Module 6 Portfolio Milestone

**Figure 1**

*Pseudocode for Shopping Cart Part 2*

START

CLASS ItemToPurchase

ATTRIBUTES

Item\_name = “none”

Item\_price = 0

Item\_quantity = 0

FUNC print\_item\_cost()

Cost = price \* quantity

print (name + quantity + @ + price + = + $ + cost)

CLASS ShoppingCart

ATTRIBUTES

Customer\_name = “none”

Current\_date = “January 1, 2020”

Cart\_items = []

FUNC add\_item(self, item)

cart\_items.append(item)

FUNC remove\_item(self, item\_name)

For item in cart:

If item\_name.lower() == item[“name”]

car\_items.remove(item)

// OUTPUT

Return

Else

PRINT item not found

FUNC modify\_item(self, item)

Pass for now

FUNC get\_num\_items\_in\_cart(self)

Total\_quantity = 0

For item in cart:

Total\_quantity += quantity

Return total\_quantity

FUNC get\_cost\_of\_cart(self)

Total = 0

For item in cart:

Total += price \* quantity

Return total

FUNC print\_total(self)

If items in cart:

// OUTPUT

For item in cart:

item.print\_item\_cost()

// OUTPUT

ELSE:

PRINT cart empty

FUNC print\_descriptions(self)

If items in cart:

// OUTPUT

For item in cart:

Description = item.get(“description”, DEFAULT)

Else:

PRINT cart empty

END

*Note. This pseudocode illustrates a shopping cart script in Python that allows the user to add, remove, modify, and produce output related to their cart.*

**Figure 2**

*Source Code for Shopping Cart Part 2*

***# File Name: Cline\_Jason\_PortfolioMilestoneW6.py***

***# Author: Jason Todd Cline***

***# Institution: Colorado State University Global***

***# Class: CSC500-1***

***# Term: 24FB***

***# Module: 6***

***# Date Created: 09/22/2024***

***# Last Modified: 09/22/2024***

**WIDTH = 40**

**class ItemToPurchase:**

**def \_\_init\_\_(**

**self,**

**item\_name: str = "none",**

**item\_price: float = 0,**

**item\_quantity: int = 0,**

**) -> None:**

**"""**

**Initializes an instance of ItemToPurchase class.**

**Args:**

**item\_name (str): The name of the item. Defaults to "none".**

**item\_price (float): The price of the item. Defaults to 0.**

**item\_quantity (int): The quantity of the item. Defaults to 0.**

**"""**

**self.item = {**

**"name": item\_name,**

**"price": item\_price,**

**"quantity": item\_quantity,**

**}**

**def print\_item\_cost(self) -> None:**

**"""**

**Prints the cost of the item.**

**"""**

**cost: float = self.item["price"] \* self.item["quantity"]**

**print(**

**f"{self.item['name']} {self.item['quantity']} @ ${self.item['price']:.2f} = ${cost:.2f}".center(WIDTH)**

**)**

**class ShoppingCart:**

**def \_\_init\_\_(**

**self,**

**customer\_name: str = "none",**

**current\_date: str = "January 1, 2020",**

**) -> None:**

**self.customer\_name: str = customer\_name**

**self.current\_date: str = current\_date**

**self.cart\_items: list = []**

**def add\_item(self, item: ItemToPurchase) -> None:**

**"""**

**Adds an item to cart\_items list. Has parameter item of type ItemToPurchase. Does not return anything.**

**Args:**

**item (ItemToPurchase): The item to be added to the cart\_items list.**

**"""**

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

**def remove\_item(self, item\_name: str) -> None:**

**"""**

**Removes item from cart\_items list. Has a string (an item's name) parameter. Does not return anything.**

**If item name cannot be found, output this message: Item not found in cart. Nothing removed.**

**"""**

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

**if item.item["name"].lower() == item\_name.lower():**

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

**print(f"{item\_name} has been removed from the cart")**

**return**

**else:**

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

**def modify\_item(item: ItemToPurchase) -> None:**

**"""**

**Modifies an item's description, price, and/or quantity. Has parameter ItemToPurchase. Does not return anything.**

**If item can be found (by name) in cart, check if parameter has default values for description, price, and quantity. If not, modify item in cart.**

