Lab14
Analyze customer data

Overall Background

While true analysis of customer data is normally done in a proper database with proper database queries, we can ask and answer a remarkable number of simple questions using a simple text file of data and some basic code that uses lists and/or dictionaries to help us organize the data in that file.

In this assignment you will be working with a file of 30,000 fake customers. (Yes, it really is fake information which was generated randomly, so no trying to sell it after class! Heh.)

- FakeCustomerData.txt (right click and choose "save as")

In order to complete this assignment, you should create a file called lab14.py. Complete the following steps in the order listed.

1. **Add comments and the printName() function:**

   Create typical header comments and place them at the top of your lab14.py file. Add the typical printName() function that prints your name(s). In addition, be sure to write function comments for all functions in this lab, including the printName() function you just added.

2. **Write the getStateDistribution() method:**

- BACKGROUND :
  - In order to understand where our customers come from we might want to search to see how many customers come from each state.
- ACTION :
  - Add to lab14.py the method called getStateDistribution().
  - This method will be a *specialized* method that works on our one file so it takes no parameters.
  - This method:
    - opens the file "FakeCustomerData.txt" and reads it in line by line

        - for each line (not including the header line) it splits the data...
        - pulls out the state for that customer...
        - and keeps track of the number of times we have seen each state.
    - This method **sorts and prints** the final counts from each state (see the screenshots following this set of instructions
- TESTING :
    - Consider the screenshot below and compare your counts to the partial list of counts listed there.

```
>>> getStateDistributions()
AK -> 120
AL -> 462
AR -> 363
AZ -> 453
CA -> 3130

...

TX -> 2322
UT -> 251
VA -> 690
VT -> 126
WA -> 634
WI -> 643
WV -> 177
WY -> 75
```

3. **Write the getColumnDistribution(filename,columnNum) method.**

- BACKGROUND :
    - The method above was designed to work one specific file by hardcoding the name of the file and the column number right into the code. But once it is completed we can imagine wanting to get a similar distribution of information on any column of data. Or for that matter, on any file.
- ACTION :
    - Add a *generalized* method called getColumnDistribution().
    - This method takes one String (filename) and one integer (columnNum) as parameters.
    - his method:
        - opens the given file and reads it in line by line
        - for each line (not including the header line) it splits the data...
        - pulls out the appropriate column for that customer...

        - and keeps track of the number of times we have seen that piece of data.
    - This method **sorts and prints** the final counts from each key observed (see the screenshots following this set of instructions

NOTE: If the user give us a column that doesn't exist, your method should print a statement explaining that no information is available for that column number.

```
>>> getColumnDistributions("FakeCustomerData.txt",20)
There is no column number 20
```

- TESTING :
    - Again, consider the following screenshot to help you check your code

```
>>> getColumnDistributions("FakeCustomerData.txt",1)
female -> 15167
male -> 14833
>>> getColumnDistributions("FakeCustomerData.txt",14)
MasterCard -> 14868
Visa -> 15132
```

4. **Write the getBirthYearDistribution() method.**

- BACKGROUND :
    - Method #2 looks like it is going to be really useful, but it isn't as helpful as we might really like because sometimes the interesting data is nested inside other data. For example, if I wanted to see how old my customers are (by looking at the year they were born) I could try to use method #2 using column 13 (birthdate) but the problem is that this only stores information for the entire column so I get a LONG dump of data that starts with:

```
>>> getColumnDistributions("FakeCustomerData.txt",13)
1/1/1941 -> 3
1/1/1942 -> 2
1/1/1943 -> 3
1/1/1945 -> 1
```

- ACTION :
    - Add to lab14.py the method called getBirthYearDistribution().
    - This method will be a *specialized* method that works on our one file so it takes no parameters.
    - This method:
        - opens the file "FakeCustomerData.txt" and reads it in line by line

        - for each line (not including the header line) it splits the data...
        - pulls out the birthdate for that customer...
        - completes another split on the birthdate to pull out the birth YEAR...
        - and keeps track of the number of times we have seen each year.
    - This method **sorts and prints** the final counts from each year (see the screenshots following this set of instructions
- TESTING :
    - Consider the screenshot below and compare your counts to the partial list of counts listed there.

```
1940 -> 651
1941 -> 677
1942 -> 660
1943 -> 663
```

...

```
1980 -> 637
1981 -> 638
1982 -> 693
1983 -> 722
1984 -> 660
1985 -> 622
```

BONUS - **Revise the getColumnDistribution(filename,columnNum) method.**

It's a pain to have to know which column number we want to analyze. Sometimes I know what the header is, but not what the column number is.

Revise getColumnDistribution() from part 2. It should CONTINUE to work EXACTLY as it did before. That is, it should work when I pass in a string for a filename and an integer for the column number. However, IN ADDITION (that was a lot of capitalized words in one paragraph) it should work if I pass in a string as a filename and a string as a column header. In this case, your code would look at all of the headers to see if this string is one of the headers. If it is, it uses that column number:

```
>>> getColumnDistributions("FakeCustomerData.txt","State")
AK -> 120
AL -> 462
AR -> 363
AZ -> 453
```

If it isn't, then it prints an appropriate message:

```
>>> getColumnDistributions("FakeCustomerData.txt","CC")
There is no column labeled CC
>>> getColumnDistributions("FakeCustomerData.txt",5.2)
The input 5.2 is unrecognized.
```

Review your textbook for how to find out what type a particular variable is holding.

Final Submission

This week I will again ask you to submit your code for electronic grading, using the eLearning submission system.

Follow the directions on the system to select the appropriate course and assignment and submit

- lab14.py

If you worked with a partner, make sure that both you and your partner's names are in the comment header at the top of the file ***and*** in the printName() function.