CS 2316
Pair Homework 9 – Newsvendor Inventory Policy

Due: Thursday, July 14th before 11:55 PM
Out of 100 points

Files to submit: 1. HW9.py

This is a PAIR PROGRAMMING Assignment: Work with your partner!
For pair programming assignments, you and your partner should turn in identical assignments. List both partners names at the top. Your submission must not be substantially similar to another pairs' submission. Collaboration at a reasonable level will not result in substantially similar code. Students may only collaborate with fellow students currently taking CS 2316, the TA's and the lecturer. Collaboration means talking through problems, assisting with debugging, explaining a concept, etc. You should not exchange code or write code for others.

For Help:
- TA Helpdesk – Schedule posted on class website.
- Email TA's or use T-Square Forums

Notes:
- **Don't forget to include the required comments and collaboration statement** (as outlined on the course syllabus).
- **Do not wait until the last minute** to do this assignment in case you run into problems.
- **Read the entire specifications document** before starting this assignment.

Premise

You own a fruit stand that sells apples, pears, and grapes, and you receive a new shipment of fruit every day. You may not hold produce for more than one day- anything that you do not sell is picked up by your vendor the next morning to be taken to the local zoo to feed the monkeys. It seems recently that you have not been ordering an amount that is appropriate for the demand. You have been collecting daily demand data for the past year and it is stored in the ***demand*** table in your database. Using this data you will calculate a new daily ordering policy.

In addition, since you have a crazy course load next semester you will be hiring an assistant who will need to be able to calculate a new policy when you get new costs from your supplier. Therefore you will need to make a GUI interface for them to use that will allow for the problem parameters to be given as user inputs.

Your code should be in one class and contain the following methods:
- __init__()
- clicked()
- computePolicy()
- connectMySQL()

Newsvendor Problem

The Newsvendor Problem is the problem where you have to determine the order policy for one item with stochastic (uncertain) demand over one period- in our case one day. There are two costs that have to be taken into consideration, the overage cost (or the cost for each unit over the quantity demanded) and the shortage cost (the cost for each piece that is demanded but that you do not have inventory for).

For this problem assume that the demand is normally distributed.

The first step is to solve for the critical ratio R where c_u is the shortage (underage) cost and c_o is the overstock cost.

$$R = \frac{c_u}{c_u + c_o}$$

The second step is to find the z-value. Mathematically we have :

$$z = \Phi^{-1}(R)$$

However this is difficult to calculate so we will use a modified z-table. Information about the standard z-table can be found at http://en.wikipedia.org/wiki/Standard_normal_table. The data from this table has been placed into a database table as described below so that it can be accessed easily for this assignment. The format of the ***zValues*** table is different than the standard z-table but the data is the same.

The final step is to calculate the daily order quantity Q* where μ is the mean of the demand, σ is the standard deviation of the demand and z is the z-value that was looked up in the z-table.

$$Q^* = \mu + \sigma z$$

Database Table Structure Information

demand table
There are 4 fields:
Auto-Incrementing product number: **id** (integer)
Product name: **name** (text)
Date of recorded demand: **date** (date)
Number ordered: **demand** (integer)

demand

id	name	Date	demand
1	Apple	2010-11-01	15
2	Pear	2010-11-01	23
3	Grape	2010-11-02	8
4	Apple	2010-11-02	21
5	Grape	2010-11-02	23
6	Pear	2010-11-03	10

zValues table
There are 2 fields:
Z-Value: **z** (float)
Critical Ratio (R): **ratio** (float)

zValues

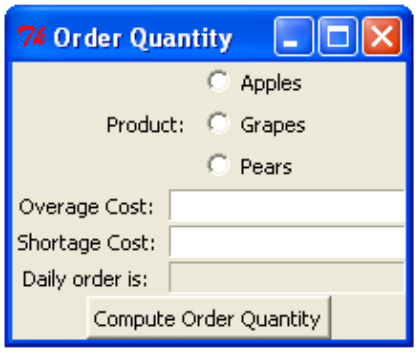
z	ratio
1.45	0.4265
3.20	0.4993

Function Name: __init__

This method is automatically called whenever you create a new instance of your object.
The __init__ method is responsible for arranging and initializing your GUI. You should create a GUI which has the following functionality but you may customize the design:

