PROBLEM 2 : (***List Comprehensions (16 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 given, your code would still calculate the correct answer.

Each of these must be written in one line and include a list comprehension.

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 numbers from the list nums that are greater than 8.

Using the list nums below, result would calculate the list: [9, 21, 38, 15]

```
nums = [5, 2, 9, 21, 38, 6, 15, 4]
result = _____________________________________________
```

PART B (4 pts)

The variable **result** should calculate the list of numbers from the list nums that are between 5 and 20. Note that does not include the numbers 5 or 20.

Using the list nums below, result would calculate the list: [9, 6, 15]

```
nums = [5, 2, 9, 21, 38, 6, 15, 4]
result = _____________________________________________
```

PART C (4 pts)

The variable **result** should create a list of the first letter from each word in the list colors.

Assume each string in the list colors is one word that is lowercase.

Using the list colors below, result would calculate the list: ['b', 'r', 'y', 'g', 'w', 'b', 'p']

```
colors = ['blue', 'red', 'yellow', 'green', 'white', 'brown', 'purple']
result = _____________________________________________
```

PART D (4 pts)

The variable result should calculate the list of words from the list colors that end in a vowel. Vowels are defined as the letters "a", "e", "i", "o", and "u". Assume each word in the list colors is lowercase. You may also use the String variable vowels defined below.

Using the list colors below, result would calculate the list: ['blue', 'aqua', 'white', 'purple']

```
vowels = "aeiou"
colors = ['blue', 'red', 'yellow', 'aqua', 'green', 'white', 'purple']
result = _____________________________________________
```

PROBLEM 3 : (*Simple Code (8 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 value of the list given or other variables given, your code would still calculate the correct answer.

In solving these problems, you can use additional lines with additional variables if you want.

PART A (4 pts)

The variable **result** should calculate the sum of those numbers in the list nums that are divisible by 3 and are even numbers.

Using the list nums below, result would calculate 24. That is because 18 and 6 are the only numbers in the list nums that are even numbers that are divisible by 3.

```
nums = [5, 18, 2, 9, 21, 1, 29, 22, 6, 15, 4]
result = _____________________________________________
```

PART B (4 pts)

The variable **result** should use the variables phrase and letter and create a new word that is composed of words in phrase that start with letter, without that first letter, and those words are combined with a dash between each pair of such words.

Assume each word in phrase is lowercase, phrase has at least one word in it and two consecutive words are separated by a blank. Assume letter is one letter and is lowercase.

Using the phrase and letter below, result would calculate the string: "ug-ird-oat-lue". Note there are four words in phrase that start with the letter 'b': "bug", "bird", "boat" and "blue". Those words without their first letter are combined together with the dash between consecutive words.

```
phrase = "bug car bird top red boat ship tree eye blue"
letter = 'b'
result = _____________________________________________
```

PROBLEM 4 : (*Debugging and Tracing (8 pts)*)

Consider the following function named process that takes as input a list of numbers named data and is supposed to return the sum of only increasing numbers starting with the first number. As soon as a consecutive number is not increasing, then no more numbers are added to the sum. Assume data always has at least one number in its list.

This function does not work correctly.

```
1   def process(data):
2      sum = data[0]
3      index = 1
4      while (data[index] > data[index-1]):
5         sum += data[index]
6         index = index + 1
7      return sum
```

Shown here are two calls to process that work as expected. In each call, they start with the first number and continue to add the next number in the list if it is larger than the previous number.

call	**returns**	**comment**
process([5, 8, 13, 24, 9, 15, 4, 27, 3, 11, 7])	50	5 + 8 + 13 + 24, 9 is smaller than 24
process([3, 4, 8, 2, 5, 11, 14])	15	3 + 4 + 8, 2 is smaller than 8

a) Give another example of a call to process that has a list with at least four numbers, that works fine and returns the correct answer.

b) Give an example of a call to process that has a list with at least four numbers, that results in an error.

c)Give the line number causing the error and explain what the error is.

d) Explain how to fix the code by changing one line so it works correctly. Give the line of code and explaining where it goes in the code above.

PROBLEM 5 : (***Dribbling along(40 pts)***)

There are five functions to write in this part. Your functions should work for any valid data, not just the examples shown.

