###############################################################

Student: Aditya Singh Sandhu

Course: CSC 500 – Principles of Programming

Module: 8 – Final Portfolio Assignment

###############################################################

Python3 Code ~

# COPY THIS LINE OF CODE - BEGINNING

class ObjectToProcure:

def \_\_init\_\_(self, product\_name="none", dollar\_value\_of\_product=0, numerical\_quantity\_of\_product=0):

self.product\_name = product\_name

self.dollar\_value\_of\_product = dollar\_value\_of\_product

self.numerical\_quantity\_of\_product = numerical\_quantity\_of\_product

def console\_log\_product\_cost(self):

total\_grocery\_cart\_cost = self.dollar\_value\_of\_product \* self.numerical\_quantity\_of\_product

print(

f"{self.product\_name} {self.numerical\_quantity\_of\_product} @ ${self.dollar\_value\_of\_product} = ${total\_grocery\_cart\_cost}")

def initial\_product\_to\_cart(cart):

product\_name = input("Whats the description of the first Object:\n")

product\_dollar\_value = float(input(f"Whats the dollar value of the {product\_name}:\n"))

product\_quantity = int(input(f"Whats the quantity of the {product\_name}:\n"))

Object = ObjectToProcure(product\_name, product\_dollar\_value, product\_quantity)

cart.add\_product(Object)

print("\nHERE IS THE TOTAL COST OF THE Object")

Object.console\_log\_product\_cost()

total\_grocery\_cart\_cost = cart.get\_cost\_of\_cart()

print(f"\nTotal: ${total\_grocery\_cart\_cost}")

class GroceryCartAtStore:

def \_\_init\_\_(self, grocery\_shoppers\_name="none", todays\_shopping\_date="September 7, 2024"):

self.grocery\_shoppers\_name = grocery\_shoppers\_name

self.todays\_shopping\_date = todays\_shopping\_date

self.products\_in\_shopping\_cart = []

def add\_product(self, Object: ObjectToProcure):

self.products\_in\_shopping\_cart.append(Object)

def remove\_product(self, product\_name: str):

found = False

for Object in self.products\_in\_shopping\_cart:

if Object.product\_name == product\_name:

self.products\_in\_shopping\_cart.remove(Object)

found = True

break

if not found:

print("The Object to be Removed, wasn't found in the cart, So nothing was REMOVED.")

def adjust\_Object(self, product\_to\_modify: ObjectToProcure):

found = False

for Object in self.products\_in\_shopping\_cart:

if Object.product\_name == product\_to\_modify.product\_name:

if product\_to\_modify.dollar\_value\_of\_product != 0:

Object.dollar\_value\_of\_product = product\_to\_modify.dollar\_value\_of\_product

if product\_to\_modify.numerical\_quantity\_of\_product != 0:

Object.numerical\_quantity\_of\_product = product\_to\_modify.numerical\_quantity\_of\_product

found = True

break

if not found:

print("The Object to be Modified, wasn't found in the cart, So nothing was MODIFIED.")

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

total\_quantity = sum(Object.numerical\_quantity\_of\_product for Object in self.products\_in\_shopping\_cart)

return total\_quantity

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

total\_grocery\_cart\_cost = sum(Object.dollar\_value\_of\_product \* Object.numerical\_quantity\_of\_product for Object in

self.products\_in\_shopping\_cart)

return total\_grocery\_cart\_cost

def total\_output(self):

print(f"{self.grocery\_shoppers\_name}'s Grocery Cart - {self.todays\_shopping\_date}")

if not self.products\_in\_shopping\_cart:

print("THERE IS NOTHING IN THE Grocery Cart, IT IS EMPTY!")

else:

print(f"Total Number of Objects in the Grocery Cart: {self.get\_num\_Objects\_in\_cart()}")

for Object in self.products\_in\_shopping\_cart:

