Assignment : Online Store
Course: CSE111
Marks : 20
Submission Deadline: 05 - 04- 23 (11:59 PM)

You have been tasked with creating an online store system using object-oriented programming in Python. The system should consist of classes for products, online store items, customers, orders, and the online store itself.

Requirements:

1. Create a `Product` class with attributes for `name`, `quantity` and `price`. Implement two methods called `calculate_price` that returns the calculated price by considering quantity and price variables and `__str__` that returns a string representation of the product.
`[calculated_price = price * quantity]`

2. Create a subclass called `OnlineStoreItem` that inherits from `Product`. The `OnlineStoreItem` class should add an attribute for `shipping_weight` and create the `calculate_shipping_cost` method to return the cost of shipping based on the `shipping_weight` attribute. `[shipping_cost = 2.5 * shipping_weight].`

3. Create a `Customer` class with attributes for `name`, `email`, `address`, and `phone_number`. Implement a method called `__str__` that returns a string representation of the customer.

4. Create an `Order` class with attributes for `customer`, `products`, and `quantities`. Implement methods to calculate the order total and remove a product from the order. Implement a method called `__str__` that returns a string representation of the order.

5. Create a `OnlineStore` class that has attributes for `name` and `products`. Implement methods to add and remove products, register customers, and place an order. Implement a method called `__str__` that returns a string representation of the store, including all of its products.

Tasks:

1. Create the `Product` class and test it by creating a few instances of `Product` and printing them out using the `__str__` method.

2. Create the `OnlineStoreItem` class and test it by creating a few instances of `OnlineStoreItem` and printing them out using the `__str__` method. Verify that the `calculate_shipping_cost` method works as expected.

3. Create the `Customer` class and test it by creating a few instances of `Customer` and printing them out using the `__str__` method.

4. Create the `Order` class and test it by creating a few instances of `Order` and printing them out using the `__str__` method. Verify that the `calculate_total` and `remove_product` methods work as expected.
5. Create the `OnlineStore` class and test it by creating a few instances of `OnlineStore` and printing them out using the `__str__` method. Verify that the `add_product`, `remove_product`, `register_customer`, and `place_order` methods work as expected.

```
# Driver code
# Create some products
product1 = OnlineStoreItem("Apple", 0.5, 0, 0.2)
product2 = Product("Banana", 0.3, 0)
product3 = OnlineStoreItem("Orange", 0.4, 0,
0.3)

# Create an online store
store = OnlineStore()

# Add the products to the store
store.add_product(product1, 100)
store.add_product(product2, 50)
store.add_product(product3, 75)

# View all products in the store
print("1====================")
print(store)
print("2====================")
# Create a new customer
customer1 = Customer("Alice",
"alice@example.com", "123 Main St.","555-1234")

# Register the customer with the store
store.register_customer(customer1)

store.customers_info()
print("3====================")
# Place an order for the customer
order1 = store.place_order(customer1, [product1,
```

Output:

```
1====================
Products in stock:
Apple ($0.50) x 100
Banana ($0.30) x 50
Orange ($0.40) x 75
2====================
Total Customer :  1
Name : Alice
Email : alice@example.com
Address : 123 Main St.
Phone : 555-1234
3====================
Order for Alice:
Name x Stock x Ordered
Apple ($0.50) x 50 x 50
Banana ($0.30) x 25 x 25
Total: $33.00
4====================
Order for Alice:
Name x Stock x Ordered
Banana ($0.30) x 25 x 25
Total: $7.50
4====================
```

```
product2], [50, 25])

# View the order
print(order1)
print("4====================")
# Remove a product from the order
order1.remove_product(product1)

# View the updated order
print(order1)
print("4====================")
```

Submission:

Submit a Python file that contains all of the classes and methods required for this assignment, as well as code that creates instances of each class and tests their functionality. Include comments in your code to explain what each class and method does.

Submission Form : https://forms.gle/MKeqfgeJPfwpHrEp7