# Module 4 Milestone

# **Online Shopping Cart**

**Step 1:**

Build the ItemToPurchase class with the following specifications:

* Attributes
* item\_name (string)
* item\_price (float)
* item\_quantity (int)
* Default constructor
* Initializes item's name = "none", item's price = 0, item's quantity = 0
* Method
* print\_item\_cost()

**Solution:**

1. Defined a class ItemToPurchase with given attributes. I have assigned appropriate data types to the data attributes as per the requirements(Programiz, 2020). Below code block defines class with attributes item\_name as string, item\_price as float and item\_quantity as integer.

**class ItemToPurchase:**

**item\_name : str**

**item\_price : float**

**item\_quantity : int**

1. Defined/created constructor accepting 3 parameters with given initialized values (geeksforgeeks, 2023). Constructors assist us to assign values to the data attributes at the time of creating the object itself. While creating an object, if we don’t pass any values, the constructor will use the default value to assign initial values. In this case, it is defined as this - name = 'none', price = 0, quantity = 0. Constructors can be used to enforce encapsulation, by ensuring that the object’s attributes are initialized correctly and in a controlled manner.

def \_\_init\_\_(self, name = 'none', price = 0, quantity = 0):

self.item\_name = name

self.item\_price = price

self.item\_quantity = quantity

1. Defined/created Method print\_item\_cost(Programiz, 2020).This will calculate cost of an item for given price and quantity. It prints this calculation with appropriate format. Here, I have used String Formatter to print $ sign and currency format to display all the price value '${:.2f}'.format(value)(ZyBook, 2019).

def print\_item\_cost(self):

print(self.item\_name, self.item\_quantity, '@', '${:.2f}'.format(self.item\_price), '=', '${:.2f}'.format(self.item\_price \* self.item\_quantity))

**Step 2:**

In the main section of your code, prompt the user for two items and create two objects of the ItemToPurchase class.

**Solution:**

1. Prompt the user the details for two items, create two objects of the class item\_to\_purchase.
2. Here, I have tried to incorporate two different ways for creating objects and assigning values.
   1. Option 1: Object creation of class ItemToPurchase using default constructor with default initialized value. Get user input values and assign them to the data members via object assignment
   2. Option 2: Get user Input and save those values into local variables. Use the parameterized constructor to pass those values during object creation.

# Technique 1: Object creation of class ItemToPurchase using default constructor with default initialized value

item1 = ItemToPurchase()

print('Item 1')

# assign user input value via property assignment

item1.item\_name = input('Enter Item Name: \n')

item1.item\_price = float(input('Enter Item Price: \n'))

item1.item\_quantity = int(input('Enter Item Quantity: \n'))

# Technique 2: Take User Input and store them in local variables.

print('\nItem 2')

in\_item\_name = input('Enter Item Name: \n')

in\_item\_price = float(input('Enter Item Price: \n'))

in\_item\_quantity = int(input('Enter Item Quantity: \n'))

# Object Creation of class ItemToPurchase using parameterized constructor with passing attribute values during initialization

item2 = ItemToPurchase(in\_item\_name, in\_item\_price, in\_item\_quantity)

**Step 3:**

Add the costs of the two items together and output the total cost.

**Solution:**

1. Invoke print\_item\_cost for both the objects to print the individual item cost.
2. Calculate Total Price. Format the output value with currency format using string formatter (Zybook, 2019).
3. Added some code for displaying values in user friendly view/format.

