PROBLEM 1: \(\textbf{What is the output? (18 pts)}\)

For the following code, write the output to the right of each print statement. The output for the first three print statements is already shown.

```python
seta = set([5, 4, 2])
print(seta) \{5, 4, 2\}
print(list(seta)) [5, 4, 2]
print(sorted(seta)) [2, 4, 5]

#----------------------------
seta = set([6, 7, 3, 1, 3, 6, 7, 6])
seta.add(6)
seta.add(8)
print(sorted(seta))

#----------------------------
seta = set([4, 4, 7, 3])
seta.remove(4)
print(sorted(seta))

#----------------------------
seta = set([7, 3, 1])
setb = set([6, 3, 2])
print(sorted(seta & setb))
print(sorted(setb ^ seta))
print(sorted(setb - seta))

#----------------------------
lst = [8, 5, 3, 4]
dict = {"E":4, "X":7, "C":4, "R":5}
print(sorted(dict.keys()))
print([dict[k] for k in dict if dict[k] in lst])
dict["H"] = 8
dict["X"] = 4
print(sorted(dict.keys()))
print(sorted(dict.values()))
```
PROBLEM 2: (Short code (21 pts))

For each of these problems, calculate the answer with code, such that the variable result’s value should be the resulting answer.
If we changed the list(s) or dictionaries given, your code should still calculate the correct answer.

You can write your answer in more than one line, and with more than one variable, but be sure result’s value should be the answer.

Here is an example.
Calculate the list of words from list vehicles that have the letter ’a’ in their word. Assume each string in the list is one word that is lowercase. The answer should be stored in the variable result.
Using the list vehicles below, result’s value would be
[‘train’, ’airplane’, ’car’, ’longboard’]

vehicles = [‘train’, ’airplane’, ’car’, ’truck’, ’bike’, ’longboard’]
result =
ANSWER:
result = [w for w in vehicles if ’a’ in w]

ALTERNATIVE ANSWER:
result = [ ]
for w in vehicles:
    if ’a’ in w:
        result.append(w)

With both of these the answer is calculated and stored in result.

This problem has six parts. The six problems start on the next page.
PART A (3 pts)
Given the list of strings named `words`, where each string is one word, calculate a list of tuples, where each tuple has two items, 1) the word and 2) the length of the word. The tuples should appear with the words in the same order they are in the list words. The answer should be stored in the variable `result`.
Using the list `lst` below, `result`'s value would be: `[('cat', 3), ('cake', 4), ('dolphin', 7), ('sugar', 5)]`

Write your code below and be sure `result`'s value is the answer.
```
words = ['cat', 'cake', 'dolphin', 'sugar' ]
```

PART B (4 pts)
You are given the list of strings named `words` and the list of strings named `colors`, where each string in each list is one word. The $k$th word in the list `colors` is the color of the $k$th item in the list `words`.
Calculate a list of tuples, where each tuple has two items, 1) a word from the list words and 2) the color of that item. The tuples should appear with the words in the same order they are in the list words. The answer should be stored in the variable `result`.
Using the list `lst` below, `result`'s value would be: `[('tire', 'black'), ('cat', 'orange'), ('apple', 'red'), ('cake', 'blue'), ('dolphin', 'gray')]`

Write your code below and be sure `result`'s value is the answer.
```
words = ['tire', 'cat', 'apple', 'cake', 'dolphin']
colors = ['black','orange','red', 'blue', 'gray']
```
PART C (3 pts)
You are given the list of strings `lsta`, and the list of strings `lstb`, where each string in both lists is one word. Calculate a sorted list of the unique words from the lists `lsta` and `lstb` that are in both lists. The answer should be stored in the variable `result`.
Using the lists below, `result` would be: `[‘bag’, ‘line’]`. Note there is only one occurrence of each word and the list is sorted.

Write your code below and be sure result’s value is the answer.
```
```

PART D (3 pts)
You are given the list of strings `lsta` and the list of strings `lstb`, where each string in both lists is one word. Calculate a sorted list of the unique words from the lists `lsta` and `lstb` that are in one of the two lists, but not in both of the lists. The answer should be stored in the variable `result`.
Using the lists below, `result` would be: `[‘blue’, ‘car’, ‘jug’, ‘tip’, ‘tree’]`. Note there is only one occurrence of each word and the list is sorted.

