PROBLEM 1: (What is the output? (16 pts) (8 minutes))

For the following code, write the output to the right of each print statement.

```
lista = ['dogs', 'hay', 'cats', 'go']
print(sorted(lista))
#---------------------------------------------

lista = ['golf', 'red', 'golf', 'blue', 'red', 'golf']
listb = [(w, lista.count(w)) for w in set(lista)]
print(sorted(listb))
#---------------------------------------------

lista = [8, 47, 25, 68]
listb = sorted(lista, reverse=True)
print(listb)
#---------------------------------------------

lista = ['green', 'red', 'mint']
listb = sorted(lista, key=len)
print(listb)
#---------------------------------------------

lista = [(9,6), (3,7), (4,7,2)]
listb = sorted(lista, key=min)
print(listb)
#---------------------------------------------

listb = sorted(lista, key=lambda x:x[2])
print(listb)
#---------------------------------------------

d = {'X': [7, 4], 'N':[3, 8], 'P':[5,6]}
ans = sorted(d.keys())
print(ans)
ans = sorted(d.items(), key=lambda x:x[1][1])
print(ans[-1])
```
Complete the following functions. This problem has three parts.

PART A (6 pts) (6 minutes)
Write the function named `sumItUp` that has one parameter, a dictionary named `somedict`, where each key is a string mapped to a list of one or more integers.
This function returns the sum of all the integers in the dictionaries values.
For example, assume the dictionary `d` is the following:

```
somedict = {"C": [2, 1], "P": [2, 3], "K": [4]}
```

The call `sumItUp(somedict)` would return 12, which is the sum of 2+1+2+3+4, the sum of all the integers in the lists.
Complete the function below.

```python
def sumItUp(somedict):
```
PART B (6 pts) (6 minutes)
Write the function named `inCommon` that has two string parameters named `word1` and `word2`. This function returns a sorted list of the unique letters in common between `word1` and `word2`. We give several examples of calls to this function.

<table>
<thead>
<tr>
<th>call</th>
<th>returns</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>inCommon(&quot;wonderful&quot;,&quot;something&quot;)</code></td>
<td>['e', 'n', 'o']</td>
</tr>
<tr>
<td><code>inCommon(&quot;cat&quot;,&quot;ark&quot;)</code></td>
<td>['a']</td>
</tr>
<tr>
<td><code>inCommon(&quot;sweeter&quot;,&quot;teacher&quot;)</code></td>
<td>['e', 'r', 't']</td>
</tr>
</tbody>
</table>

Complete the function below.

```python
def inCommon(word1, word2):
```

```python
def inCommon(word1, word2):
```
PART C (6 pts) (6 minutes)
Write the function named putInOrder that has one parameter named lista that is a list of tuples, where each tuple has three items in this order: an integer, a word that is a string, and an integer.

This function returns a list of the tuples sorted in the following way:

1. sorted by the word in each tuple, in reverse alphabetical order
2. break ties by sorting by the last integer in each tuple, in reverse numeric order
3. after both 1) and 2) break ties by sorting by the first integer in numeric order

For example, the call
putInOrder([(9, 'ant', 3), (4, 'tie', 5), (4, 'tie', 7), (6, 'ant', 3), (7, 'ant', 3)]) returns the list
[(4, 'tie', 7), (4, 'tie', 5), (6, 'ant', 3), (7, 'ant', 3), (9, 'ant', 3)]
Complete the function below.

def putInOrder(lista):
PROBLEM 3: (Favorite Foods at the market (40 pts) (40 minutes))

This problem is about data related to favorite foods at a market.
There are five functions to write in this part. Your functions should work for any valid data, not just the examples shown.
Most of the problems have `datalist` as one of the parameters. The parameter `datalist` is a list of lists, with each inner list representing information about one food item as 1) a string representing a food item 2) a float representing the price of the food item and 3) a list of strings of names of people who like the food item.
Note that each food item appears only in one innerlist. Within any list of names related to a food item, the names are unique.
For example, assume `datalist` is the lists of lists shown below. The first inner list represents the food peaches, which costs $8.00, and is liked by four people: Sarah, Kyren, Raj, and Michael. The second inner list represents sweet corn, which costs $10.75, and is liked by five people: Raj, Myra Sue, Jiao, Kyren, and Purnima.

```
datalist = [['peaches', 8.0, ['Sarah', 'Kyren', 'Raj', 'Michael']],
            ['sweet corn', 10.75, ['Raj', 'Myra Sue', 'Jiao', 'Kyren', 'Purnima']],
            ['okra', 6.5, ['Michael', 'Susan', 'Ajani']],
            ['corn bread', 4.25, ['Jiao', 'Sarah', 'Susan', 'Michael']],
            ['watermelon', 3.75, ['Myra Sue', 'Sarah', 'Raj', 'Ajani', 'Purnima']],
            ['shrimp', 12.5, ['Ajani', 'Susan', 'Kyren', 'Xiaowei']],
            ['field peas', 4.5, ['Jiao', 'Sarah', 'Michael', 'Myra Sue']],
            ['blackeyed peas', 4.8, ['Raj', 'Purnima', 'Susan', 'Jiao']],
            ['chicken', 10.5, ['Raj', 'Myra Sue', 'Jiao', 'Susan', 'Kyren']]
```

In solving the problems that follow, you may call any of the other functions in this problem. Go to the next page to start Part A of this problem.
Part A (8 pts) (8 minutes)

Write the function named **likedCostMoreThan** that has two parameters named **datalist**, which is a list of lists in the format described earlier, and **amount**, which is a float.