**# Invoke print\_item\_cost() for each object and print item cost.**

**# Calculate Total Cost and format the output values appropriately.**

**print('\n-------TOTAL COST---------\n')**

**item1.print\_item\_cost()**

**item2.print\_item\_cost()**

**print('\n\tTotal:', '${:.2f}'.format((item1.item\_price \* item1.item\_quantity) + (item2.item\_price \* item2.item\_quantity)))**

**print('==============================')**

**print('Thank you for shopping with us!\n')**

**Full Code:**

**# Step 1: Define a class ItemToPurchase**

**class ItemToPurchase:**

**item\_name : str**

**item\_price : float**

**item\_quantity : int**

**# Constructor with 3 parameters with initialized values**

**def \_\_init\_\_(self, name = 'none', price = 0, quantity = 0):**

**self.item\_name = name**

**self.item\_price = price**

**self.item\_quantity = quantity**

**# Method for calculating cost of one item for given quantity at user provided value and Print them.**

**def print\_item\_cost(self):**

**print(self.item\_name, self.item\_quantity, '@', '${:.2f}'.format(self.item\_price), '=', '${:.2f}'.format(self.item\_price \* self.item\_quantity))**

**print('\n⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀')**

**print('⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀')**

**print('⠀⠈⠛⠻⠶⣶⡄⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀')**

**print('⠀⠀⠀⠀⠀⠈⢻⣆⣀⣀⣀⣀⣀⣀⣀⣀⣀⣀⣀⣀⣀⣀⣀⣀⣀⣀⣀⠀⠀⠀')**

**print('⠀⠀⠀⠀⠀⠀⠀⢻⡏⠉⠉⠉⠉⢹⡏⠉⠉⠉⠉⣿⠉⠉⠉⠉⠉⣹⠇⠀⠀⠀')**

**print('⠀⠀⠀⠀⠀⠀⠀⠈⣿⣀⣀⣀⣀⣸⣧⣀⣀⣀⣀⣿⣄⣀⣀⣀⣠⡿⠀⠀⠀⠀')**

**print('⠀⠀⠀⠀⠀⠀⠀⠀⠸⣧⠀⠀⠀⢸⡇⠀⠀⠀⠀⣿⠁⠀⠀⠀⣿⠃⠀⠀⠀⠀')**

**print('⠀⠀⠀⠀⠀⠀⠀⠀⠀⢹⣧⣤⣤⣼⣧⣤⣤⣤⣤⣿⣤⣤⣤⣼⡏⠀⠀⠀⠀⠀')**

**print('⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⢿⠀⠀⢸⡇⠀⠀⠀⠀⣿⠀⠀⢠⡿⠀⠀⠀⠀⠀⠀')**

**print('⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⢸⣷⠤⠼⠷⠤⠤⠤⠤⠿⠦⠤⠾⠃⠀⠀⠀⠀⠀⠀')**

**print('⠀⠀⠀⠀⠀⠀⠀⠀⠀⢀⣾⠁⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀')**

**print('⠀⠀⠀⠀⠀⠀⠀⠀⠀⢾⣷⢶⣶⠶⠶⠶⠶⠶⠶⣶⠶⣶⡶⠀⠀⠀⠀⠀⠀⠀')**

**print('⠀⠀⠀⠀⠀⠀⠀⠀⠀⠸⣧⣠⡿⠀⠀⠀⠀⠀⠀⢷⣄⣼⠇⠀⠀⠀⠀⠀⠀⠀')**

**print('⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀')**

**print('⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀⠀')**

**print('\n\tWelcome to the Online Shopping Cart')**

**# Step 2: prompt the user for two items and create two objects of the ItemToPurchase class.**

**# Technique 1: Object creation of class ItemToPurchase using default constructor with default initialized value**

**item1 = ItemToPurchase()**

**print('Item 1')**

**# assign user input value via property assignment**

**item1.item\_name = input('Enter Item Name: \n')**

**item1.item\_price = float(input('Enter Item Price: \n'))**

**item1.item\_quantity = int(input('Enter Item Quantity: \n'))**

**# Technique 2: Take User Input and store them in local variable.**

**print('\nItem 2')**

**in\_item\_name = input('Enter Item Name: \n')**

**in\_item\_price = float(input('Enter Item Price: \n'))**

**in\_item\_quantity = int(input('Enter Item Quantity: \n'))**

**# Object Creation of class ItemToPurchase using parameterized constructor with passing attribute values during initialization**

