PROBLEM 1 : (***What is the output? (16 points)***)

Part A. What is the output of the following code segments?
Write the output to the right. Note that there is only output for the print statements.

OUTPUT

```
lista = ['B', 'P']
listb = ['S', 'G']
lista.extend(listb)
print lista
lista = ['B', 'P', 'W', 'C']
lista.insert(1,'B')
print lista
lista.pop()
print lista
lista.remove('B')
print lista
# -------------------------
seta = set([5,6,2,2,6,2,5])
seta.add(4)
seta.add(6)
print sorted(list(seta))
seta = set([5,6,2,2,6,2,5])
seta.remove(2)
print sorted(list(seta))
# -------------------------
seta = set([7, 4, 2])
setb = set([2, 1, 7])
print setb.symmetric_difference(seta)
print setb.union(seta)
print seta & setb
# -------------------------
lista = ['N', 'L', 'H', 'J']
d = {'J':6, 'H':3, 'Q':3, 'D':2}
print sorted(d.keys())
print [d[i] for i in lista if i in d]
d['H'] = 5
d['Y'] = 6
print sorted(d.keys())
print sorted(d.values())
```

Part B. What is the output of the following code segment? **Write the output after the code segment**. Note that there is only output for the print statements.

```
colors = ['red', 'blue', 'green', 'white', 'purple', 'black', 'pink']
dict = {}
for item in colors:
    if len(item) not in dict:
        dict[len(item)] = 1
    else:
        dict[len(item)] += 1
print dict.keys()
print dict.values()
print max(dict.values())
```

What is the output?

PROBLEM 2 : (*Short Code/Answer (17 points)*)

For parts that involve writing code, your code should also work if the given list was modified with different values.

A. (3 pts) Consider the following code.

What is the value of temp after this line executes?

```
values = [18, 25, 17, 40, 6, 12]
temp = [n for n in values if n%5 < 2]
```

B. (3 pts) Consider the following code.

What is the value of temp after this line executes?

```
words = ['yellow','purple','green','blue']
temp = sorted([words[i]+words[i-1] for i in range(len(words)) if i>1])
```

C. (5 pts) **Write one line of code that includes a list comprehension** to assign to the variable `temp` a list of the strings from the list `words` in which each word from the original list that is of length four or more has the first two letters swapped. The resulting list should have the modified words in the same order as the original list.

```
words = ['yellow','purple','green','blue','red', 'pink']
```

For example, if `words` was the list above, then after executing the list comprehension, then `temp` would be the list `['eyllow', 'uprple', 'rgeen', 'lbue', 'ipnk']`

Write the list comprehension below.

```
temp =
```

D. (6 pts) Write a function named `nameinfo` that has one parameter `names` that is a list of strings representing names. Each name is one or more words separated by a blank. This function returns a list of tuples. For each name that has two or more words, a tuple is created. The first part of the tuple is a string of the first and last words of the name separated by a blank, and the second part of the tuple is the number of words between the first and last word in the name.

For example, suppose `names` has the following value:

```
names = ['long john silver', 'mary sue ellen jo cole', 'prince', 'bruce wayne']
```

Then the call `nameinfo(names)` would return the list

`[('long silver', 1), ('mary cole', 3), ('bruce wayne', 0)]`

Note that only those names with two or more words result in a tuple in the new list, and the tuples appear in the same order the names were in the original list.

```
def nameinfo (names):
```

PROBLEM 3 : (*Wins and Losses (15 points)*)

Consider the following data file of information on club basketball teams. Each line in the file represents two teams playing each other and their scores. The format of each line in the file is team1, followed by a hyphen, followed by the number of points team1 made, followed by a colon, followed by team2, followed by a hyphen, and followed by the number of points team2 made. The first team on each line is the home team, where the game was played.

