**Course**: CSC500\_1

**Module & Assignment**: 6 & Portfolio Milestone

**Student**: Jegan Palaniyandi

**Understanding of the problem statement**

This is a continuation of Portfolio Milestone 4 where we have created a shopping cart to display all the items with their price and the total cost.

In this milestone, we continued to develop a **shopping cart class** with an init method declared with parameters. We need to write methods to **create, delete and update** the **shopping cart**. We also need to include methods that **display item & description**, a method to return the **number of items in the shopping cart**.

The **main method** should **ask the user details**, then **display the shopping cart options** to perform the shopping and perform shopping based on the option selected until ‘**q**’ is entered.

**Code**

**class ItemToPurchase:**

***"""***

***ItemToPurchase class has a default constructor with three parameters***

***initialized with three parameters with default values as***

***item\_name = none***

***item\_price = 0 &***

***item\_quantity= 0.***

***The class has two methods,***

***1. add\_print\_list - returns the item name, count and price formatted.***

***This is used by list in the main method that prints***

***all the items to the receipt.***

***2. calculate\_item\_price - returns item\_price by multiplying price and***

***count of items.***

***"""***

**def \_\_init\_\_(self, item\_name='none', item\_price=0.00, item\_quantity=0):**

**self.item\_name = item\_name**

**self.item\_price = item\_price**

**self.item\_quantity = item\_quantity**

**def add\_print\_list(self):**

**return '{:<32} --> {:>6.2f}$'.format(self.item\_name + ' x ' + str(self.item\_quantity),**

**self.item\_price \* self.item\_quantity)**

**def calculate\_item\_price(self):**

**return self.item\_price \* self.item\_quantity**

**class ShoppingCart:**

***"""***

***Class: ShoppingCart***

***Constructor:***

***non-default params:***

***cart\_items -> List***

***Default params:***

***customer\_name='none'***

***current\_date='January 1, 2020'***

***"""***

**def \_\_init\_\_(self, cart\_items, customer\_name='none', current\_date='January 1, 2020'):**

**self.customer\_name = customer\_name**

**self.current\_date = current\_date**

**self.cart\_items = cart\_items**

**self.available\_items = {'Nike Romaleos': 'Volt color, Weightlifting shoes',**

**'Chocolate Chips': 'Semi - sweet',**

**'Powerbeats 2 Headphones': 'Bluetooth headphones'**

**}**

**self.total\_cost = 0**

**def add\_item(self):**

***"""***

***'add\_item' handles adding a new item to shopping cart***

***It prompts the user to enter the item name, price and quantity***

***and saves it to a list which saves itemToPurchase objects.