CPSC 223P Python Programming

Spring 2020

Instructor: Dr. Tseng-Ching J. Shen

Take Home Final Exam
Due Date: May 11

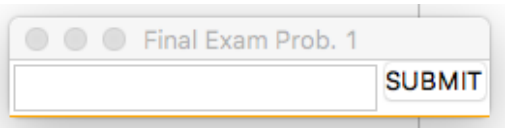
Note:

1. Please create one sub-folder for each problem to put the source files.
2. Please add a comment to include the following information: your CWID, name, and problem # at the beginning of each source file you create.
3. Please zip all source files you create into a file named as <your first name>_<your last name>.zip after you are done.
4. Upload the zip file to Titanium.

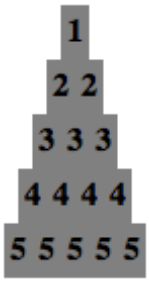
[Problem 1] Python Tkinter Library (50%)

In this problem, you will create a Python script to display a 'triangle' (defined below) the height (number of layers) of which will be input by the user.

Here is how the script is supposed to work. When the application starts, it shows a screen where three widgets: *Entry*, *Button* and *Frame* widget are displayed. The layout of the widgets is shown as follows. Since the Frame widget initially contains nothing, it's 'invisible' on the screen.



User will enter the number of levels in the Entry widget and click on the Button widget. Once the Button widget is clicked, the script will display a 'Label widget-triangle' inside the Frame widget. The triangle consists of n (entered by the user) levels and the i-th level of the triangle will consist of i Label widgets each of which displays the given i on it. Below is an example of such a triangle that consists of 5 levels of Label widgets.



Pleas use only the Pack Layout Manager for your implementation and your implementation needs to meet the following requirements:

1. The Entry widget will remain the same size and stay the top left corner of the screen when you enlarge the window.
2. The Button widget will remain the same size and stay next to the right-hand side of the Entry widget when the window is enlarged.
3. The 'triangle' should always stay in the center of the Frame widget when you enlarge the window.

Problem 2 Python Pandas Library (50%)

In this problem, you will develop a Pandas application that will

- Read and parse two input files,
- Create on a DataFrame object with the parsed data, and
- Perform some analysis on the DataFrame object.

The columns of DataFrame the application will create includes three customer related columns: '*CUSTOMER ID', 'GENDER', 'AGE'* and a various number of columns each of which is named after one product name. You can decide your own set of product names for your testing. Each row of the DataFrame consists of the order information for a given customer. The value of a given product-name column for a given row contains the quantity the customer orders for the corresponding product. For example, the following is one such a DataFrame:

CUSTIMER ID	GENDER	AGE	PROD_1	PROD_2	PROD_3	PROD_4
56748	M	35	5	0	0	4

This DataFrame has seven columns and four of them are named after product name 'PROD_1', 'PROD_2', 'PROD_3' and 'PROD_4'. Row with 'CUSTOMER ID' column = 56748 has the order information for customer 56748. It says that he ordered 5 pieces of 'PROD_1' product and 4 pieces of of 'PROD_4' product respectively.

The application will perform the following steps to create the DataFrame object.

1. Process the customer input file. The file consists of records having the following format:

   CUSTOMER ID : <cust_id> ; GENDER : <gender> ; AGE : <age>

   Your application will parse each record and retrieve cust_id, gender, and age for the customer with that cust_id. Once data is retrieved, the application will insert the data into a DataFrame object.

2. Process the order input file. The file consists of records having the following format:

   CUSTOMER ID : <cust_id> ; PRODUCT NAME : <prod_name> ; QUANTITY : <qty>

   Your application will parse each record and retrieve cust_id, prod_name and qty. With the cust_id, your application will query the DataFrame to get the customer's record and set the value of the column named prod_name with qty. Note that the DataFrame object should automatically add the column when you intend to set value on a missing column.

After the DataFrame object is created, you are ready to perform some analysis. Before you perform analysis on the DataFrame object, you need to set the corresponding column value to 0 for customers who did not order any for a given product name. You can use DataFrame fillna() method to accomplish this. Please browse the web for the documentation of the method and apply here.

You will write two functions to be used for your analysis. The first function takes two product names as inputs and calculates the correlation coefficient between these two products. What does this correlation coefficient tell you about these two products? Could you come up with some sample data to illustrate your points?

You will implement another function also taking two product names as inputs. However, this function will first group the DataFrame records using the GENDER column. For each group, the function will calculate the correlation coefficient between two given product. Now what does the number tell us?