Fielder	46
Ryan Howard	45
Miguel Cabrera	44
Mark Reynolds	44
Curtis Granderson	43
Josh Hamilton	43
Jose Bautista	43
Edwin Encarnacion	42
Albert Pujols	42
Threshold	41.10172

Let's try another run. What if you want to find the players who hit a lot of homeruns, but also strike out a lot of times. Here we seek player data whose values are greater than 2 stdevs above average:

```
>>> main()
Input file contains 1600 records.
There are 29 columns:
Rk    Player      HR    Year  Age   Tm    Lg    G     PA    AB    R     H
2B    3B    RBI   BB    IBB   SO    HBP   SH    SF    GDP   SB    CS    BA
OBP   SLG   OPS   Pos

Select a column to include (or press ENTER to quit) HR
Select a column to include (or press ENTER to quit) SO
Select a column to include (or press ENTER to quit)

How many standard deviations is your threshold? (i.e., 1) 2
21 player data written to outputSet.csv
```

And the file contains:

Name	HR	SO
Ryan Howard	48	199
Ryan Howard	45	186
Mark Reynolds	44	223
Curtis Granderson	43	195
Josh Hamilton	43	162
Adam Dunn	41	222
Curtis Granderson	41	169
Adam Dunn	40	164
Matt Kemp	39	159
Carlos Pena	39	163
Adam Dunn	38	199
Adam Dunn	38	177
Mark Reynolds	37	196
Dan Uggla	36	156
Jason Bay	36	162
Jayson Werth	36	156
Jay Bruce	34	155
Giancarlo Stanton	34	166
Chris Davis	33	169
Ryan Howard	33	172
Jack Cust	33	197
Threshold	32.06385281	152.626652

How To Get Started On This Assignment

A template HW67_Template.py file is provided to you. Use the same data as for Lab 4.

The following functions are independent and can be completed without depending on any other functions. I would tackle this problem in the following order:

- `numRecords()` February 7 2013
- `retrieveHeader()` February 7 2013
- `computeAverage(columnIndex)` February 8 2013
- `computeStdev(columnIndex)` February 8 2013
- `selectColumns(headerList)` February 10 2013

With these done, the following function depends on computeStdev and computeAverage, which it invokes directly (I'll leave it to you to see how).

- `computeThresholds(indices, multiple)`

Submit your HW67.py file using the web-based turnin system. As we have mentioned in class, only one of the team members needs to submit the assignment. But just make sure that something gets submitted!

Also, if you are going this route, please email cs110x-staff@cs.wpi.edu about your intentions.