Object.console\_log\_product\_cost()

print(f"\nTotal Cost of the Cart: ${self.get\_cost\_of\_cart()}")

def print\_identity(self):

print(f"{self.grocery\_shoppers\_name}'s Grocery Cart - {self.todays\_shopping\_date}")

print("Object identity")

for Object in self.products\_in\_shopping\_cart:

print(f"{Object.product\_name}: {Object.dollar\_value\_of\_product}")

def print\_menu(cart: GroceryCartAtStore):

menu = """

MENU

ADD - Add Object to cart

REMOVE - Remove Object from cart

MODIFY - Modify Object quantity

IDENTIFY - Output Objects' identity

OUTPUT - Output Grocery Cart

QUIT - Quit

"""

while True:

print(menu)

selection = input(f"{cart.grocery\_shoppers\_name}, choose an option to edit your Grocery Cart:\n")

if selection == 'QUIT':

break

elif selection == 'ADD':

name = input(

f"{cart.grocery\_shoppers\_name}, what's the description of the Object to add to the existing Grocery Cart:\n")

price = float(input(

f"{cart.grocery\_shoppers\_name}, what's the price of the {name} being added to the existing Grocery Cart:\n"))

quantity = int(input(

f"{cart.grocery\_shoppers\_name}, what's the quantity of the {name} being added to the existing Grocery Cart:\n"))

new\_Object = ObjectToProcure(name, price, quantity)

cart.add\_product(new\_Object)

elif selection == 'REMOVE':

name = input(

f"{cart.grocery\_shoppers\_name}, what's the description of the Object to remove from the existing Grocery Cart:\n")

cart.remove\_product(name)

elif selection == 'MODIFY':

name = input(

f"{cart.grocery\_shoppers\_name}, what's the description of the Object to modify in the existing Grocery Cart:\n")

price = float(

input(f"{cart.grocery\_shoppers\_name}, what's the new price of {name} (or 0 to leave unchanged):\n"))

quantity = int(

input(f"{cart.grocery\_shoppers\_name}, what's the new quantity of {name} (or 0 to leave unchanged):\n"))

modified\_Object = ObjectToProcure(name, price, quantity)

cart.adjust\_Object(modified\_Object)

elif selection == 'IDENTIFY':

cart.print\_identity()

elif selection == 'OUTPUT':

cart.total\_output()

def main():

grocery\_shoppers\_name = input("What's the Grocery Shoppers Name:\n")

todays\_shopping\_date = input("What's the Date of the Grocery Shopping [Format ex: Month Date, Year [Sept 1st, 2024]]:\n")

print(f"Client name: {grocery\_shoppers\_name}")

print(f"Today's date: {todays\_shopping\_date}")

cart = GroceryCartAtStore(grocery\_shoppers\_name, todays\_shopping\_date)

initial\_product\_to\_cart(cart)

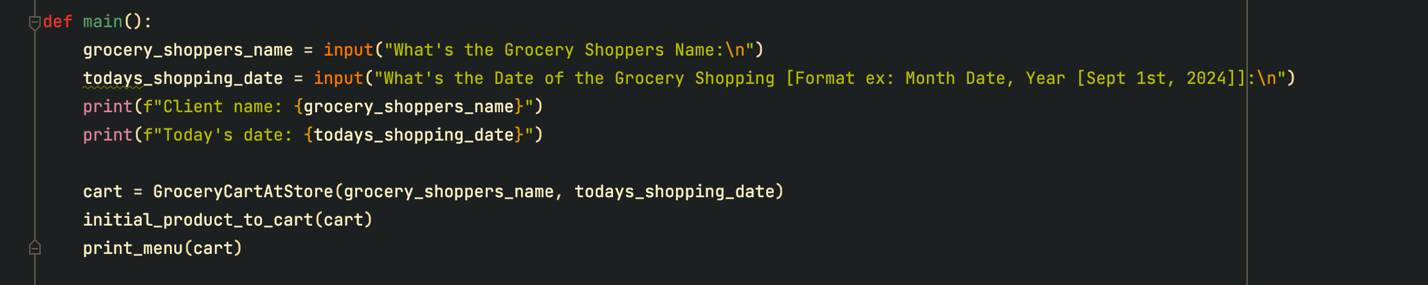
print\_menu(cart)

