PROBLEM 2 : (***Short code segments (32 pts)***)

For each of the following problems, assign the variable **result** to a python expression that calculates the answer. That means if we changed the list(s) or dictionary given, your code would still calculate the correct answer.

Unless indicated, each of these must be written in one line for full credit. For partial credit you can solve the problem in more than one line.

Here is an example.

The variable result should calculate the list of words from vehicles that have the letter 'a' in their word. Assume each string in the list is one word that is lowercase.

Using the list vehicles below, result would calculate the list: ['train', 'airplane', 'car', 'longboard']

```
vehicles = ['train', 'airplane', 'car', 'truck', 'bike', 'longboard']
result =
```

ANSWER:

```
result = [w for w in vehicles if 'a' in w]
```

PART A (4 pts)

The variable **result** should calculate the list of words from the list food that have the letter 'a' exactly once in their word. Assume each string in the list is one word that is lowercase.

Using the list food below, result would calculate the list: ['cake']

```
food = ['cookies', 'cake', 'brownies', 'popsicles', 'oatmeal']
result = _______________________________________________
```

PART B (4 pts)

The variable result should calculate the list of words from the list words that have the letter 'a' in the first half of the word. Assume each string in the list is one word that is lowercase, and each word is of even length.

Using the list words below, result would calculate the list: ['flavor', 'strawberry', 'hand', 'an', 'abacas']

```
words = ['sugary', 'stripe', 'flavor', 'strawberry', 'hand', 'an', 'goat',
'abacas']
result = _______________________________________________
```

PART C (4 pts)

Here you are given two list of words. Assume the words in the lists are all lowercase letters.

The variable result should calculate the list of unique words from list1 and list2 that appear in both lists.

Using the lists list1 and list2 below, result would calculate the list: ['green', 'blue', 'red', 'yellow']

The order of the resulting words does not matter.

```
list1 = ['red', 'blue', 'yellow', 'brown', 'green', 'red', 'orange']
list2 = ['blue', 'white', 'red', 'green', 'yellow', 'black', 'purple']
result = ______________________________________________
```

PART D (4 pts)

Here you are given two list of words. Assume the words in the lists are all lowercase letters.

The variable result should calculate the list of unique words from list1 and list2 that appear in only one of the lists.

Using the lists list1 and list2 below, result would calculate the list: ['orange', 'white', 'black', 'brown', 'purple']

The order of the resulting words does not matter.

```
list1 = ['red', 'blue', 'yellow', 'brown', 'green', 'red', 'orange']
list2 = ['blue', 'white', 'red', 'green', 'yellow', 'black', 'purple']
result = ______________________________________________
```

PART E (4 pts)

Here you are given two lists: a is a list of words, all lowercase, and b is a list of integers. The two lists are the same length. An integer in list b represents a rating of the word in list a that is in the same index position. For example, in lists a and b below, note that the word 'red' is rated 4, the word 'blue' is rated 7, the word 'orange' is rated 2, etc.

The variable result should calculate the highest rated word in list a. Assume there is just one such word.

Using the lists a and b below, result would calculate the string: 'yellow'

This one can be written in one line. We will allow more than one line for this one for full credit.

```
a = ['red', 'blue', 'orange', 'yellow', 'green']
b = [4, 7, 2, 8, 1]
result = ______________________________________________
```

PART F (4 pts)

Here you are given a dictionary d.

The variable result should calculate the sorted list of the keys from the dictionary d.

Using the dictionary d below, result would calculate the list: ['a', 'c', 'h', 'j', 'm', 'p']

```
d = {"a":6, "h":  3, "j":7, "c":3, "m":2, "p":6 }
result = ______________________________________________
```

PART G (4 pts)

Here you are given a dictionary d.

The variable result should calculate the unique sorted values from the dictionary d.

Using the dictionary d below, result would calculate the list: [2, 3, 6, 7]

```
d = {"a":6, "h":  3, "j":7, "c":3, "m":2, "p":6 }
result = ______________________________________________
```

PART H (4 pts)

Here you are given a dictionary d.