**item2 = ItemToPurchase(in\_item\_name, in\_item\_price, in\_item\_quantity)**

**# Step 3: Add the costs of the two items together and output the total cost**

**# Invoke print\_item\_cost() for each object and print item cost.**

**# Calculate Total Cost and format the output values appropriately.**

**print('\n-------TOTAL COST---------\n')**

**item1.print\_item\_cost()**

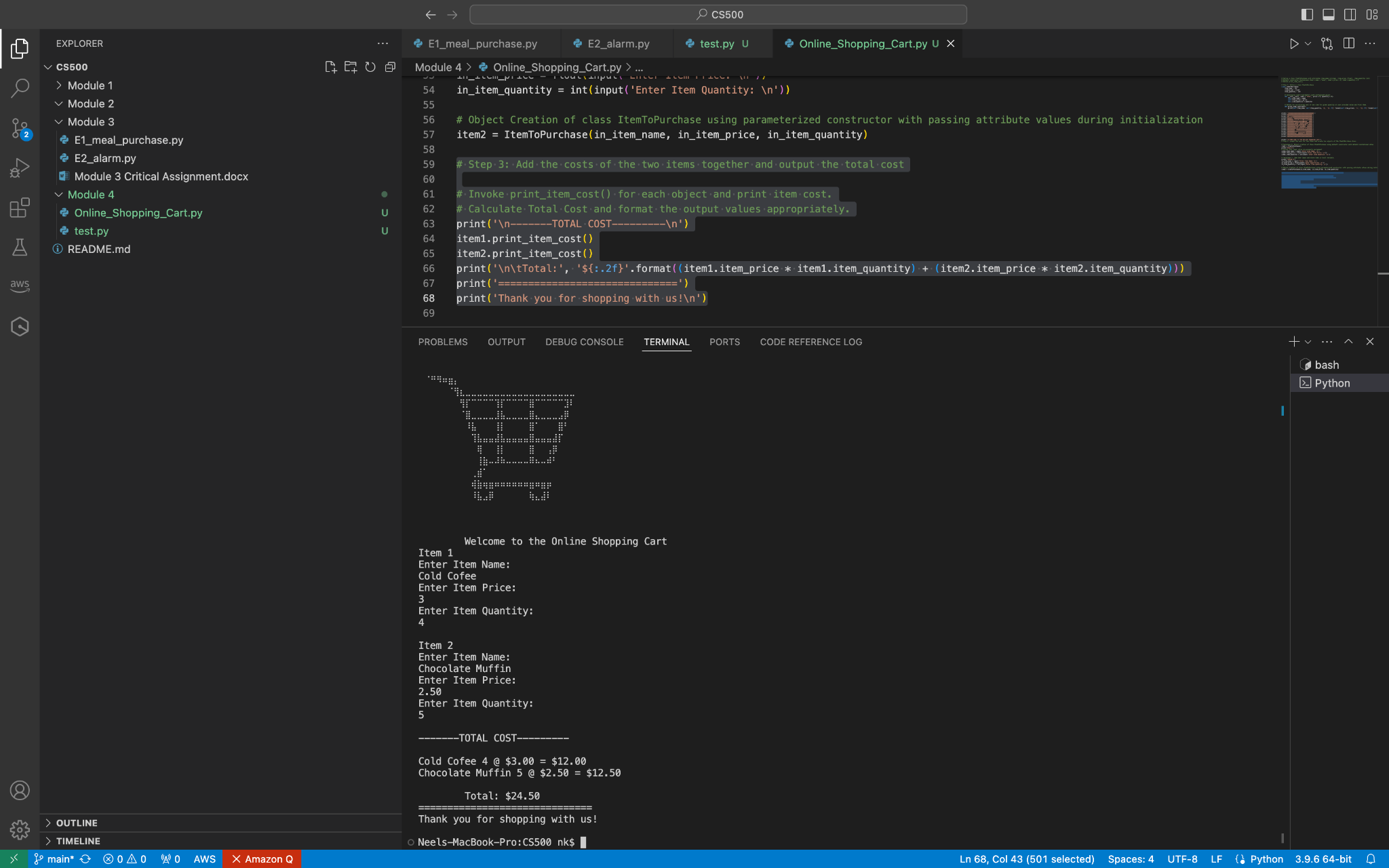
**item2.print\_item\_cost()**

**print('\n\tTotal:', '${:.2f}'.format((item1.item\_price \* item1.item\_quantity) + (item2.item\_price \* item2.item\_quantity)))**