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

main()

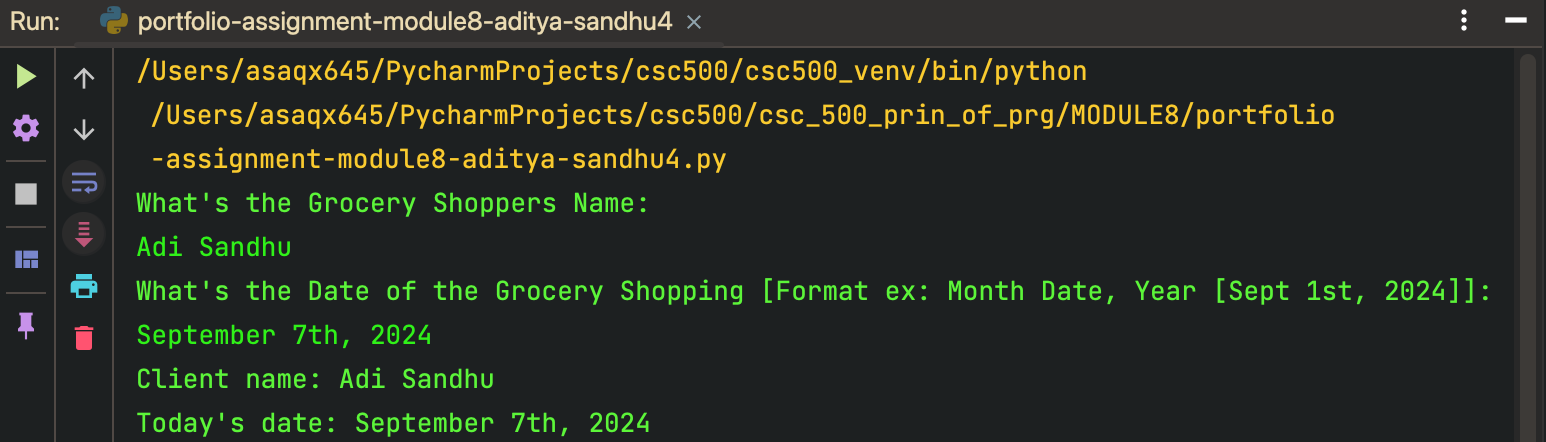
# COPY THIS LINE OF CODE – ENDING

"Screenshot 1," "Main Function Line one and two execution, Input Prompt from user”