We repeat the format of parameter datalist, which is a list of lists. Each inner list is information about one food item as 1) a string representing the food item 2) a float representing the price of the food item and 3) a list of names of people who like the food item.

This function returns a sorted list of unique names of people who like at least one food item in datalist that costs amount or more.

For example, assume datalist is the list of lists shown on the first page of this problem.

<table>
<thead>
<tr>
<th>call</th>
<th>returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>likedCostMoreThan(datalist, 11.00)</td>
<td>['Ajani', 'Kyren', 'Susan', 'Xiaowei']</td>
</tr>
<tr>
<td>likedCostMoreThan(datalist, 9.00)</td>
<td>['Ajani', 'Jiao', 'Kyren', 'Myra Sue', 'Purnima', 'Raj', 'Susan', 'Xiaowei']</td>
</tr>
</tbody>
</table>

Complete the function below.

```python
def likedCostMoreThan(datalist, amount):
```
Part B (8 pts) (8 minutes)

Write the function named `itemsNotInBoth` 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 named `storeItems` is a list of food items available at a grocery store.

We repeat the format of parameter `datalist`, which is a list of lists. Each inner list is information about one food item as 1) a string representing the food item 2) a float representing the price of the food item and 3) a list of names of people who like the food item.

This function returns a sorted unique list of those food items that are in `datalist` or `storeItems` but not both. Using the datalist shown on the first page of this problem, if the list `storeItems` is `"lemons", "sweet corn", "watermelon", "blackeyed peas", "flounder", "corn bread"` then the call `itemsNotInBoth(datalist, storeItems)` would return the list: `[‘chicken’, ‘field peas’, ‘flounder’, ‘lemons’, ‘okra’, ‘peaches’, ‘shrimp’].`

Complete the function below.

```python
def itemsNotInBoth(datalist, storeItems):
```
Part C (8 pts) (8 minutes)

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

We repeat the format of parameter `datalist`, which is a list of lists. Each inner list is information about one food item as 1) a string representing the food item 2) a float representing the price of the food item and 3) a list of names of people who like the food item.

This function returns a dictionary of each person’s name mapped to the list of food items from `datalist` they like. For example, using the `datalist` described on the first page of this problem, the call `dictPersonToItems` results in the dictionary:

```python
dictPersonToItems(datalist):
```

```python
{'Sarah': ['peaches', 'corn bread', 'watermelon', 'field peas'],
 'Kyren': ['peaches', 'sweet corn', 'shrimp', 'chicken'],
 'Raj': ['peaches', 'sweet corn', 'watermelon', 'blackeyed peas', 'chicken'],
 'Michael': ['peaches', 'okra', 'corn bread', 'field peas'],
 'Myra Sue': ['sweet corn', 'watermelon', 'field peas', 'chicken'],
 'Jiao': ['sweet corn', 'corn bread', 'field peas', 'blackeyed peas', 'chicken'],
 'Purnima': ['sweet corn', 'watermelon', 'blackeyed peas'],
 'Susan': ['okra', 'corn bread', 'shrimp', 'blackeyed peas', 'chicken'],
 'Ajani': ['okra', 'watermelon', 'shrimp'],
 'Xiaowei': ['shrimp']}
```
Part D (8 pts) (8 minutes)
Write the function named **mostLiked** that has one parameter named **datalist**, which a list of lists in the format described earlier.

We repeat the format of parameter datalist, which is a list of lists. Each inner list is information about one food item as 1) a string representing the food item 2) a float representing the price of the food item and 3) a list of names of people who like the food item.

This function returns a **sorted** list of unique names of people that liked the most food items in datalist.

For example, using the datalist on the first page of this problem, the call **mostLiked(datalist)** returns the list ['Jiao’, ’Raj’, ’Susan’], as each of them liked five food items, which was the largest number of liked food items.

Complete the function below.

```python
def mostLiked(datalist):
```
Part E (8 pts) (8 minutes)

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

We repeat the format of parameter `datalist`, which is a list of lists. Each inner list is information about one food item as 1) a string representing the food item 2) a float representing the price of the food item and 3) a list of names of people who like the food item.

This function returns a sorted list of tuples of pairs of numbers, where the first number is an integer, say N, and the second number is the number of food items that were liked that cost at least N and less than N+1. A food item is counted as many times as it is liked. Only those tuples with second number greater than 0 are in the list. The tuples are sorted on the second number in reverse order, with ties broken by the first number.

For example, consider the `datalist` example given at the beginning of this problem. The call `howManyInRange(datalist)` would return the list: `[(4, 12), (10, 10), (3, 5), (8, 4), (12, 4), (6, 3)]`. In the first tuple (4, 12), the 4 shows there are food items that cost at least 4.0 and less than 5.0, and the 12 shows the food items are liked by 12 people. Note cornbread is $4.25 (liked by 4 people), field peas is $4.50 (liked by 4 people) and blackeyed peas is $4.80 (liked by 4 people). The tuple (4, 12) is listed first because 12 is the largest second number.

```python
def howManyInRange(datalist):
```