The variable result should calculate the number of integers over all of the value lists from the dictionary d.

Using the dictionary d below, result would calculate the number: 11

Note that the key "red" has 3 values, the key "blue" has 4 values, etc.

```
d = {"red":[5, 8, 2], "blue":[8, 2, 6, 1], "green":[3, 1], "gray":[9,3] }
result = ______________________________________________
```

PROBLEM 3 : (***What's wrong with my code? (6 points)***)

Consider the following list of strings that has several names, all lowercase, with each name followed by one or more integer scores, each as a string.

```
[ 'fred', '98', '56', '83', '74',
  'sarah', '92', '97', '91',
   'meg', '76', '92', '94', '82', '89',
   'bala', '91', '96',
   'junyang', '73', '76', '84', '82']
```

Consider the following functions below. The function isAlpha returns True if letter is a lowercase letter, and false otherwise. This function is correct. The function processScores has one parameter data, which is a list in the format described above. This function is supposed to return a list of tuples where the first item in the tuple is the name of a person, and the second item in the tuple is a float that is the average of that person's scores. The tuples are supposed to be sorted in reverse order by the average scores.

The correct output for the list above is supposed to be (note bala has the highest average score and fred has the lowest average score):

```
[('bala', 93.5), ('sarah', 93.33333333333333), ('meg', 86.6),
 ('junyang', 78.75), ('fred', 77.75)]
```

but instead it is returning the list:

```
[('fred', 77.75), ('sarah', 93.33333333333333), ('meg', 86.6), ('bala', 93.5)]
```

Here are the two functions:

```
    def isAlpha(letter):
        return letter in "abcdefghijklmnopqrstuvwxyz"

1   def processScores(data):
2       answer = []
3       index = 0
4       name = ""
5       nums = []
6       first = True
7       while index<len(data):
8           if isAlpha(data[index][0]):
9               if not first:   # found another one
10                  answer.append((name, sum(nums)/len(nums)))
11              else: # first one
12                  first = False
13              name = data[index]
14              nums = []
15          else:    # number
16              nums.append(float(data[index]))
17          index += 1
18      return answer
```

Explain briefly what the error(s) are and give the code below to fix this function. Do not rewrite the function, but give only a few lines of code to fix it and say where that code goes above by referring to the line numbers.

PROBLEM 4 : (***Visit the NC State Fair sometime (48 pts)***)

Consider the following list of lists of information from the NC State fair, assigned to the variable named data.

Each inner list has the following information in it. The first string is a one word name of a game at the fair. The second string is a one word name of a person. The third string is an amount for the cost to play the game. The fourth string is either the word "won" or the word "lost". If the fourth string is "won", then there is a fifth string that is the type of stuffed animal the person won playing the game.

For example in the first inner list below, the person jackson played the ringtoss game that costs $3 to play and lost the game. In the second list, the person xiaobai played the darts game that costs $2 to play and won a stuffed dog.

```
data = [ ['ringtoss', 'jackson', '3.00', 'lost'],
  ['darts', 'xiaobai', '2.00', 'won', 'dog'],
  ['basketball', 'jackson', '3.00', 'lost'],
  ['ringtoss', 'kim', '3.00', 'lost'],
  ['bingo', 'brett', '2.00', 'won', 'camel'],
  ['darts', 'hattie', '2.00', 'lost'],
  ['ladderclimb', 'jackson', '2.00', 'lost'],
  ['ringtoss', 'xiaobai', '3.00', 'won', 'cat'],
  ['bingo', 'hattie', '2.00', 'lost'],
  ['platebreak', 'sarah', '3.00', 'lost'],
  ['darts', 'hattie', '2.00', 'won', 'dog'],
  ['basketball', 'sarah', '3.00', 'won', 'cat'],
  ['ringtoss', 'xiaobai', '3.00', 'won', 'turtle'],
  ['ringtoss', 'jackson', '3.00', 'won', 'pikachu'],
  ['platebreak', 'jackson', '3.00', 'lost'],
  ['darts', 'sarah', '2.00', 'won', 'camel'],
  ['bingo', 'brett', '2.00', 'won', 'cat'] ]
```

This list is an example list. Your code for the problems that follow should work with any such list in this format.