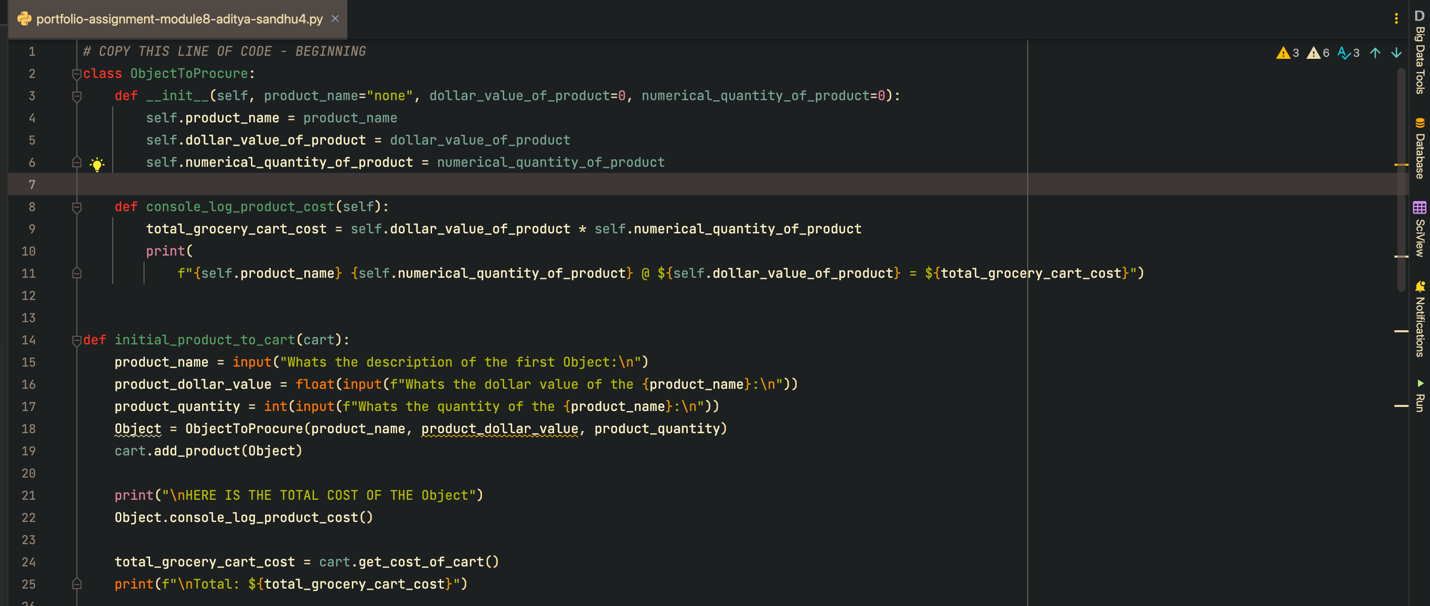


Asking for the Grocery Shoppers Name and the format of the data when the shopping is occurring

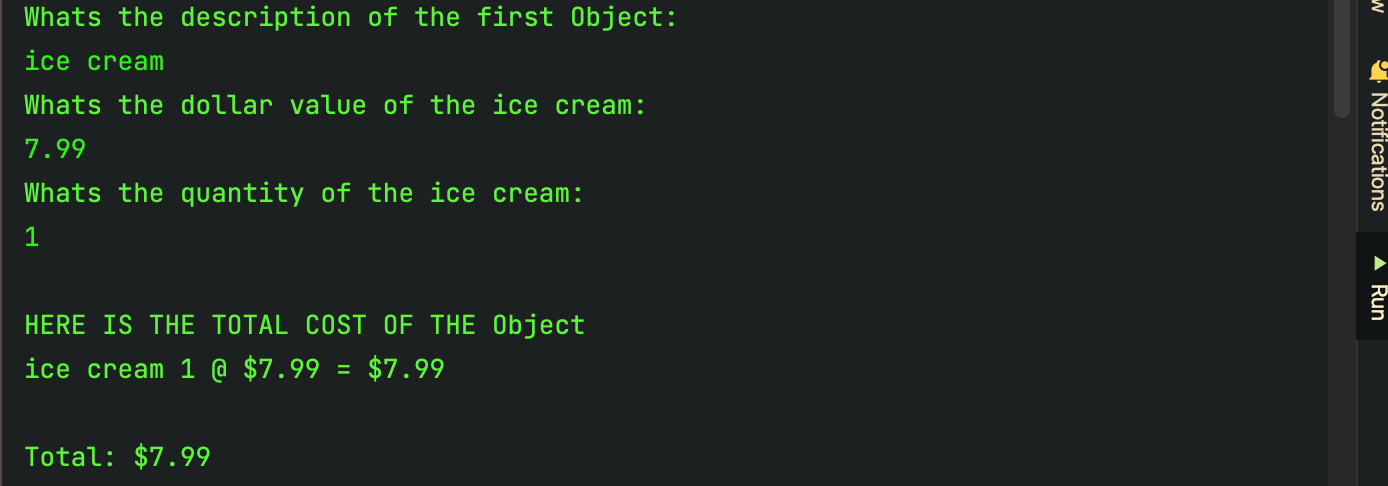
“Screenshot 2”, “Output of the Name and Date Input from Shopper”