This problem is about data related to high school basketball players. Assume that there is no player with the same name at the same school.

Part A (8 pts)

Write the function named **schoolOfPlayer** that has two parameters. The first parameter is named **data**, which is a list of lists with each inner list having the following five fields: a String representing the name of a high school, a String representing the name of a basketball player, the player's average score (a float), the player's average number of rebounds (a float) and the player's average number of assists (a float). The second parameter is a String named **player** and is the name of a basketball player.

This function returns the name of the high school for the specified player. If the player's name is not in data then return "None". If the player's name occurs more than once in data, then return the first such school.

For example, assume data is the list of lists shown below. Note that in the first inner list, the high school is 'Sanderson', the player is 'Smith', the player's average score is 10.2, the player's average number of rebounds is 2.1 and the player's average number of assists is 4.5.

```
data = [ ['Sanderson', 'Smith', 10.2, 2.1, 4.5],
  ['Cary Academy', 'WJohnson', 11.4, 5.7, 10.8],
  ['Sanderson', 'Kiline', 15.7, 1.2, 3.2],
  ['Broughton', 'Williams', 17.2, 8.1, 1.3],
  ['Leesville Road', 'Smith', 18.6, 12.3, 4.1],
  ['Cary Academy', 'Blossom', 4.3, 1.4, 7.3],
  ['Enloe', 'Jackson', 4.5, 1.7, 7.3],
  ['Rolesville', 'Moore', 8.9, 3.1, 4.8],
  ['Leesville Road', 'Miller', 8.7, 10.3, 10.1],
  ['Broughton', 'Smith', 16.9, 3.8, 4.3],
  ['Rolesville', 'Oneal', 9.6, 4.2, 2.7],
  ['Cary Academy', 'SJohnson', 17.4, 11.1, 3.2],
  ['Enloe', 'Gildner', 3.8, 2.6, 1.7],
  ['Broughton', 'Grayson', 12.4, 5.9, 5.8] ]
```

Here are several examples of calls to this function. In the third example, notice there are several players named "Smith", so the first such one is returned.

call	**returns**
schoolOfPlayer(data, "Moore")	"Rolesville"
schoolOfPlayer(data, "Spellings")	"None"
schoolOfPlayer(data, "Smith")	"Sanderson"

Complete the function below.

```
def schoolOfPlayer(data, player):
```

Part B (8 pts)

Write the function named **topPoints** that has one parameter named **data** that is a list of lists, where each inner list is in the format described in Part A, and described again here. Each inner list in **data** has the following five fields: a String representing the name of a high school, a String representing the name of a basketball player, the player's average score (a float), the player's average number of rebounds (a float) and the player's average number of assists (a float).

This function calculates the player with the highest average score and returns a String in the format "PLAYER at SCHOOL" that includes the player's name first, followed by a blank, followed by "at", followed by a blank, and finally the name of the school the player is at.

For example, assume data is the list of lists shown in Part A and shown again below. Note that in the first list, the high school is 'Sanderson', the player is 'Smith', the player's average score is 10.2, the player's average number of rebounds is 2.1 and the player's average number of assists is 4.5.

```
data = [ ['Sanderson', 'Smith', 10.2, 2.1, 4.5],
  ['Cary Academy', 'WJohnson', 11.4, 5.7, 10.8],
  ['Sanderson', 'Kiline', 15.7, 1.2, 3.2],
  ['Broughton', 'Williams', 17.2, 8.1, 1.3],
  ['Leesville Road', 'Smith', 18.6, 12.3, 4.1],
  ['Cary Academy', 'Blossom', 4.3, 1.4, 7.3],
  ['Enloe', 'Jackson', 4.5, 1.7, 7.3],
  ['Rolesville', 'Moore', 8.9, 3.1, 4.8],
  ['Leesville Road', 'Miller', 8.7, 10.3, 10.1],
  ['Broughton', 'Smith', 16.9, 3.8, 4.3],
  ['Rolesville', 'Oneal', 9.6, 4.2, 2.7],
  ['Cary Academy', 'SJohnson', 17.4, 11.1, 3.2],
  ['Enloe', 'Gildner', 3.8, 2.6, 1.7],
  ['Broughton', 'Grayson', 12.4, 5.9, 5.8] ]
```