**If item cannot be found (by name) in cart, output this message: Item not found in cart. Nothing modified.**

**"""**

**pass**

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

**"""**

**Returns quantity of all items in cart. Has no parameters.**

**"""**

**total\_quantity: int = 0**

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

**total\_quantity += item.item["quantity"]**

**return total\_quantity**

**def get\_cost\_of\_cart(self) -> float:**

**"""**

**Determines and returns the total cost of items in cart. Has no parameters.**

**"""**

**total: int = 0**

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

**total += item.item["price"] \* item.item["quantity"]**

**return total**

**def print\_total(self) -> None:**

**"""**

**Outputs total of objects in cart.**

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

**"""**

**if self.cart\_items:**

**print(f"{self.customer\_name}'s Shopping Cart - {self.current\_date}".center(WIDTH))**

**print(f"Number of items: {len(self.cart\_items)}".center(WIDTH))**

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

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

**print(f"Total: ${self.get\_cost\_of\_cart():.2f}".center(WIDTH))**

**print()**

**else:**

**print(f"SHOPPING CART IS EMPTY".center(WIDTH))**

**def print\_descriptions(self) -> None:**

**"""**

**Outputs each item's description.**

**"""**

**if self.cart\_items:**

**print(**

**f"{self.customer\_name}'s Shopping Cart - {self.current\_date}".center(WIDTH)**

**)**

**print(f"Item Descriptions".center(WIDTH))**

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