**print('==============================')**

**print('Thank you for shopping with us!\n')**

**Output:**

**Source Code:**

[**https://github.com/nshahcsu/MSAI/blob/main/CS500/Module%204/Online\_Shopping\_Cart.py**](https://github.com/nshahcsu/MSAI/blob/main/CS500/Module%204/Online_Shopping_Cart.py)

**References:**

GeeksforGeeks. (2023, March 1). *Constructors in Python*. GeeksforGeeks. https://www.geeksforgeeks.org/constructors-in-python/

Programiz. (2020, December 3). *Object-oriented Programming (OOP) in Python (Easy to* understand *guide) #20* [Video]. YouTube. <https://www.youtube.com/watch?v=pnWINBJ3-yA>

zyBooks. (2019 August). CSC500: Principles of Programming. Module 2.19

## Update after Zoom Video:

**Showcasing usage of List, While and For loop:**

I have added another File in GIT HUB - Online\_Shopping\_Cart\_using\_loop after watching your zoom video. I have tried to showcase a usage of While and For loop to prompt user for adding multiple items into the cart (up to max count 10)

**Code block 1:**

1. This program allows user to add up to 10 items.
2. Added functionality to ask user if they want to add items to cart and add items to the list object to make it dynamic (GeeksforGeeks, 2024).
3. If user types in Y/y (upper/lower case), the while loop continues to allow to add items up to max items count (Sturtz, 2020).
4. If user types N/n (upper/lower case), the while loop breaks and it runs the remaining main program.
5. If user types wrong value, allow user to enter one more time before exiting the loop.

MAX\_ITEM = 10

item\_count = 0

items = []

while (item\_count< MAX\_ITEM):

temp\_item = ItemToPurchase()

item\_count += 1

print('\nItem', item\_count)

# assign user input value via attribute assignment

temp\_item.item\_name = input('Enter Item Name: \n')

temp\_item.item\_price = float(input('Enter Item Price: \n'))

temp\_item.item\_quantity = int(input('Enter Item Quantity: \n'))

items.append(temp\_item)

if(item\_count< MAX\_ITEM):

user\_input = input('Do you want to add more item to the cart? (Y/N)\n')

if(user\_input == 'Y' or user\_input == 'y'):

continue

elif (user\_input == 'N' or user\_input == 'n'):

break

else:

user\_input = input('Invalid Entry. Please select Type Y/N.\n')

if(user\_input == 'Y' or user\_input == 'y'):

continue

else:

break

**Code block 2:**

1. Create a variable Total\_cost to hold the value for cost of entire cart and initialize with 0.
2. Iterate through the list using ‘FOR Loop’ (Sturtz, 2020).
3. Invoke print\_item\_cost for each list member.
4. Calculate total cost.
5. Print the value in currency format.

Total\_Cost = 0

# Invoke print\_item\_cost() for each object and print item cost.

# Calculate Total Cost and format the output values appropriately.

print('\n-------TOTAL COST---------\n')

for item in items:

item.print\_item\_cost()