***

***"""***

**print(f'\nEnter item details:')**

**print('-' \* 19)**

**name = input('Enter item name --> ')**

**price = float(input('Enter item price($) --> '))**

**quantity = int(input('Enter item quantity --> '))**

**item = ItemToPurchase(name, price, quantity)**

**self.cart\_items.append(item)**

**print('{} {}(s) priced at ${:.2f} added successfully to cart.'.format(quantity, name, price))**

**def remove\_item(self):**

***"""***

***print('Item not found in cart. Nothing removed.')***

***"""***

**def modify\_item(self):**

***"""***

***If item can be found (by name) in cart,***

***check if parameter has default values for description, price,***

***and quantity. If not, modify the item in the cart.***

***If item cannot be found (by name) in cart,***

***output this message: Item not found in cart. Nothing modified.***

***"""***

**def get\_num\_items\_in\_cart(self):**

***"""***

***:return:# of items in the shopping cart***

***"""***

**num\_items = 0**

**for item in self.cart\_items:**

**num\_items += item.item\_quantity**

**return num\_items**

**def get\_cost\_of\_cart(self):**

***"""***

***Outputs total of objects in cart.***

***If cart is empty, output this message: SHOPPING CART IS EMPTY***

***:return: none***

***"""***

**for item in self.cart\_items:**

**self.total\_cost += item.calculate\_item\_price()**

**# Following section prints the receipt by looping the**

**# 'itemList' and print the total cost at the bottom using**

**# 'totalCost' variable**

**print('-' \* 44)**

**print('Total {:>36.2f}$'.format(self.total\_cost))**

**print('-' \* 44)**

**def print\_total(self):**

***"""***

***prints the total cost of items in the cart. Has no parameters.***

***:return:none***

***"""***

**print()**

**print('\*' \* 44)**

**print(' ' \* 12 + '{}\'s Shopping Cart'.format(self.customer\_name))**

**print(' ' \* 16 + '{}'.format(self.current\_date))**

**print('\*' \* 44)**

**print('{:^44}\n'.format('Total # of items : ' + str(self.get\_num\_items\_in\_cart())))**

**for item in self.cart\_items:**

**print(item.add\_print\_list())**

**def print\_descriptions(self):**

***"""***

***prints available items and descriptions in the mart***

***:return:none***

***"""***