Here is an example of a call to this function. Note that Smith at the school Leesville Road has an average of 18.6 points per game, which is the highest overall average for points per game.

call	**returns**
topPoints(data)	"Smith at Leesville Road"

Complete the function below.

```
def topPoints(data):
```

Part C (8 pts)

Write the function named **playersFromSchool** that has two parameters. The first parameter is named **data**, which is a list of lists, where each inner list is in the format described in Part A, and described again here. Each inner list in **data** has the following five fields: a String representing the name of a high school, a String representing the name of a basketball player, the player's average score (a float), the player's average number of rebounds (a float) and the player's average number of assists (a float). The second parameter is named **school** and it is a String that is the name of a high school.

This function returns a list of player's names for the specified high school.

For example, assume data is the list of lists shown in Part A and shown again below. Note that in the first list, the high school is 'Sanderson', the player is 'Smith', the player's average score is 10.2, the player's average number of rebounds is 2.1 and the player's average number of assists is 4.5.

```
data = [ ['Sanderson', 'Smith', 10.2, 2.1, 4.5],
  ['Cary Academy', 'WJohnson', 11.4, 5.7, 10.8],
  ['Sanderson', 'Kiline', 15.7, 1.2, 3.2],
  ['Broughton', 'Williams', 17.2, 8.1, 1.3],
  ['Leesville Road', 'Smith', 18.6, 12.3, 4.1],
  ['Cary Academy', 'Blossom', 4.3, 1.4, 7.3],
  ['Enloe', 'Jackson', 4.5, 1.7, 7.3],
  ['Rolesville', 'Moore', 8.9, 3.1, 4.8],
  ['Leesville Road', 'Miller', 8.7, 10.3, 10.1],
  ['Broughton', 'Smith', 16.9, 3.8, 4.3],
  ['Rolesville', 'Oneal', 9.6, 4.2, 2.7],
  ['Cary Academy', 'SJohnson', 17.4, 11.1, 3.2],
  ['Enloe', 'Gildner', 3.8, 2.6, 1.7],
  ['Broughton', 'Grayson', 12.4, 5.9, 5.8] ]
```

Here are several examples of calls to this function.

call	**returns**
playersFromSchool(data, "Broughton")	['Williams', 'Smith', 'Grayson']
playersFromSchool(data, "Rolesville")	['Moore', 'Oneal']
playersFromSchool(data, "Cary Academy")	['WJohnson', 'Blossom', 'SJohnson']

Complete the function below.

```
def playersFromSchool(data, school):
```

Part D (8 pts)

Write the function named **topPlayers** that has one parameter named **data**, which is a list of lists, where each inner list is in the format described in Part A, and described again here. Each inner list in **data** has the following five fields: a String representing the name of a high school, a String representing the name of a basketball player, the player's average score (a float), the player's average number of rebounds (a float) and the player's average number of assists (a float).

This function returns a list of tuples of a school and a player's name, for all the players that have at least two of their three averages at 10.0 or higher.

For example, assume data is the list of lists shown in Part A and shown again below. Note that in the first list, the high school is 'Sanderson', the player is 'Smith', the player's average score is 10.2, the player's average number of rebounds is 2.1 and the player's average number of assists is 4.5.

```
data = [ ['Sanderson', 'Smith', 10.2, 2.1, 4.5],
  ['Cary Academy', 'WJohnson', 11.4, 5.7, 10.8],
  ['Sanderson', 'Kiline', 15.7, 1.2, 3.2],
  ['Broughton', 'Williams', 17.2, 8.1, 1.3],
  ['Leesville Road', 'Smith', 18.6, 12.3, 4.1],
  ['Cary Academy', 'Blossom', 4.3, 1.4, 7.3],
  ['Enloe', 'Jackson', 4.5, 1.7, 7.3],
  ['Rolesville', 'Moore', 8.9, 3.1, 4.8],
  ['Leesville Road', 'Miller', 8.7, 10.3, 10.1],
  ['Broughton', 'Smith', 16.9, 3.8, 4.3],
  ['Rolesville', 'Oneal', 9.6, 4.2, 2.7],
  ['Cary Academy', 'SJohnson', 17.4, 11.1, 3.2],
  ['Enloe', 'Gildner', 3.8, 2.6, 1.7],
  ['Broughton', 'Grayson', 12.4, 5.9, 5.8] ]
```

