Module 12 Assignment
Points: 100 points

1. Design a **Box** class stored in module **box.py** with data attributes for integer length, width, and height, **__init__** method with default values of 1 for each data attribute, accessor and mutators methods for each data attribute, string representation methods **__str__** and **__repr__**, **__eq__** method where boxes are considered the same with equal dimensions, **__repr__** method which is the same as **__eq__**, and methods for surface **surface_area** (sum of areas of the six rectangular sides) and **volume** (length times height times width). Include comments on methods. This class can assume valid data and does not need to use data validation on data type or range.

- Note you are writing a general purpose **Box** class that can be used in any application that needs a box. Not all methods might be called in this particular application.
- Test your **Box** class yourself.

2. Write a main application that uses two **Box** objects to model a snow globe box and a shipping box. Packing material costs $0.0023 per cubic inch. Packing material is only needed on a partially filled shipping box. Plastic wrap around a sealed shipping box costs $0.0008 per square inch. These values should be named constants.

<u>Application:</u>
A company sells snow globes in cubic boxes with choices of 4x4x3 ($5.99) , 6x6x5 ($9.99) or 10x6x8 ($13.99) dimensions in inches.

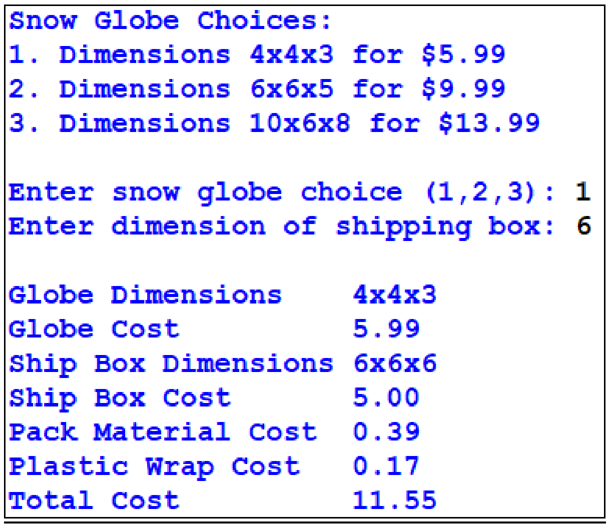
<u>Input:</u>
Display a menu of snow globe choices and run a data validation loop asking the user for the choice of snow globe until a correct choice is entered. You only need to do range checking and can assume an integer is entered. Create a snow globe **Box** object. Ask the user for dimension of the shipping box. Shipping boxes are cubic so a dimension of 4 would indicate a box that is 4x4x4 inches. Use a loop to ensure a valid shipping box dimension is entered that will hold the snow globe. Create a shipping **Box** object. A shipping box costs $5.00.

<u>Processing:</u>
1. Compute the cost of packing material to fill the empty space in the shipping box around the snow globe using the **Box volume** method.

2. Compute the cost of plastic wrap around the ship box using the **Box surface_area** method.

3. Report dimensions of snow globe and shipping box using **Box accessor** methods and costs for ship box, packing material, and plastic wrap and total price in a neat chart using f strings for column width and money amounts with two decimals. See sample output for report design.