**description = item.item.get("description", "No description available")**

**print(f"{item.item['name']}: {description}".center(WIDTH))**

**else:**

**print(f"{'SHOPPING CART IS EMPTY'}".center(WIDTH))**

**def main() -> None:**

**"""**

**Main function to...**

**"""**

**cart: ShoppingCart = ShoppingCart("John Doe", "February 1, 2020")**

**def print\_menu(cart: ShoppingCart):**

**while True:**

**print()**

**print(f"{'MENU':^{WIDTH}}")**

**print(f"{'a - Add item to cart':^{WIDTH}}")**

**print(f"{'r - Remove item from cart':^{WIDTH}}")**

**print(f"{'c - Change item quantity':^{WIDTH}}")**

**print(f"{'i - Output items\' descriptions':^{WIDTH}}")**

**print(f"{'o - Output shopping cart':^{WIDTH}}")**

**print(f"{'q - Quit':^{WIDTH}}")**

**print(f"{'Choose an option: ':^{WIDTH}}")**

**user\_input = input(f"{'':^{WIDTH / 2}}")**

**match user\_input:**

**case "q":**

**print(f"{'Thanks for shopping with us. Goodbye!':^{WIDTH}}")**

**return**

**case "a":**

**item\_name: str = input("Enter the item name: ")**

**item\_price: float = float(input("Enter item price: "))**

**item\_quantity: int = int(input("Enter item quantity: "))**

**cart.add\_item(ItemToPurchase(item\_name, item\_price, item\_quantity))**

**case "r":**

**item\_name = input("Enter the item name to remove: ")**

**cart.remove\_item(item\_name)**

**case "c":**

**item\_name = input("Enter the item name to modify: ")**

**cart.modify\_item(item\_name)**

**case "i":**

**print()**

**print(f"{'OUTPUT ITEMS\' DESCRIPTIONS':^{WIDTH}}")**

**cart.print\_descriptions()**

**case "o":**

**print()**

**print(f"{'OUTPUT SHOPPING CART':^{WIDTH}}")**

**cart.print\_total()**

**case \_:**

**print(f"{'Invalid input. Please choose a valid option.':^{WIDTH}}")**

**print\_menu(cart)**

**print("\n")**

***# Hard-coded for assignment as proof of concept without user input:***

***# Create