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

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

```python
lista = ['pom', 'top', 'bee', 'goat']
print(sorted(lista))
#

lista = ['bee','to','word','ate']
listb = [w, len(w) for w in set(lista)]
print(sorted(listb))
#

lista = [35, 7, 46, 28]
listb = sorted(lista, reverse=True)
print(listb)
#

lista = ['purple', 'mop', 'couch', 'jot', 'word']
listb = sorted(lista, key=len)
print(listb)
#

lista = [(7,5,1),(4,13),(6,11,2)]
listb = sorted(lista, key=max)
print(listb)
#

lista = [(8,3,6),(2,9,4),(5,1,7)]
listb = sorted(lista, key=lambda x:x[1])
print(listb)
#

d = {'R': [9, 3], 'N':[6,1], 'P':[7,4]}
ans = sorted(d.keys())
print(ans)
ans = sorted(d.items(), key=lambda x:x[1][1])
print(ans[0])
```

OUTPUT:
PROBLEM 2:  (Short problems (18 pts) (18 minutes) )

Complete the following functions. This problem has three parts. Your functions should work for any valid data, not just the examples shown.

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

dictnums = {"B":[6,5,2], "J":[7,3,9,1], "R":[3,4], "Y":[8,5,1]}

The call addMaxes would return 27, which is the sum of 6+9+4+8, the sum of the largest integers from each of the lists. Complete the function below.

def addMaxes(dictnums):
PART B (6 pts) (6 minutes)
Write the function named `makeWord` that has two string parameters named `word1` and `word2`. This function returns a **tuple of two items**. The first item is True if you can form `word2` from the letters in `word1`, and False if you cannot. Note each letter from `word1` can only be used once in `word2`. The second item in the tuple is the sorted listed of letters that are in `word2` that were not found in `word1`. We give several examples of calls to this function.

<table>
<thead>
<tr>
<th>call</th>
<th>returns</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>makeWord(&quot;wonderful&quot;, &quot;fun&quot;)</code></td>
<td>(True, [])</td>
</tr>
<tr>
<td><code>makeWord(&quot;something&quot;, &quot;float&quot;)</code></td>
<td>(False, ['a', 'f', 'l'])</td>
</tr>
<tr>
<td><code>makeWord(&quot;missprint&quot;, &quot;mississippi&quot;)</code></td>
<td>(False, ['i', 'i', 'p', 's', 's'])</td>
</tr>
</tbody>
</table>

*Note: Need 4 i's, so missing two i's*

Complete the function below.

```python
def makeWord(word1, word2):
```
PART C (6 pts) (6 minutes)

Write the function named orderTuples that has one parameter named lista that is a list of tuples, where each tuple has three strings.

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

1. sorted by the first string in each tuple
2. break ties by sorting with the last string in each tuple, in reverse order
3. after 1) and 2) break ties sorting by the second string in each tuple, in reverse order

For example, the call orderTuples([(‘j’,’u’,’e’), (‘k’,’p’,’e’), (‘k’,’s’,’e’), (‘f’,’u’,’e’), (‘f’,’c’,’e’), (‘f’,’c’,’d’)])
returns the list:
[(‘f’,’u’,’e’), (‘f’,’c’,’e’), (‘f’,’c’,’d’), (‘j’,’u’,’e’), (‘k’,’s’,’e’), (‘k’,’p’,’e’)]

Complete the function below.

def orderTuples(lista):
This problem is about data related to nursing homes with patients. 

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 patient and one of their drugs. More specifically, each inner list has:
1) a string representing a patient name,
2) a string representing the name of the nursing home where the patient lives,
3) a string representing one of the patient’s drugs and
4) a float indicating how many years a patient has taken this drug.