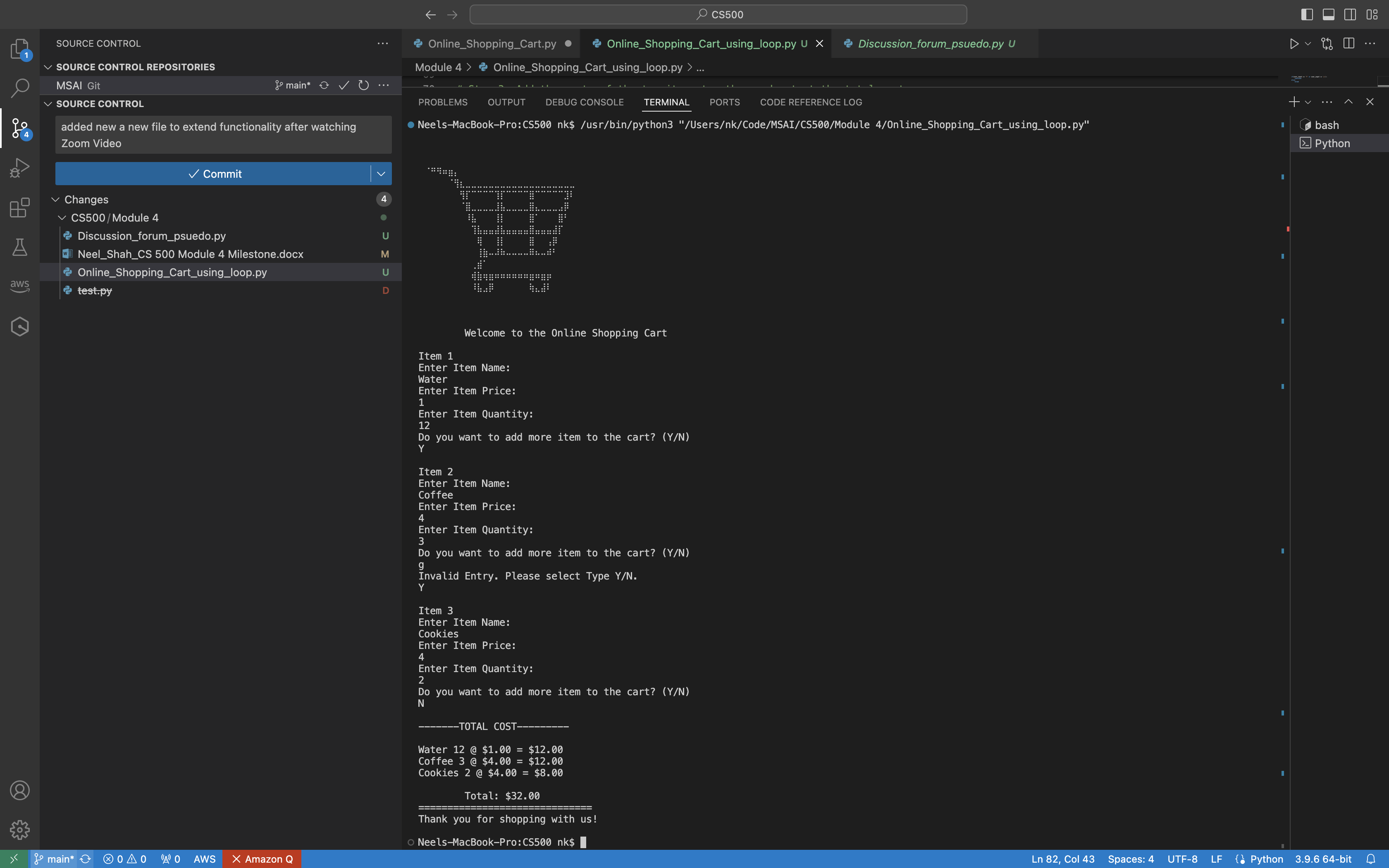
Total\_Cost += (item.item\_price \* item.item\_quantity)

print('\n\tTotal:', '${:.2f}'.format(Total\_Cost))

print('==============================')

print('Thank you for shopping with us!\n')

**Output:**

****

**GitHub link:**

**Additional References:**

GeeksforGeeks. (2024, June 5). Python lists. GeeksforGeeks. https://www.geeksforgeeks.org/python-lists/

Sturtz, J. (2020). Python “while” loops (indefinite iteration). https://realpython.com/python-while-loop/

Sturtz, J. (2020). Python “for” loops (definite iteration). Real Python. https://realpython.com/python-for-loop/