items with hardcoded descriptions as placeholders***

**item1 = ItemToPurchase("Nike Romaleos", 189.00, 2)**

**item1.item['description'] = "Volt color, Weightlifting shoes" *# Placeholder description for now***

**item2 = ItemToPurchase("Chocolate Chips", 3.00, 5)**

**item2.item['description'] = "Semi-sweet" *# Placeholder description for now***

**item3 = ItemToPurchase("Powerbeats 2 Headphones", 128.00, 1)**

**item3.item['description'] = "Bluetooth headphones" *# Placeholder description for now***

***# Add items to shopping cart***

**shopping\_cart = ShoppingCart(customer\_name = "John Doe", current\_date = "February 1, 2020")**

**shopping\_cart.add\_item(item1)**

**shopping\_cart.add\_item(item2)**

**shopping\_cart.add\_item(item3)**

***# Print item descriptions as proof of concept***

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

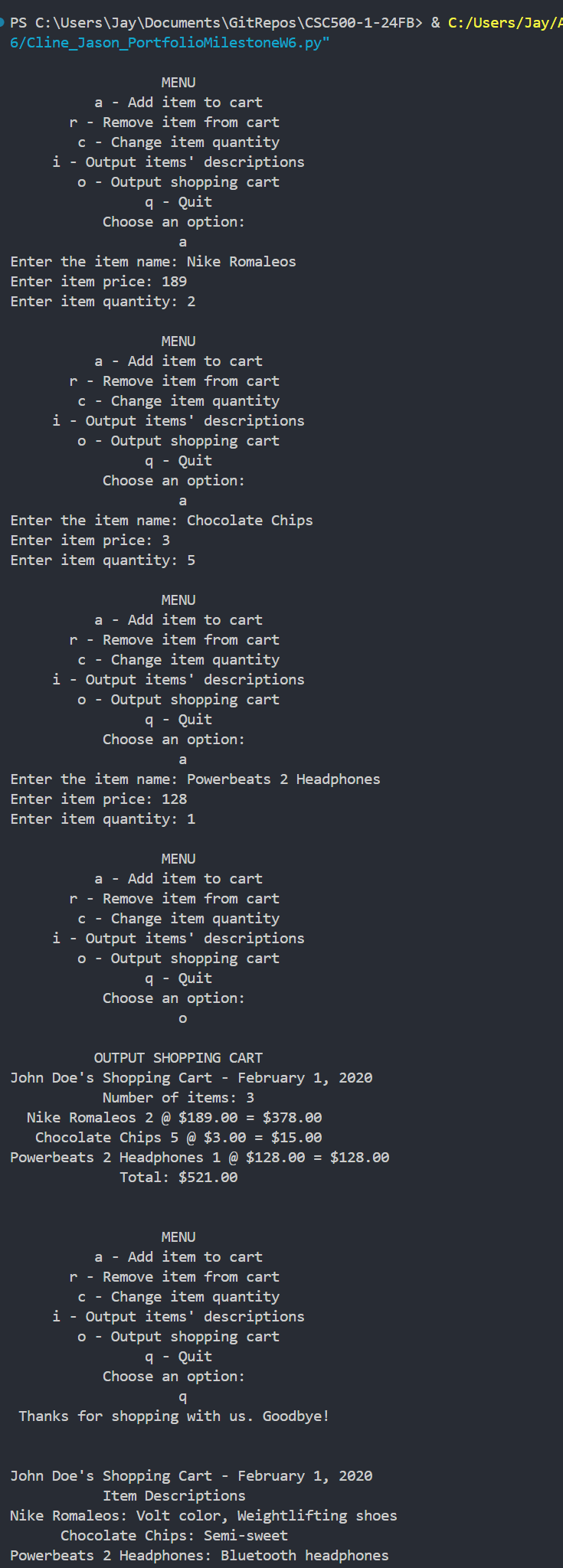
**if \_\_name\_\_ == "\_\_main\_\_":**

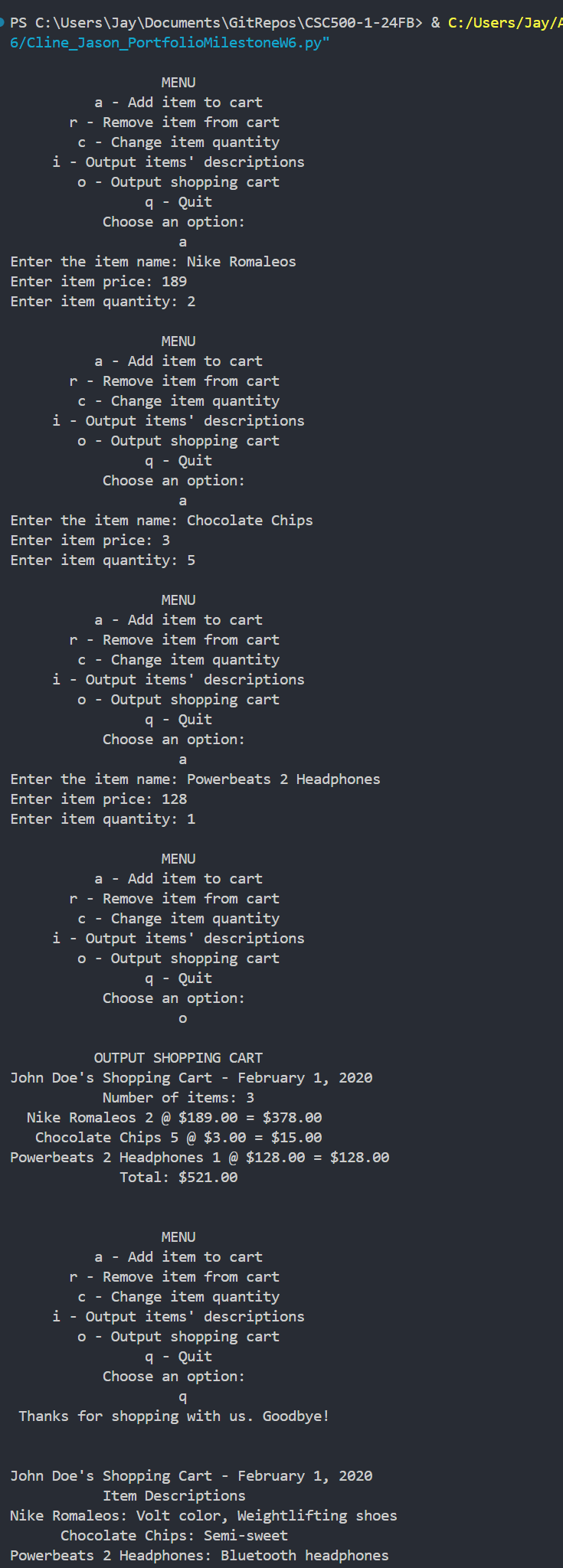
**main()**

*Note. Source code that illustrates a shopping cart script in Python that allows the user to add, remove, modify, and produce output related to their cart. Some methods are incomplete at the moment besides adding and displaying item totals and descriptions.*

**Figure 3**

*Execution and Testing for Shopping Cart Part 2*

**

**

*Note. Execution and testing of a shopping cart in Python that allows the user to add, remove, change, and output descriptions and total. Testing shows adding items, displaying totals, and displaying descriptions. Descriptions are hard-coded as part of the second milestone as proof of concept.*

References

Cline, J. T. [Jay4rmTheBay]. (2024). *CSC500-1-24FB* [Source code]. GitHub.<https://github.com/Jay4rmTheBay/CSC500-1-24FB>