Write your code below and be sure result’s value is the answer.
```
```
PART E (4 pts)

Here you are given a dictionary named \( d \), in which each key is mapped to a list of integers. Calculate a sorted list of the first integers from each list in the dictionary. The answer should be stored in the variable \( \text{result} \).

Using the dictionary \( d \) below, \( \text{result} \) would be: \([1, 2, 6, 7]\). Note that 6 is the first integer in the list that "N" maps to, 7 is the first integer in the list that "B" maps to, 2 for "Y", and 1 for "S". Also note the calculated list is sorted.

Write your code below and be sure result’s value is the answer.

\[
d = \{"N":[6,2,1], "B":[7,8], "Y":[2,3,9], "S":[1,7,1,4]\}
\]
PROBLEM 3: (Teams and Players (31 pts))

This problem is about data related to teams of players from schools for an unspecified sport. There are five functions to write in this part. Your functions should work for any valid data, not just the examples shown.

All five problems have datalist as one of the parameters. The parameter datalist is a list of lists, with each inner list having the following four items: 1) a string representing one game between two schools in the format school1name, followed by a dash "-", followed by school2name, the two school names are in alphabetical order, 2) a string representing a school name, 3) a string representing a player’s name from the school name in 2), and 4) an integer for the number of points this player scored in this game. We will assume player’s names are unique between all schools, and two schools only play one game against each other.

For example, assume datalist is the lists of lists shown below. Note that the first item in the first inner list is the game "duke-unc" meaning the one game played between the teams ‘duke’ and ‘unc’. The second item in the first inner list is the school ‘duke’, the third item is the duke player ‘bwalton’, and the fourth item indicates player ‘bwalton’ scored 19 points in this game.

datalist = [[‘duke-unc’, ’duke’, ’bwalton’, 19],
            [‘miami-unc’, ’miami’, ’gsmith’, 12],
            [‘duke-miami’, ’duke’, ’nhua’, 17],
            [‘duke-miami’, ’duke’, ’bwalton’, 14],
            [‘duke-unc’, ’duke’, ’ppoe’, 6],
            [‘miami-unc’, ’unc’, ’yxiao’, 24],
            [‘ncsu-unc’, ’ncsu’, ’byavat’, 18],
            [‘duke-ncsu’, ’duke’, ’bwalton’, 28],
            [‘duke-unc’, ’unc’, ’mlowe’, 15],
            [‘miami-unc’, ’miami’, ’rtopp’, 10],
            [‘miami-unc’, ’unc’, ’yxiao’, 12],
            [‘duke-miami’, ’miami’, ’rtopp’, 21],
            [‘duke-miami’, ’duke’, ’ppoe’, 10],

Go to the next page to start Part A of this problem.
Part A (6 pts)
Write the function named `playerList` that has two parameters. The first parameter is named `datalist`, which is a list of lists in the format described earlier, and the second parameter is a string named `school`, for the name of a school.

We repeat the format of `datalist`, a list of lists, with each inner list having the following four items: 1) a string representing one game between two schools in the format `school1name`, followed by a ”-”, followed by `school2name`, with the two school names in alphabetical order, 2) a string representing a school name, 3) a string representing a player’s name from the school name in 2), and 4) an integer for the number of points this player scored in this game.

This function returns a sorted list of the unique names of players from `school`. Note that all players from all teams have at least one entry in `datalist`.

For example, assume `datalist` is the list of lists shown on the first page of this problem. We give several examples of calls to this function.

<table>
<thead>
<tr>
<th>call</th>
<th>returns</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>playerList(datalist, &quot;duke&quot;)</code></td>
<td>['bwalton', 'nhua', 'ppoe']</td>
</tr>
<tr>
<td><code>playerList(datalist, &quot;unc&quot;)</code></td>
<td>['mlowe', 'yxiao']</td>
</tr>
<tr>
<td><code>playerList(datalist, &quot;miami&quot;)</code></td>
<td>['gsmith', 'rtopp']</td>
</tr>
</tbody>
</table>

Complete the function below.

```python
def playerList(datalist, school):
```
Part B (6 pts)
Write the function named `dictPlayerTotalPoints` that has two parameters, The first parameter is named `datalist`, which is a list of lists in the format described earlier, and the second parameter is a string named `school`.

We repeat the format of `datalist`, a list of lists, with each inner list having the following four items: 1) a string representing one game between two schools in the format school1name, followed by "-", followed by school2name, with the two school names in alphabetical order, 2) a string representing a school name, 3) a string representing a player’s name from the school name in 2), and 4) an integer for the number of points this player scored in this game.