Window Title
    Descriptive title for your GUI
User input to select product- “Apples”, “Grapes”, “Pears”
    You may use an entry, radio buttons, drop down menu, etc
    You may choose to have a default value or not
User input for overage cost- any floating point or integer ≥ 0
    You may use an entry, slider, drop down menu, etc
    You may choose to have a default value or not
User input for underage cost- any floating point or integer ≥ 0
    You may use an entry, slider, drop down menu, etc
    You may choose to have a default value or not
Button that triggers the order quantity calculation
    Must have descriptive text (“compute policy”, “find order qty”, etc)
Display order policy
    You may use whatever display method you prefer (read only entry, label, etc)

Sample GUI:



Function Name: clicked

When the “Compute Daily Order Quantity” (with whatever name you have given it) button is clicked the clicked() method must be called. This method must check to see if all of the data input entry inputs have valid values. If there is data missing or it is of a type/format that cannot be used (ex: “two” instead of 2) your code should produce an error dialog box informing the user of the problem. After this nothing else should be done until the button is clicked again.

If all of the input data is present and valid the computePolicy method should be called. The value returned from this method must then be displayed in the GUI in whatever format you have chosen.

Function Name: connectMySQL

This method will connect to the database. The parameters are the hostname, username, and password. These are the server for the course database, the class username and the class password, respectively (which you can find on T-Square).

Function Name: computePolicy

This method will take in as parameters the user specified values for the product, the overage cost and the shortage cost.

From the data in the ***demand*** table you will need to find the average daily demand (μ) as well as the standard deviation(σ) . *Hint: there is a SQL function stddev_samp() for standard deviation.

Calculate the critical ratio (R) and round to 4 decimal places.

After the critical ratio has been calculated, find the corresponding z-value in the ***zValues*** table. Because your calculated R may not exist in the table you need to select the z value that corresponds to the ratio that is closest (smallest absolute difference) to your R value. Fortunately, your MySQL database administrator has given you some hints about this query:

1. You will need to use the absolute value function (ABS) when trying to find the differences between ratios (i.e. ABS(first-second)). You may use this function inside of WHERE clauses without a nested SELECT as it is not an aggregate function.
2. You will need to use a nested SELECT in this query. This nested query should find the minimum difference between the ratio you want and the ratios in the zValues table (Make sure you use ABS for the difference INSIDE the call to the minimum function, not outside!).
3. You want to only select a zValue if the difference between your ratio and the ratio in the table is equal to the minimum you found in the nested SELECT described above.

You *must* find the zValue you need by using only MySQL queries executed through Python (i.e. your query must return the correct zValue); you may not find the zValue by selecting all the zValues then finding the zValue in Python.

This method must return the daily order quantity as an integer. Any value that would have been fractional should be **rounded up** instead of truncated (You can't order 0.75 of a grape). *Hint: there is a function in the math class - math.ceil(input) – that may be helpful.

Example Calculations:

Given the demand table on page 2, the zValues table in the database, and the Co and Cu values below, the following policies would be obtained (You may use the sampleDemand table in the database for your convenience). Your actual submission, however, must use the actual demand table rather than the sampleDemand table – make sure your submission uses the correct table!

Co and Cu Values

Fruit	Cu	Co
Apples	3	0.50
Pears	4	0.75
Grapes	2	0.25

Results

	Average	STDEV	R	z	Q*
Apple	18	4.242641	0.8571	1.07	23
Pear	16.5	9.192388	0.8421	1.0	26
Grape	15.5	10.6066	0.8889	1.22	29

Grading:

You will earn points as follows for each piece of functionality that works correctly according to the specifications.

General		**10**
Comments are used to explain each function and query	10	
The GUI		**25**
GUI has all required components and is easy to use	15	
Correct binding	5	
Multiple trials can be conducted without restarting	5	
clicked()		**20**
Checks that all required inputs are present and valid	10	
Calls computePolicy() with the correct parameters	5	
Updates GUI with the computed order quantity	5	
computePolicy()		**40**
Policy computed for the selected product	5	
Correctly finds the average daily demand	10	
Valid query and usable result		
Correctly computes the R value	5	
Correctly finds the z value	10	
Valid query and usable result		
Uses MySQL query to find result		
Correctly computes the daily order quantity	5	
Returns the order quantity	5	
connectMySQL()		**5**
Correctly connects to the database	5	