Part A (8 pts)

Write the function named **howMuchSpent** that has two parameters, one named **data**, which is the list of lists described earlier, and one named **name**. This function returns a decimal number representing how much money the person named name has spent on fair games.

We state again that each inner list in the list of lists has the following information in it. The first string is a one word name of a game at the fair. The second string is a one word name of a person. The third string is an amount for the cost to play the game. The fourth string is either the word "won" or the word "lost". If the fourth string is "won", then there is a fifth string that is the type of stuffed animal the person won playing the game. Here is that example list of lists shown again.

```
data = [ ['ringtoss', 'jackson', '3.00', 'lost'],
  ['darts', 'xiaobai', '2.00', 'won', 'dog'],
  ['basketball', 'jackson', '3.00', 'lost'],
  ['ringtoss', 'kim', '3.00', 'lost'],
  ['bingo', 'brett', '2.00', 'won', 'camel'],
  ['darts', 'hattie', '2.00', 'lost'],
  ['ladderclimb', 'jackson', '2.00', 'lost'],
  ['ringtoss', 'xiaobai', '3.00', 'won', 'cat'],
  ['bingo', 'hattie', '2.00', 'lost'],
  ['platebreak', 'sarah', '3.00', 'lost'],
  ['darts', 'hattie', '2.00', 'won', 'dog'],
  ['basketball', 'sarah', '3.00', 'won', 'cat'],
  ['ringtoss', 'xiaobai', '3.00', 'won', 'turtle'],
  ['ringtoss', 'jackson', '3.00', 'won', 'pikachu'],
  ['platebreak', 'jackson', '3.00', 'lost'],
  ['darts', 'sarah', '2.00', 'won', 'camel'],
  ['bingo', 'brett', '2.00', 'won', 'cat'] ]
```

Here are several examples of calls to this function using the example list shown above.

call	**returns**
howMuchSpent(data, "jackson")	14.0
howMuchSpent(data, "sarah")	8.0

Complete the function below.

```
def howMuchSpent(data, name):
```

Part B (8 pts)

Write the function named **uniqueGamesWon** that has two parameters, one named **data**, which is the list of lists described earlier, and one named **name**. This function returns a sorted list of the unique games the person named name has won at the fair.

We state again that each inner list in the list of lists has the following information in it. The first string is a one word name of a game at the fair. The second string is a one word name of a person. The third string is an amount for the cost to play the game. The fourth string is either the word "won" or the word "lost". If the fourth string is "won", then there is a fifth string that is the type of stuffed animal the person won playing the game. Here is that example list of lists shown again.

```
data = [ ['ringtoss', 'jackson', '3.00', 'lost'],
  ['darts', 'xiaobai', '2.00', 'won', 'dog'],
  ['basketball', 'jackson', '3.00', 'lost'],
  ['ringtoss', 'kim', '3.00', 'lost'],
  ['bingo', 'brett', '2.00', 'won', 'camel'],
  ['darts', 'hattie', '2.00', 'lost'],
  ['ladderclimb', 'jackson', '2.00', 'lost'],
  ['ringtoss', 'xiaobai', '3.00', 'won', 'cat'],
  ['bingo', 'hattie', '2.00', 'lost'],
  ['platebreak', 'sarah', '3.00', 'lost'],
  ['darts', 'hattie', '2.00', 'won', 'dog'],
  ['basketball', 'sarah', '3.00', 'won', 'cat'],
  ['ringtoss', 'xiaobai', '3.00', 'won', 'turtle'],
  ['ringtoss', 'jackson', '3.00', 'won', 'pikachu'],
  ['platebreak', 'jackson', '3.00', 'lost'],
  ['darts', 'sarah', '2.00', 'won', 'camel'],
  ['bingo', 'brett', '2.00', 'won', 'cat'] ]
```