This function returns a dictionary of players from school, with each player mapped to the total number of points they scored in all games. We give several examples of calls to this function.

<table>
<thead>
<tr>
<th>call</th>
<th>returns</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>dictPlayerTotalPoints(datalist, 'unc')</code></td>
<td>{'yxiao': 56.0, 'mlowe': 15.0}</td>
</tr>
<tr>
<td><code>dictPlayerTotalPoints(datalist, 'ncsu')</code></td>
<td>{'byavat': 18}</td>
</tr>
<tr>
<td><code>dictPlayerTotalPoints(datalist, 'duke')</code></td>
<td>{'bwalton': 61, 'nhua': 17, 'ppoe': 16}</td>
</tr>
</tbody>
</table>

Complete the function below.

```python
def dictPlayerTotalPoints(datalist, school):
```

Write the function named `playerMostPoints` that has two parameters. The first parameter is named `datalist`, which is a list of lists in the format described earlier, and the second parameter is a string named `school`.

We repeat the format of `datalist`, a list of lists, with each inner list having the following four items: 1) a string representing one game between two schools in the format `school1name`, followed by "-", followed by `school2name`, with the two school names in alphabetical order, 2) a string representing a school name, 3) a string representing a player’s name from the school name in 2), and 4) an integer for the number of points this player scored in this game.

This function returns a string that is the name of the player at the school `school` that scored the most points over all of this school’s games. Assume there is only one player with the most points. We give several examples of calls to this function.

<table>
<thead>
<tr>
<th>call</th>
<th>returns</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>playerMostPoints(datalist, 'miami')</code></td>
<td>'rtopp'</td>
</tr>
<tr>
<td><code>playerMostPoints(datalist, 'duke')</code></td>
<td>'bwalton'</td>
</tr>
<tr>
<td><code>playerMostPoints(datalist, 'unc')</code></td>
<td>'yxiao'</td>
</tr>
</tbody>
</table>

In writing this function, you MUST call the function `dictPlayerTotalPoints` from Part B in a meaningful way. Complete the function below.

```python
def playerMostPoints(datalist, school):
```
Part D (6 pts)

Write the function named `dictGameToListPlayers` that has one parameter named `datalist`, which is a list of lists in the format described earlier.

We repeat the format of `datalist`, a list of lists, with each inner list having the following four items: 1) a string representing one game between two schools in the format school1name, followed by "-", followed by school2name, with the two school names in alphabetical order, 2) a string representing a school name, 3) a string representing a player’s name from the school name in 2), and 4) an integer for the number of points this player scored in this game.

This function returns a dictionary that maps the game (a string of two school names with a "dash" between them) to a list of the players from both teams that played in that game. We give an example of a call to this function. The call, `dictGameToListPlayers(datalist)` returns the dictionary:

```python
{'duke-unc': ['bwalton', 'ppoe', 'mlow', 'yxiao'],
 'miami-unc': ['gsmith', 'yxiao', 'rtopp', 'yxiao'],
 'duke-miami': ['nhua', 'bwalton', 'rtopp', 'ppoe'],
 'ncsu-unc': ['byavat'],
 'duke-ncsu': ['bwalton'] }
```

Complete the function below.

```python
def dictGameToListPlayers(datalist):
```
Part E (7 pts)
Write the function named **playersDidNotPlay** that has two parameters, The first parameter is named **datalist**, which is a list of lists in the format described earlier, and the second parameter is a string named **game**.

We repeat the format of datalist, a list of lists, with each inner list having the following four items: 1) a string representing one game between two schools in the format school1name, followed by ",", followed by school2name, with the two school names in alphabetical order, 2) a string representing a school name, 3) a string representing a player’s name from the school name in 2), and 4) an integer for the number of points this player scored in this game.

This function returns a **sorted** list of **unique** player names from the two schools for that game that did not play in that game. Assume all the players on a team are in datalist. We give several examples of calls to this function.

<table>
<thead>
<tr>
<th>call</th>
<th>returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>playersDidNotPlay(datalist, 'duke-unc')</td>
<td>['nhua']</td>
</tr>
<tr>
<td>playersDidNotPlay(datalist, 'duke-ncsu')</td>
<td>['byavat', 'nhua', 'ppoe']</td>
</tr>
</tbody>
</table>

You must call at least two functions in parts A)-D) in meaningful ways in solving this problem. Complete the function below.

```python
def playersDidNotPlay(datalist, game):
```