**Module6 Portfolio Milestone Git Repo**

|  |
| --- |
|  |

**source code**

**Problem statement:** Building the ShoppingCart.

**Code:**

|  |
| --- |
| ## Cart Item object  class ItemToPurchase:      def \_\_init\_\_(self, item\_name= None, item\_disc= None, price= None, quantity= None):          self.item\_name = item\_name          self.ite\_disc = item\_disc          self.price = price          self.quantity = quantity      ## Shopping Cart object, contains utility functions to dd, remove, modify and print the Cart items  class ShoppingCart:        def \_\_init\_\_(self, customer\_name= None, current\_date= "January 1, 2020", cart\_items= []):          self.customer\_name = customer\_name          self.current\_date = current\_date          self.cart\_items = cart\_items        # Add item to the cart      def add\_item(self, item: ItemToPurchase):          self.cart\_items.append(item)          print(f"item: {item.item\_name} added to the cart")        # Remove item from the Cart      def remove\_item(self, item\_name):          for item in self.cart\_items:              if (item.item\_name == item\_name):                    ## Logic to ask the user to input how many quantity of the item to remove from the cart                  ## if the input quantity is more than available in the cart, it will remove the item but no error                  remove\_quantity = int(input(f"Cart item {item.item\_name} has {item.quantity} quantity, How many you want to remove: "))                  if(not validate\_item\_date(quantity=remove\_quantity)):                      return                  if(remove\_quantity < item.quantity):                      item.quantity = item.quantity - remove\_quantity                      self.modify\_item(item)                      return                  if(remove\_quantity >= item.quantity):                      self.cart\_items.remove(item)                      print(f"item: {item\_name} removed from the cart\n\n")                      return            print(f"Item: {item\_name} not found in cart. Nothing removed.\n\n")        # Modify the existing item from the Cart      def modify\_item(self, updated\_item: ItemToPurchase):          for item in self.cart\_items:              if (item.item\_name == updated\_item.item\_name):                  is\_updated = False                  # Update the item details                  if(updated\_item.ite\_disc != None):                      item.ite\_disc = updated\_item.ite\_disc                      is\_updated = True                  if (updated\_item.price != None):                      item.price = updated\_item.price                      is\_updated= True                  if (updated\_item.quantity != None):                      item.quantity = updated\_item.quantity                      is\_updated = True                    if(not is\_updated):                      print(f"No updates available for the item: {updated\_item.item\_name}\n\n")                    print(f"item: {updated\_item.item\_name} modified\n\n")                  return            print(f"Item:{updated\_item.item\_name} not found in cart. Nothing modified.\n\n")        # Get total number of items from the Cart      def get\_num\_items\_in\_cart(self) -> int:          return len(self.cart\_items)        # Total cost of the itesm in the Cart      def get\_cost\_of\_cart(self) -> float:          if (self.get\_num\_items\_in\_cart() == 0):              print("SHOPPING CART IS EMPTY")          else:              total\_price = 0              for item in self.cart\_items:                  total\_price += (item.quantity \* item.price)                return total\_price        # Search for item in the cargt      def is\_item\_exist(self, itemname) -> bool:          for item in self.cart\_items:              if (item.item\_name == itemname):                  return True          return False        # Show the Cart item details      def print\_total(self):          print("OUTPUT SHOPPING CART")          print(f"{self.customer\_name}'s Shopping Cart - {self.current\_date}")          print(f"Number of Items: {len(self.cart\_items)}")            total\_price = 0          for item in self.cart\_items:              total\_price += (item.quantity \* item.price)              print(f"{item.item\_name} {item.quantity} @ ${round(item.price, 2): .2f} = ${item.quantity \* item.price}")            print(f"Total: ${round(total\_price, 2): .2f}\n\n")        # Show the Cart item and its description      def print\_descriptions(self):          print("OUTPUT ITEMS' DESCRIPTIONS")          print(f"{self.customer\_name}'s Shopping Cart - {self.current\_date}")          print("Item Descriptions")          for item in self.cart\_items:              print(f"{item.item\_name}: {item.ite\_disc}")          print("\n\n")  ### Main Execution starts here  # validate input values  def validate\_item\_date(price= None, quantity= None) -> bool:      is\_valid\_date = True      if (price != None and price < 0):          print(f"please enter valid price")          is\_valid\_date = False      if (quantity != None and quantity <= 0):          print(f"please enter valid quantity")          is\_valid\_date = False        return is\_valid\_date    # Print Menu  def menu\_items():      print("MENU")      print("a - Add item to cart")      print("r - Remove item from cart")      print("c - Change item description, price, and/or quantity")      print("i - Output items' descriptions")      print("o - Output shopping cart")      print("q - Quit")      print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n")  # Print the Menu and promt user to input the operation to perform, like add, remove, modify, print the Cart details  # optionally we can input 'q' to quit the execution  def print\_menu():      customer\_name = input("Please enter your Name: ").strip()      cart = ShoppingCart(customer\_name=customer\_name)      while True:          print("\n")          menu\_items()          menu\_option = input("Choose an option from Menu: ").strip()          if(menu\_option == "q"):              exit(1)          elif(menu\_option == "a"):              item\_name = input("Enter item name: ").strip()                # item name is mandatory field in the cart              if(item\_name == None or item\_name == ""):                  print("Item name can not be null")                  continue                # Check if the item already exist in the cart              if(cart.is\_item\_exist(itemname=item\_name)):                  print(f"Item {item\_name} already exist in the cart, please choose to update the cart with option 'c'")                  continue                item\_desc = input("Enter item description: ").strip() or None              price = input("item price: $").strip()              price = round(float(price), 2) if price else None                quantity= input("quantity: ").strip()              quantity = int(quantity)              # item name is mandatory field in the cart              if(not validate\_item\_date(price=price, quantity= quantity)):                  continue                item = ItemToPurchase(item\_name=item\_name,item\_disc=item\_desc, price=price, quantity=quantity)              cart.add\_item(item)            elif(menu\_option == "r"):              item\_name = input("Enter item name to remove: ").strip()              cart.remove\_item(item\_name)          elif(menu\_option == "c"):              item\_name = input("Enter item name to modify: ").strip()                if(cart.is\_item\_exist(itemname=item\_name)):                  item\_desc = input("Enter item description: ").strip() or None                    price = input("item price: ").strip()                  price = round(float(price), 2) if price else None                    quantity= input("quantity: ").strip()                  quantity = int(quantity) if quantity else None                    updated\_item = ItemToPurchase(item\_name=item\_name, item\_disc=item\_desc, price=price, quantity=quantity)                  cart.modify\_item(updated\_item)              else:                  print(f"item: {item\_name} not found in the shopping cart")          elif(menu\_option == "i"):              cart.print\_descriptions()          elif(menu\_option == "o"):              cart.print\_total()          else:              print("plase enter valid choice:")      ## Main Function call  if \_\_name\_\_ == "\_\_main\_\_":      print\_menu() |

**Code execution:**

**A screenshot of a computer program

AI-generated content may be incorrect.**

**A screenshot of a computer program

AI-generated content may be incorrect.**

**A screenshot of a computer program

AI-generated content may be incorrect.**

**A screenshot of a computer program

AI-generated content may be incorrect.**

**A screenshot of a computer program

AI-generated content may be incorrect.**