**Python Programming Exercise (Basic OOP)**

**Naming Rule: All module, method and function names (except the __init__, __str_ and main methods) MUST include your initials at the end of the name.**

For example:

- If your class module is called ProgramExercise and your name is John Smith, your class file should be called ProgramExerciseJS
    - Try not to call your class file ProgramExercise – pick a name that better suits thecontent of your class module.
- If your driver module is called ExerciseDriver and your name is John Smith, your class file should be called ExerciseDriverJS
    - Try not to call your class file ExerciseDriver – pick a name that better suits thecontent of your class module.
- If a method is to be called setPrice, and your name is John Smith, your method should be called setPriceJS.
- The __init__ and __str_ method in your class module and the main function in your driver should be the only exceptions to this rule.
- Please include #comments and descriptive """documentation""" in your code.
- Not following the rules can result in deductions.
- Upload the files in your GitHub repository and share the link on or before 12 midnight on Wednesday, December 8 in the forum thread.

---

**Fancy Shopping list**

A client has asked you to make some unique purchases on their behalf. They are interested in purchasing food from a unique grocery. The grocery specializes in rare (and expensive) food and has a very limited inventory (due in part to rarity of items as well as their commitmentto ensuring that all items are ethically sourced). You check out the inventory and find the following items in stock.

| Food Item | Price per Pound (U.S.) |
|---|---|
| Dry Cured Iberian Ham | $177.80 |
| Wagyu Steaks | $450.00 |
| Matsutake Mushrooms | $272.00 |
| Kopi Luwak Coffee | $306.50 |
| Moose Cheese | $487.20 |
| White Truffles | $3600.00 |
| Blue Fin Tuna | $3603.00 |
| Le Bonnotte Potatoes | $270.81 |

You see this as an opportunity to practice your programming so that you can spend the client's money carefully once they decide which items and amount (in pounds) that they want.

You will write two programs for this endeavor. The first will be a class definition module, and the second will be the driver program. (Remember the naming rules for these modules!)

The class module should contain the following:

1. An initializer method that accepts two parameters (other than self):
    a. A parameter corresponding to the name of the food
    b. A parameter corresponding to the amount of the food (in pounds) to order

    However, the initializer method should contain **four** hidden attributes:
    a. The name of the food - to be updated using the food parameter
    b. The amount in pounds - to be updated using the amount parameter
    c. The standard price of the food item per pound - to be updated using a private method
    d. The calculated price of the ordered item (based on amount ordered) – to be updated using a public method

2. A private method that will store the list of items and their price per pound (in other words, the information provided in the table above). The method accepts no parameters (other than self) and returns no value. Use an if-elif structure to set the standard prices of the food. Reference the hidden attributes of food name and standard price to set the prices.

    Use the header of the method is (provided below) as well as the pseudocode to complete this method:

    ```
    #use this header (note the   before the name)
    def __PriceList(self):

          if foodname is 'Dry Cured Iberian Ham' #write in actual python
                then standardprice is 177.80
          #complete the method…
    ```

    Make sure to include a trailing else that sets the price to 0.00 if an item that is not on the table is referenced (or if an item is misspelled)

3. A public method to calculate the cost of the ordered food. The cost should be calculated using the formula:

    Amount of food (in pounds) x price per pound

    The method accepts no parameters (other than self), but returns the calculated cost.

4. Accessors as needed (or an __str_ method if you prefer).

The driver program will import the class module you created. This module should contain the following components:

1. A function that creates a list of objects. The function accepts no parameters but returns the list of objects. The function should:
    a. Create an empty list.
    b. Prompt the user for the number of items. This value will be used to determine the number of repetitions for a necessary loop. (Make sure to include input validation to ensure that the number of items purchases is at least 1.)
    c. Contain a loop that prompts the user for the name of the item and the amount of the item in pounds. (Make sure to include input validation to ensure that the number of pounds is greater than 0.) The loop should use this information to create an object, and append the object to the list
    d. Returns the list (once the loop is completed)
2. A function to display the contents of the list. This function accepts a list of objects as a parameter but does not return a value. The function should:
    a. display the contents of each object in the list (all 4 attributes). Make sure to include appropriate formatting for prices.
3. A function that calculates the total cost of all items. Recall, your object will only have the cost of each item (based on the amount of pounds ordered for that item). This function accepts the list of objects as a parameter and returns a value. This function should:
    a. access the individual cost of each ordered item
    b. calculate the total cost of all the items in the list
    c. return the total cost
4. A main function. Your main function should:
    a. Call the three aforementioned functions

Sample output (includes input validation)

How many items will you order today? -1
Number of items must be at least 1.
How many items will you order today? 4
Item #1-
Enter food: Wagyu Steaks
Enter amount of pounds: 3

Item #2-
Enter food: Matsutake Mushrooms
Enter amount of pounds: 0.5

Item #3-
Enter food: Le Bonnotte Potatoes
Enter amount of pounds: 4

Item #4-
Enter food: Moose Cheese
Enter amount of pounds: 0
Amount of pounds must be greater than 0.
Enter amount of pounds: 1.5

Here's a summary of the items purchased:
---
Item: Wagyu Steaks
Amount ordered: 3.0 pounds
Price per pound: $450.00
Price of order: $1350.00

Item: Matsutake Mushrooms
Amount ordered: 0.5 pounds
Price per pound: $272.00
Price of order: $136.00

Item: Le Bonnotte Potatoes
Amount ordered: 4.0 pounds
Price per pound: $270.81
Price of order: $1083.24

Item: Moose  Cheese
Amount ordered: 1.5 pounds
Price per pound: $487.20
Price of order: $730.80

Total cost: $3300.04

Sample output (includes item not on list)

```
How many items will you order today? 2
Item #1-
Enter food: Blue Fin Tuna
Enter amount of pounds: 2.5

Item #2-
Enter food: Bacon, lots of Bacon
Enter amount of pounds: 5

Here's a summary of the items purchased:
----------------------------
Item: Blue Fin Tuna
Amount ordered: 2.5 pounds
Price per pound: $3603.00
Price of order: $9007.50

Item: Bacon, lots of Bacon
Amount ordered: 5.0 pounds
Price per pound: $0.00
Price of order: $0.00

Total cost: $9007.50
```