**print('\n{:>55}'.format('Available Items and Descriptions'))**

**print('-' \* 88)**

**print('|{:>32} |{:>34} |'.format('ITEM', 'DESCRIPTION'))**

**print('-' \* 88)**

**for l\_item, desc in self.available\_items.items():**

**print('|{:>32} |{:>34} |'.format(l\_item, desc))**

**print('-' \* 88)**

**def output\_cart(self):**

***"""***

***prints the list of items with their total price***

***and at the bottom prints the***

***:return:***

***"""***

**if self.get\_num\_items\_in\_cart():**

**self.print\_total()**

**self.get\_cost\_of\_cart()**

**else:**

**print('SHOPPING CART IS EMPTY')**

**def print\_menu\_options():**

***"""***

***Prints Menu with options and***

***prompt the user to enter an option***

***:return: shopping cart options menu***

***"""***

**menu\_options = ['a - Add item to cart', 'r - Remove item from cart', 'c - Change item quantity',**

**'i - Output items\'s descriptions', 'o - Output shopping cart', 'q - Quit']**

**print()**

**for l\_option in menu\_options:**

**print(f'{l\_option:^80}')**

**selected\_option = input('{:>49}'.format('Choose an option: '))**

**return selected\_option**

**if \_\_name\_\_ == '\_\_main\_\_':**

**# Print CSU groceries banner**

**print('\*' \* 100)**

**print(' ' \* 25, 'CSU Groceries - User-friendly Online Shopping')**

**print('\*' \* 100)**

**# Get and print user and date**

**user\_name = input('\nEnter your name : ')**

**today\_date = input('Enter today\'s date as <Month> <date>, <year> : ')**

**print('\n{:>50} \'{}\''.format('Welcome valued customer', user\_name))**

**print('{:>42} {}'.format('Shopping date :', today\_date))**

**# Create and Initialize an empty shopping cart with entered username and date**

**user\_cart\_items = []**

**user\_cart = ShoppingCart(user\_cart\_items, user\_name, today\_date)**

**# Print Menu for options until user enters 'q'**

**option\_selected = ''**

**while option\_selected != 'q':**

**if option\_selected == 'a':**

**user\_cart.add\_item()**

**elif option\_selected == 'i':**

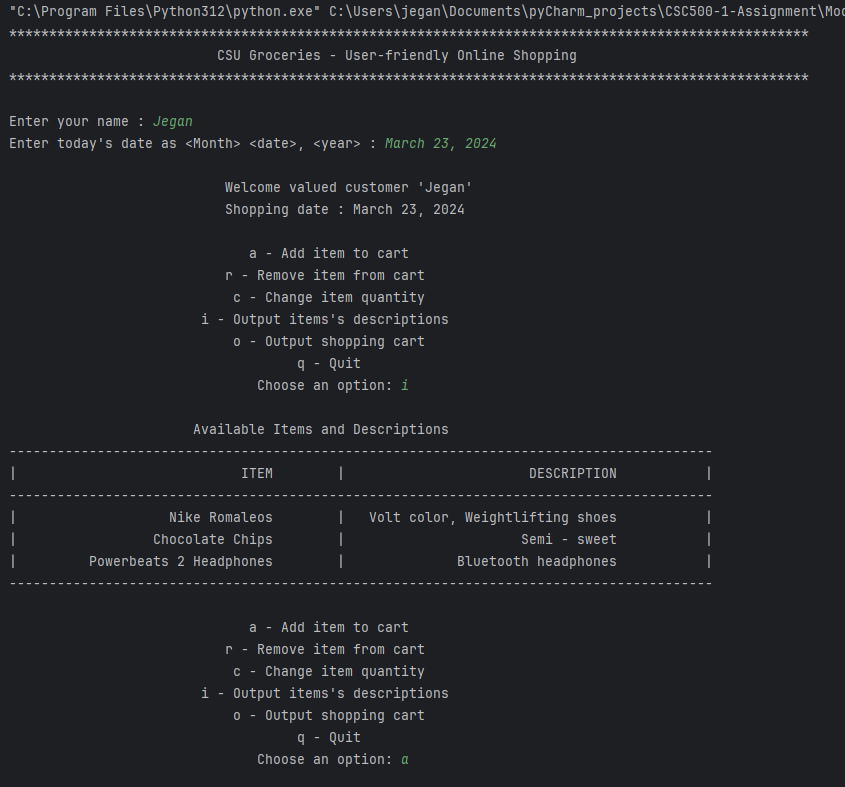
**user\_cart.print\_descriptions()**

**elif option\_selected == 'o':**

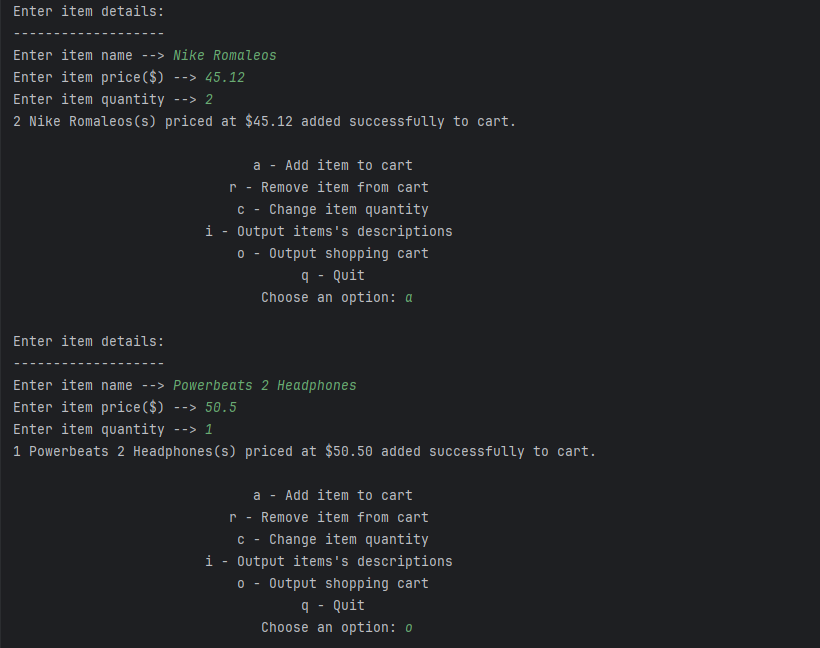
**user\_cart.output\_cart()**

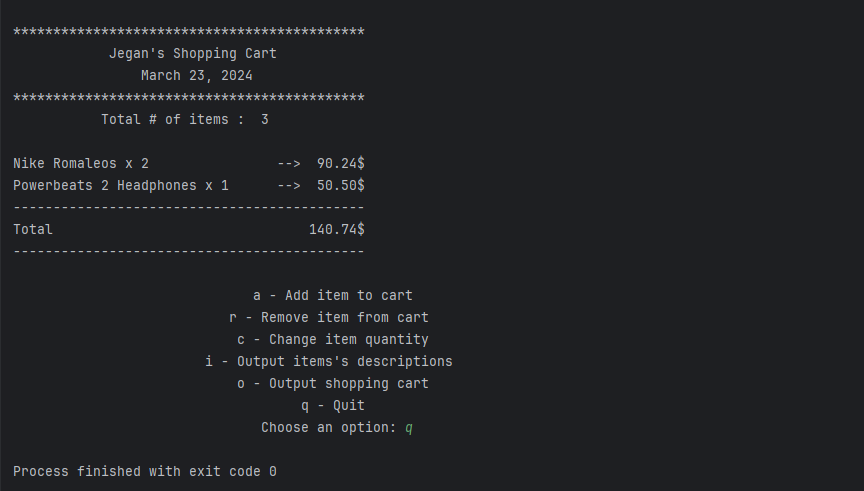
**option\_selected = print\_menu\_options()**

**Output**



Output continued…





**GitHub Repository**

[**CSC500/moduleSix at main · jeganpalaniyandi/CSC500 (github.com)**](https://github.com/jeganpalaniyandi/CSC500/tree/main/moduleSix)