For the call, topPlayers(data) it would return the list:

[('Cary Academy', 'WJohnson'), ('Leesville Road', 'Smith'), ('Leesville Road', 'Miller'), ('Cary Academy', 'SJohnson')]

Note that WJohnson at Cary Academy has an 11.4 average points per game, and 10.8 average assists per game. Each of the other three players have at least two of their averages at 10.0 or higher.

Complete the function below.

```
def topPlayers(data):
```

Part E (8 pts)

How did we create the list data from Part A? We need to read the high school basketball data from a file.

Complete the function named **fileToList** that has one parameter named **filename**. This function reads the file named filename that is in the format listed below, and returns the list of lists format used in the previous parts A-D and described again here. Each inner list has the following five fields: a String representing the name of a high school, a String representing the name of a basketball player, the player's average score (a float), the player's average number of rebounds (a float) and the player's average number of assists (a float).

Information about high school basketball players is stored in a file in the following format. Each line represents one player from one high school. For each line there are five pieces of information and three separators (in this order): the name of the school (one or more words), a blank, the name of a basketball player (one word), a colon, the player's average number of points scored per game (float), a dash ("-"), the player's average number of rebounds per game, a dash ("-"), and the player's average number of assists per game.

Shown below is a sample file. Your program should work for any file in the specified format. In the first line below, the high school is "Sanderson", the player is "Smith", Smith's average number of points per game is 10.2, Smith's average number of rebounds per game is 2.1 and Smith's average number of assists is 4.5 points per game. In the second line below, the high school is "Cary Academy", the player is "WJohnson", WJohnson's average number of points per game is 11.4, WJohnson's average number of rebounds per game is 5.7, and WJohnson's average number of assists per game is 10.8.

```
Sanderson Smith:10.2-2.1-4.5
Cary Academy WJohnson:11.4-5.7-10.8
Sanderson Kiline:15.7-1.2-3.2
Broughton Williams:17.2-8.1-1.3
Leesville Road Smith:18.6-12.3-4.1
Cary Academy Blossom:4.3-1.4-7.3
Enloe Jackson:4.5-1.7-7.3
Rolesville Moore:8.9-3.1-4.8
Leesville Road Miller:8.7-10.3-10.1
Broughton Smith:16.9-3.8-4.3
Rolesville Oneal:9.6-4.2-2.7
Cary Academy SJohnson:17.4-11.1-3.2
Enloe Gildner:3.8-2.6-1.7
Broughton Grayson:12.4-5.9-5.8
```

In calling the function fileToList on this file, the list shown below would be returned.

```
[ ['Sanderson', 'Smith', 10.2, 2.1, 4.5],
  ['Cary Academy', 'WJohnson', 11.4, 5.7, 10.8],
  ['Sanderson', 'Kiline', 15.7, 1.2, 3.2],
  ['Broughton', 'Williams', 17.2, 8.1, 1.3],
  ['Leesville Road', 'Smith', 18.6, 12.3, 4.1],
  ['Cary Academy', 'Blossom', 4.3, 1.4, 7.3],
  ['Enloe', 'Jackson', 4.5, 1.7, 7.3],
  ['Rolesville', 'Moore', 8.9, 3.1, 4.8],
  ['Leesville Road', 'Miller', 8.7, 10.3, 10.1],
  ['Broughton', 'Smith', 16.9, 3.8, 4.3],
  ['Rolesville', 'Oneal', 9.6, 4.2, 2.7],
  ['Cary Academy', 'SJohnson', 17.4, 11.1, 3.2],
  ['Enloe', 'Gildner', 3.8, 2.6, 1.7],
  ['Broughton', 'Grayson', 12.4, 5.9, 5.8] ]
```

Complete the function below. You just need to show the missing lines. You do not need to indent the first line. We will assume it is the first line of the body of the for loop.

```
def fileToList(filename):
    f = open(filename)
    answer = []
    for line in f:

        MISSING LINES

    return answer
```