Here are several examples of calls to this function using the example list shown above.

call	**returns**
uniqueGamesWon(data, "brett")	['bingo']
uniqueGamesWon(data, "sarah")	['basketball', 'darts']
uniqueGamesWon(data, "xiaobai")	['darts', 'ringtoss']

Complete the function below

```
def uniqueGamesWon(data, name):
```

Part C (8 pts)

Write the function named **checkdata** that has one parameter named **data**, which is the list of lists described earlier. This function checks the list data to see if each individual game charges the same amount to any person who plays that game. If it finds a game that charges more than one amount it prints the words "Error in price" followed by the name of that game. This will provide an alert that the datafile needs to be checked.

We state again that each inner list in the list of lists has the following information in it. The first string is a one word name of a game at the fair. The second string is a one word name of a person. The third string is an amount for the cost to play the game. The fourth string is either the word "won" or the word "lost". If the fourth string is "won", then there is a fifth string that is the type of stuffed animal the person won playing the game. Here is that example list of lists shown again.

```
data = [ ['ringtoss', 'jackson', '3.00', 'lost'],
  ['darts', 'xiaobai', '2.00', 'won', 'dog'],
  ['basketball', 'jackson', '3.00', 'lost'],
  ['ringtoss', 'kim', '3.00', 'lost'],
  ['bingo', 'brett', '2.00', 'won', 'camel'],
  ['darts', 'hattie', '2.00', 'lost'],
  ['ladderclimb', 'jackson', '2.00', 'lost'],
  ['ringtoss', 'xiaobai', '3.00', 'won', 'cat'],
  ['bingo', 'hattie', '2.00', 'lost'],
  ['platebreak', 'sarah', '3.00', 'lost'],
  ['darts', 'hattie', '2.00', 'won', 'dog'],
  ['basketball', 'sarah', '3.00', 'won', 'cat'],
  ['ringtoss', 'xiaobai', '3.00', 'won', 'turtle'],
  ['ringtoss', 'jackson', '3.00', 'won', 'pikachu'],
  ['platebreak', 'jackson', '3.00', 'lost'],
  ['darts', 'sarah', '2.00', 'won', 'camel'],
  ['bingo', 'brett', '2.00', 'won', 'cat'] ]
```

For this part only, assume the following two lists are added at the end of the list data above.

```
['darts', 'hattie', '1.00', 'lost']
['basketball', 'jackson', '2.00', 'won', 'cat']
```

Then the call checkdata(data) would print the following since the price to play darts is sometimes 2.00 and sometimes 1.00, and the price to play basketball is sometimes 3.00 and sometimes 2.00.

```
Error in price darts
Error in price basketball
```

Complete the function below.

```
def checkdata(data):
```

Part D (8 pts)

Write the function named **dictNamesToListAnimals** that has one parameter named **data**, which is the list of lists described earlier. This function returns a dictionary that maps names to the list of animals they have won at the fair.

We state again that each inner list in the list of lists has the following information in it. The first string is a one word name of a game at the fair. The second string is a one word name of a person. The third string is an amount for the cost to play the game. The fourth string is either the word "won" or the word "lost". If the fourth string is "won", then there is a fifth string that is the type of stuffed animal the person won playing the game. Here is that example list of lists shown again.

```
data = [ ['ringtoss', 'jackson', '3.00', 'lost'],
  ['darts', 'xiaobai', '2.00', 'won', 'dog'],
  ['basketball', 'jackson', '3.00', 'lost'],
  ['ringtoss', 'kim', '3.00', 'lost'],
  ['bingo', 'brett', '2.00', 'won', 'camel'],
  ['darts', 'hattie', '2.00', 'lost'],
  ['ladderclimb', 'jackson', '2.00', 'lost'],
  ['ringtoss', 'xiaobai', '3.00', 'won', 'cat'],
  ['bingo', 'hattie', '2.00', 'lost'],
  ['platebreak', 'sarah', '3.00', 'lost'],
  ['darts', 'hattie', '2.00', 'won', 'dog'],
  ['basketball', 'sarah', '3.00', 'won', 'cat'],
  ['ringtoss', 'xiaobai', '3.00', 'won', 'turtle'],
  ['ringtoss', 'jackson', '3.00', 'won', 'pikachu'],
  ['platebreak', 'jackson', '3.00', 'lost'],
  ['darts', 'sarah', '2.00', 'won', 'camel'],
  ['bingo', 'brett', '2.00', 'won', 'cat'] ]
```