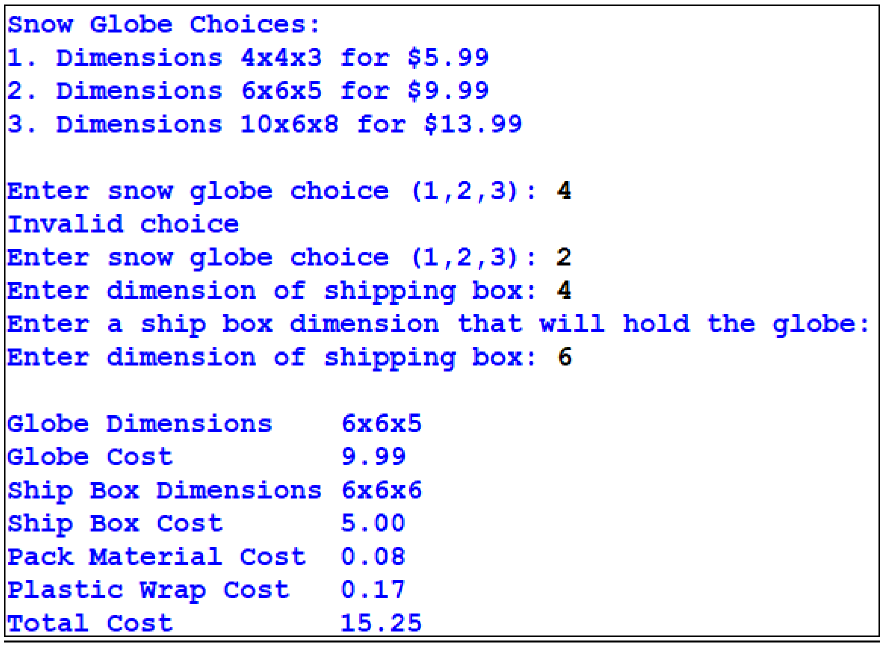
<u>Sample Output</u>: (Be sure to test code on several cases)



```
Snow Globe Choices:
1. Dimensions 4x4x3 for $5.99
2. Dimensions 6x6x5 for $9.99
3. Dimensions 10x6x8 for $13.99

Enter snow globe choice (1,2,3): 1
Enter dimension of shipping box: 6

Globe Dimensions     4x4x3
Globe Cost           5.99
Ship Box Dimensions  6x6x6
Ship Box Cost        5.00
Pack Material Cost   0.39
Plastic Wrap Cost    0.17
Total Cost           11.55
```



```
Snow Globe Choices:
1. Dimensions 4x4x3 for $5.99
2. Dimensions 6x6x5 for $9.99
3. Dimensions 10x6x8 for $13.99

Enter snow globe choice (1,2,3): 4
Invalid choice
Enter snow globe choice (1,2,3): 2
Enter dimension of shipping box: 4
Enter a ship box dimension that will hold the globe:
Enter dimension of shipping box: 6

Globe Dimensions     6x6x5
Globe Cost           9.99
Ship Box Dimensions  6x6x6
Ship Box Cost        5.00
Pack Material Cost   0.08
Plastic Wrap Cost    0.17
Total Cost           15.25
```

<u>Required Code:</u>

1. The **Box** class is stored in its own module **box.py**.
2. The **main** method is used with no global variables or code outside of functions.
3. Main code must make use of a **Box** object for the snow globe box. Once the **Box** object is created, <u>only</u> **Box** methods may be used.
4. Main code must make use of a **Box** object for the shipping box. Once the **Box** object is created, only **Box** methods may be used.
5. Data validation loops are used to check ranges on input data. Exception handling is not required to check data types. No **while True:** loops with **break** are accepted.

<u>Submission:</u> Upload **box.py** and the main application source code .py file.

Grading Rubric:

Documentation and naming of identifiers follows standards.	10 pts
The **Box __init__** method is correct.	5 pts
The **Box** accessor methods are correct.	5 pts
The **Box** mutator methods are correct.	5 pts
The **Box** string representation methods are correct.	5 pts
The **Box __eq__** method is correct.	5 pts
The **Box surface_area** method is correct.	5 pts
The **Box volume** method is correct.	5 pts
Named constants are created and commented for packing and wap costs.	5 pts
A data validation loop ensures the user enters a valid snow globe choice.	5 pts
The shipping box input data validation loop is correctly structured.	5 pts
The snow globe and shipping box objects are constructed correctly.	10 pts
The computation of wrapping cost is correct and uses **Box** methods.	10 pts
The computation of filling cost is correct and uses **Box** methods.	10 pts
The computation of total cost is correct.	5 pts
Output is complete and formatted correctly.	5 pts

Need Help?

1. Email your question with your attached source code with extension .txt or a zip file. The college email is not accepting .py extension on attachments.

2. Use the scheduling software to schedule a Zoom meeting.