For example, assume `datalist` is the lists of lists shown below. The first inner list represents the patient named Williams, who lives in the nursing home named Duke Manor, who has been taking the drug Aspirin for 30.6 years.

```
datalist = [['Williams', 'Duke Manor', 'Aspirin', 30.6]
            ['Zhang', 'Duke Manor', 'Metformin', 22.4]
            ['Feyer', 'Hillcrest', 'Maalox', 12.5]
            ['Moorthy', 'PruittHealth', 'Lisinopril', 2.3]
            ['Shippey', 'The Forest at Duke', 'Maalox', 24.5]
            ['Lippard', 'Durham Ridge', 'Levatol', 6.7]
            ['Ludwig', 'Brookdale', 'Maalox', 16.8]
            ['Feyer', 'Hillcrest', 'Vitamin D', 4.7]
            ['Moorthy', 'PruittHealth', 'Hygroton', 2.3]
            ['Cole', 'PruittHealth', 'Vitamin D', 34.8]
            ['Shippey', 'The Forest at Duke', 'Hygroton', 2.3]
            ['Lippard', 'Durham Ridge', 'Maalox', 16.2]
            ['Dias', 'Durham Ridge', 'Maalox', 1.8]
            ['Zhang', 'Duke Manor', 'Hygroton', 4.1]
            ['Feyer', 'Hillcrest', 'Metformin', 7.8]
            ['Shippey', 'The Forest at Duke', 'Levatol', 1.3]
            ['Velasco', 'Brookdale', 'Vitamin D', 3.6]
            ['Moorthy', 'PruittHealth', 'Vitamin D', 13.6]
            ['Shippey', 'The Forest at Duke', 'Metformin', 22.4]
            ['Lippard', 'Durham Ridge', 'Vitamin D', 5.9]
            ['Ludwig', 'Brookdale', 'Metformin', 2.6]]
```

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 `drugTakenLongest` 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 has 1) a string representing a patient name, 2) a string representing the name of the nursing home where the patient lives, 3) a string representing one of the patient’s drugs and 4) a float indicating how many years a patient has taken this drug.

This function calculates the drug that has been taken for the longest amount of time and returns a tuple of three items: 1) the name of the patient 2) the drug and 3) the number of years the patient has been taking this drug. If there is a tie, return the information for any one in the tie.

For example, assume `datalist` is the list of lists shown on the first page of this problem. The call `drugTakenLongest(datalist)` returns the tuple: (`'Cole'`, `'Vitamin D'`, 34.8)

Complete the function below.

```python
def drugTakenLongest(datalist):
```
Part B (8 pts) (8 minutes)

Write the function named `ourTopNHomes` 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 `topHomes` is a list of the top nursing homes.

We repeat the format of parameter `datalist`, which is a list of lists. Each inner list has 1) a string representing a patient name, 2) a string representing the name of the nursing home where the patient lives, 3) a string representing one of the patient’s drugs and 4) a float indicating how many years a patient has taken this drug.

This function returns a sorted unique list of those nursing homes that are in `datalist` that are also in `topHomes`. Using the `datalist` shown on the first page of this problem, if the list `topHomes` is `["Glenaire", "Capital Nursing", "Brookdale", "The Cardinal", "The Forest at Duke", "SunnyBrook", "PruittHealth", "The Templeton"]` then the call `ourTopNHomes(datalist, topHomes)` would return the list: `[‘Brookdale’, ‘PruittHealth’, ‘The Forest at Duke’]`

Complete the function below.

```python
def ourTopNHomes(datalist, topHomes):
```
Part C (8 pts) (8 minutes)

Write the function named `drugTakenMost` 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 has 1) a string representing a patient name, 2) a string representing the name of the nursing home where the patient lives, 3) a string representing one of the patient’s drugs and 4) a float indicating how many years a patient has taken this drug.

This function returns the name of the drug taken by the most patients. If there is a tie, then return any of the drugs that tied. For example, using the `datalist` described on the first page of this problem, the call `drugTakenMost(datalist)` returns "Vitamin D", as it is taken by five patients, the most of any drug.

```python
def drugTakenMost(datalist):
```

Part D (8 pts) (8 minutes)

Write the function named `dictHomeToPatients` 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 has 1) a string representing a patient name, 2) a string representing the name of the nursing home where the patient lives, 3) a string representing one of the patient’s drugs and 4) a float indicating how many years a patient has taken this drug.

This function returns a `dictionary` where each key is the name of a nursing home and the key is mapped to a sorted list of its unique patients.

For example, using the `datalist` on the first page of this problem, the call `dictHomeToPatients(datalist)` returns the dictionary:

```python
```

Complete the function below.

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

Write the function named **homeMostPatients** 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 has 1) a string representing a patient name, 2) a string representing the name of the nursing home where the patient lives, 3) a string representing one of the patient’s drugs and 4) a float indicating how many years a patient has taken this drug.

This function returns the name of the nursing home with the most patients. If there is a tie, of those nursing homes that tie, return the first such one in alphabetical order.

For example, consider the datalist example given at the beginning of this problem. The call **homeMostPatients(datalist)** calculates that there are four nursing homes with two patients each, the most number for any nursing home. Those are: Duke Manor, PruittHealth, Durham Ridge and Brookdale. The string ”Brookdale” is returned.

You may call any of the other functions in this problem to solve this part.

```python
def homeMostPatients(datalist):
```