When the call dictNamesToListAnimals(data) is made with the list data above, then a dictionary is returned. Here is that dictionary.

```
{'xiaobai': ['dog', 'cat', 'turtle'],
  'brett': ['camel', 'cat'],
  'hattie': ['dog'],
  'sarah': ['cat', 'camel'],
  'jackson': ['pikachu']
   }
```

Complete the function below

```
def dictNamesToListAnimals(data):
```

Part E (8 pts)

Write the function named **animalWon** that has two parameters, one parameter named **dname**, which is a dictionary of names mapped to a list of animals the person named name has won, and a second parameter named **numtimes** that is an integer.

This function returns a sorted list of the stuffed animals that have been won exactly numtimes.

Here is an example. Suppose the dictionary named dname is the resulting dictionary from the previous problem shown here.

```
dname = {'xiaobai': ['dog', 'cat', 'turtle'],
  'brett': ['camel', 'cat'],
  'hattie': ['dog'],
  'sarah': ['cat', 'camel'],
  'jackson': ['pikachu', 'cat']
   }
```

Here are several examples of calls to this function.

call	**returns**
animalWon(dname, 1)	['pikachu', 'turtle']
animalWon(dname, 2)	['camel', 'dog']

Complete the function below

```
def animalWon(dname, numtimes):
```

Part F (8 pts)

Write the function named **wonMostPrizes** that has one parameter named **data**, which is the list of lists described earlier. This function computes the person who has won the most prizes, and returns a tuple of the person's name and a sorted list of the prizes they have won.

We state again that each inner list in the list of lists has the following information in it. The first string is a one word name of a game at the fair. The second string is a one word name of a person. The third string is an amount for the cost to play the game. The fourth string is either the word "won" or the word "lost". If the fourth string is "won", then there is a fifth string that is the type of stuffed animal the person won playing the game. Here is that example list of lists shown again.

```
data = [ ['ringtoss', 'jackson', '3.00', 'lost'],
  ['darts', 'xiaobai', '2.00', 'won', 'dog'],
  ['basketball', 'jackson', '3.00', 'lost'],
  ['ringtoss', 'kim', '3.00', 'lost'],
  ['bingo', 'brett', '2.00', 'won', 'camel'],
  ['darts', 'hattie', '2.00', 'lost'],
  ['ladderclimb', 'jackson', '2.00', 'lost'],
  ['ringtoss', 'xiaobai', '3.00', 'won', 'cat'],
  ['bingo', 'hattie', '2.00', 'lost'],
  ['platebreak', 'sarah', '3.00', 'lost'],
  ['darts', 'hattie', '2.00', 'won', 'dog'],
  ['basketball', 'sarah', '3.00', 'won', 'cat'],
  ['ringtoss', 'xiaobai', '3.00', 'won', 'turtle'],
  ['ringtoss', 'jackson', '3.00', 'won', 'pikachu'],
  ['platebreak', 'jackson', '3.00', 'lost'],
  ['darts', 'sarah', '2.00', 'won', 'camel'],
  ['bingo', 'brett', '2.00', 'won', 'cat'] ]
```

As an example, if wonMostPrizes(data) is called on the list shown above, then this function would return the tuple: ('xiaobai', ['cat', 'dog', 'turtle']).

Complete the function below

```
def wonMostPrizes(data):
```