An example of the data file is shown below. For example, in the first line, duke was the home team and duke played against unc, with duke scoring 78 points and unc scoring 76 points, so duke won the game.

```
duke-78:unc-76
unc-87:virginia tech-80
wake forest-73:duke-92
miami-82:unc-79
wake forest-67:miami-77
ncsu-68:unc-70
unc-80:gatech-65
ncsu-77:virginia tech-73
```

```
virginia tech-83:wake forest-79
gatech-75:ncsu-81
gatech-81:wake forest-70
duke-76:ncsu-74
virginia tech-75:miami-74
```

A. (7 pts) Write the function `processinfo` that has one parameter `filename` which represents the name of the file. This function returns a list of lists of items in which each inner list has four items and represents one line from the file. The first item is a string of team1's name, the second item is the integer number of points team1 scored, the third item is a string of team2's name, and the fourth item is the integer number of points team2 scored.

For example, the line `data = processinfo("teamdata.txt")` where "teamdata.txt" is the file above would result in data having the value on the next page.

```
data = [ ['duke', 78, 'unc', 76],
['unc', 87, 'virginia tech', 80],
['wake forest', 73, 'duke', 92],
['miami', 82, 'unc', 79],
['wake forest', 67, 'miami', 77],
['ncsu', 68, 'unc', 70],
['unc', 80, 'gatech', 65],
['ncsu', 77, 'virginia tech', 73],
['virginia tech', 83, 'wake forest', 79],
['gatech', 75, 'ncsu', 81],
['gatech', 81, 'wake forest', 70],
['duke', 76, 'ncsu', 74],
['virginia tech', 75, 'miami', 74] ]
```

Complete the function processinfo below.

```
def processinfo(filename):
    f = open(filename)
```

B. (8 pts) Write the function `schoolsBeat` that has two parameters, `data` and `team`, where `data` is the list of lists in the format from Part A, and team is a string.

This function returns a list of tuples, where each tuple is information about a game that `team` won. Each tuple has the name of the team beat, followed by the number of points they won by.

For example, assume *data* is the lists of lists of four items on the previous page. The two examples below show the result of calling `schoolsBeat` with this filename and a team name. For example, duke beat three teams, ncsu by 2 points, unc by 2 points and wake forest by 19 points, wake forest did not beat any teams, and unc beat three teams.

call	returns
schoolsBeat(data, "duke")	[('ncsu', 2), ('unc', 2), ('wake forest', 19)]
schoolsBeat(data, "wake forest")	[]
schoolsBeat(data, "unc")	[('gatech', 15), ('ncsu', 2), ('virginia tech', 7)]

```
def schoolsBeat(data, team):
```

PROBLEM 4 : (*Movie Stars (44 points)*)

Suppose you have data about movies and actors in the format of a list of lists, where each list represents the actor in one movie. In particular, each list has **five strings**: the name of a movie, the name of an actor in that movie, the year the movie was released, the total time of the movie in minutes and the total time in minutes this actor appeared in this movie.

For example, suppose `datalist` is the list below. The first list in `datalist` represents the movie "Saving Mr. Banks". The second item is the actor "Tom Hanks", the third item is the year "2016" the movie was released, the fourth item is the total time in minutes for the movie, "125", and the fifth item is the total amount of time Tom Hanks was actually in the movie, "65" minutes.

```
datalist = [
['Saving Mr. Banks', 'Tom Hanks', '2016', '125', '65'],
['Saving Mr. Banks', 'Emma Thompson', '2016', '125', '84'],
['Enough Said', 'James Gandolfini', '2013', '93', '52'],
['Captain Phillips', 'Catherine Keener', '2013', '134', '22'],
['The Da Vinci Code', 'Tom Hanks', '2006', '149', '85'],
['Saving Mr. Banks', 'Colin Farrell', '2016', '125', '25'],
['Forrest Gump', 'Sally Field', '1994', '142', '56'],
['Mrs. Doubtfire', 'Robin Williams', '1993', '125', '94'],
['Captain Phillips', 'Tom Hanks', '2013', '134', '110'],
['Enough Said', 'Catherine Keener', '2013', '93', '21'],
['The Da Vinci Code', 'Ian McKellen', '2006', '149', '60'],
['Hello, My Name is Doris', 'Sally Field', '2015', '95', '84'],
['Alone in Berlin', 'Emma Thompson', '2016', '103', '70'],
['Forrest Gump', 'Tom Hanks', '1994', '142', '110'],
['Mrs. Doubtfire', 'Sally Field', '1993', '125', '45'] ]
```

In writing any of these functions, **you may call any other function you wrote for this problem.** Assume that function is correct, regardless of what you wrote.

A. (7 pts) Write the function named `actors` which has one parameter, `data`, that is a nonempty list of lists of five strings in the format mentioned earlier. This function returns a list of the **unique** names of actors from data, in any order.

For example, if `datalist` is the example list of lists mentioned earlier then the call `actors(datalist)` would return the list `['Robin Williams', 'Ian McKellen', 'Sally Field', 'Catherine Keener', 'Tom Hanks', 'Colin Farrell', 'Emma Thompson', 'James Gandolfini']`

```
def actors(data):
```

B. (7 pts) Write the function `moviesLength` which has two parameters, `data`, that is a nonempty list of lists of five strings in the format mentioned earlier, and an integer `time`. This function returns a **sorted unique** list of the movies that run in less than or equal to `time` in length.

For example, if `datalist` is the list of lists mentioned earlier, then `moviesLength(datalist, 100)` would return the list ['Enough Said', 'Hello, My Name is Doris'], the two movies that are 100 minutes or less in length. Note the movies are in sorted order.

```
def moviesLength(data, time):
```

C. (7 pts) Write the function `actorsNotIn` which has two parameters, `data`, that is a nonempty list of lists of five strings in the format mentioned earlier, and a list named `actorlist` that is a list of favorite actors. This function returns a list of favorite actors from actorlist that are not in any of the movies in data.

For example, if `datalist` is the list of lists mentioned earlier and if `favorite` is the list of actors ["Emma Watson", "Daniel Radcliffe", "Ralph Fiennes", "Tom Hanks"]

then the call `actorsNotIn(data, favorite)` would return the list `['Daniel Radcliffe', 'Ralph Fiennes', 'Emma Watson']`

```
def actorsNotIn(data,actorlist):
```

D. (8 pts) Write the function `moviesMostActors` which has one parameter, `data`, that is a nonempty list of lists of five strings in the format mentioned earlier. This function computes the list of the movies with the most actors, from the movies in datalist. **You must build a dictionary as part of solving this problem.**

For example, if `datalist` is the list of lists mentioned earlier, then the call `moviesMostActors (datalist)` would return the list ['Saving Mr. Banks'], which has three actors listed in datalist, the most number of any of the movies in datalist. If there was a tie, then you would return a list of all the movies that tied.

```
def moviesMostActors(data):
```

E. (7 pts) Write the function `dictActorsToMovies` which has one parameter, `data`, that is a nonempty list of lists of five strings in the format mentioned earlier. This function returns a dictionary of actors to a list of tuples, where each tuple is a movie the actor is in, and the total number of minutes the actor was in that movie.

For example, if datalist is the list of lists mentioned earlier, then the call `dictActorsToMovies(datalist)` would return a dictionary with several entries. One entry would have key 'Catherine Keener' with value [('Captain Phillips', '22'), ('Enough Said', '21')], as she is in those two movies.

```
def dictActorsToMovies(data):
```

F. (8 pts) Write the function `actorMostMinutes` which has one parameter, `data`, that is a nonempty list of lists of five strings in the format mentioned earlier. This function returns the name of the actor that has the most total time in movies from datalist. Assume there is no tie.

For example, if datalist is the list of lists mentioned earlier, then `actorMostMinutes` would return 'Tom Hanks', as he is in four movies with times 65, 85, 110 and 110 for a total of 370, more minutes than any other actor.

```
def actorMostMinutes(data):
```