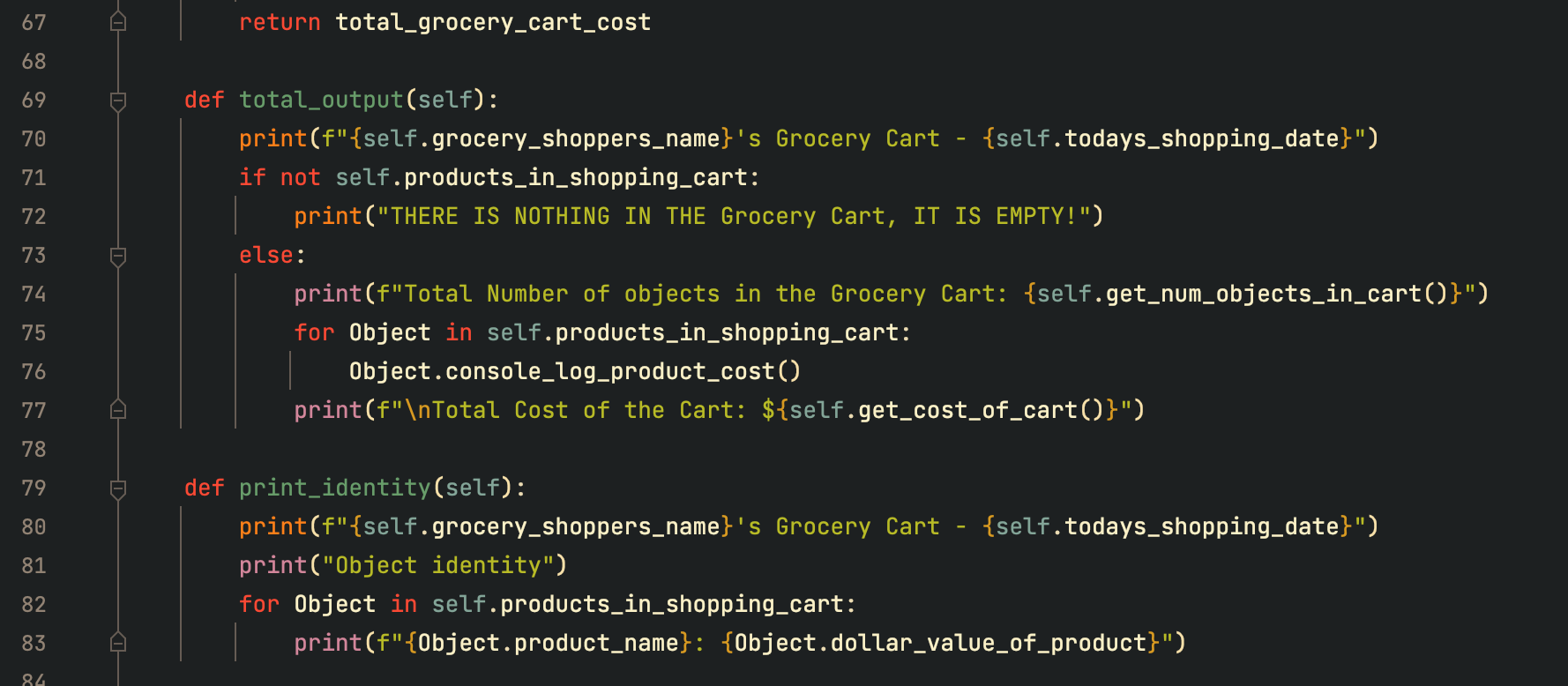
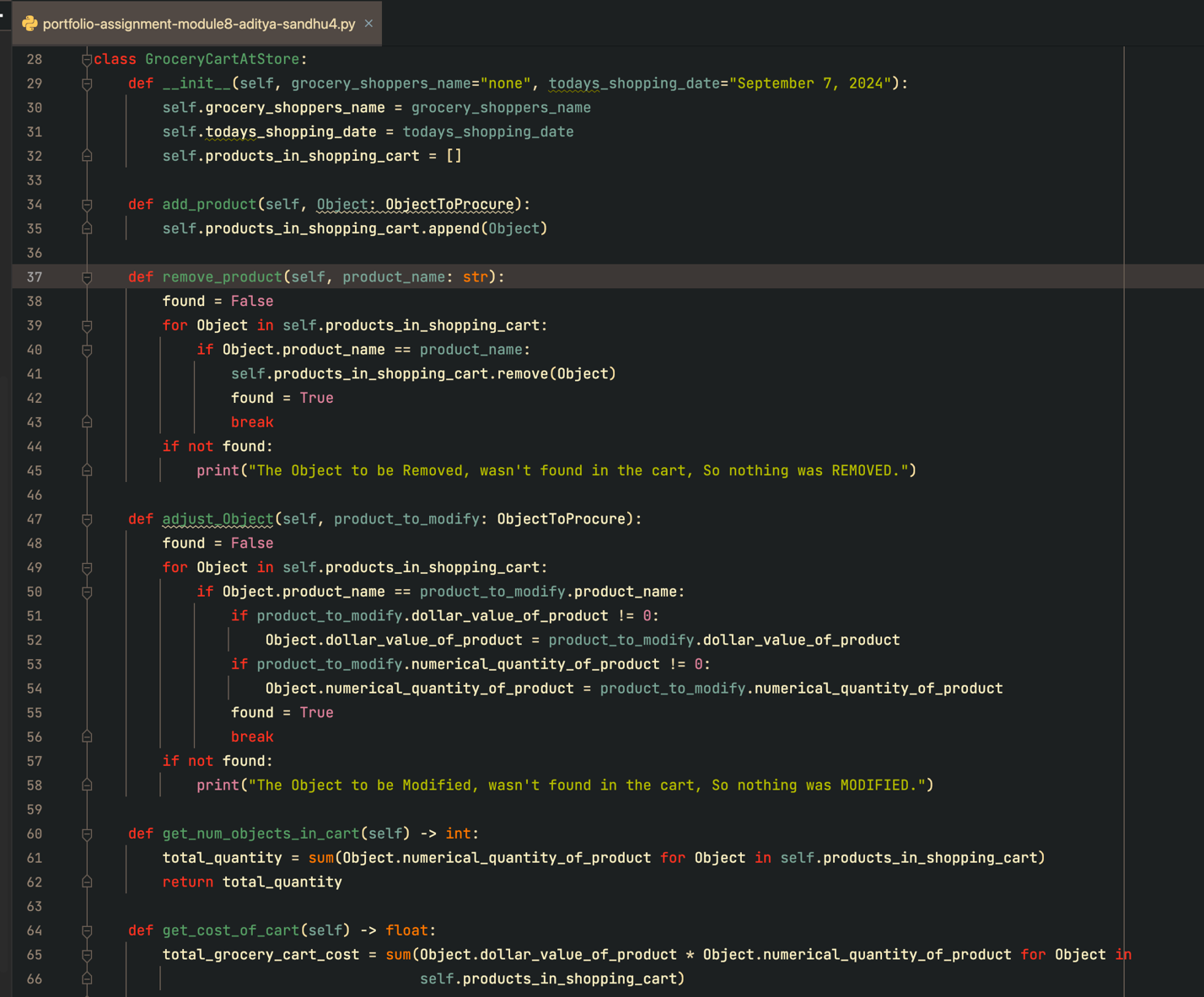
“Screenshot 3”, “The ObjectToProcure Function and its Cart Creation Logic”



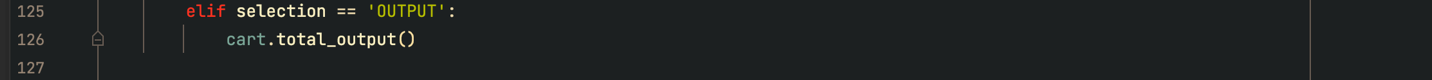
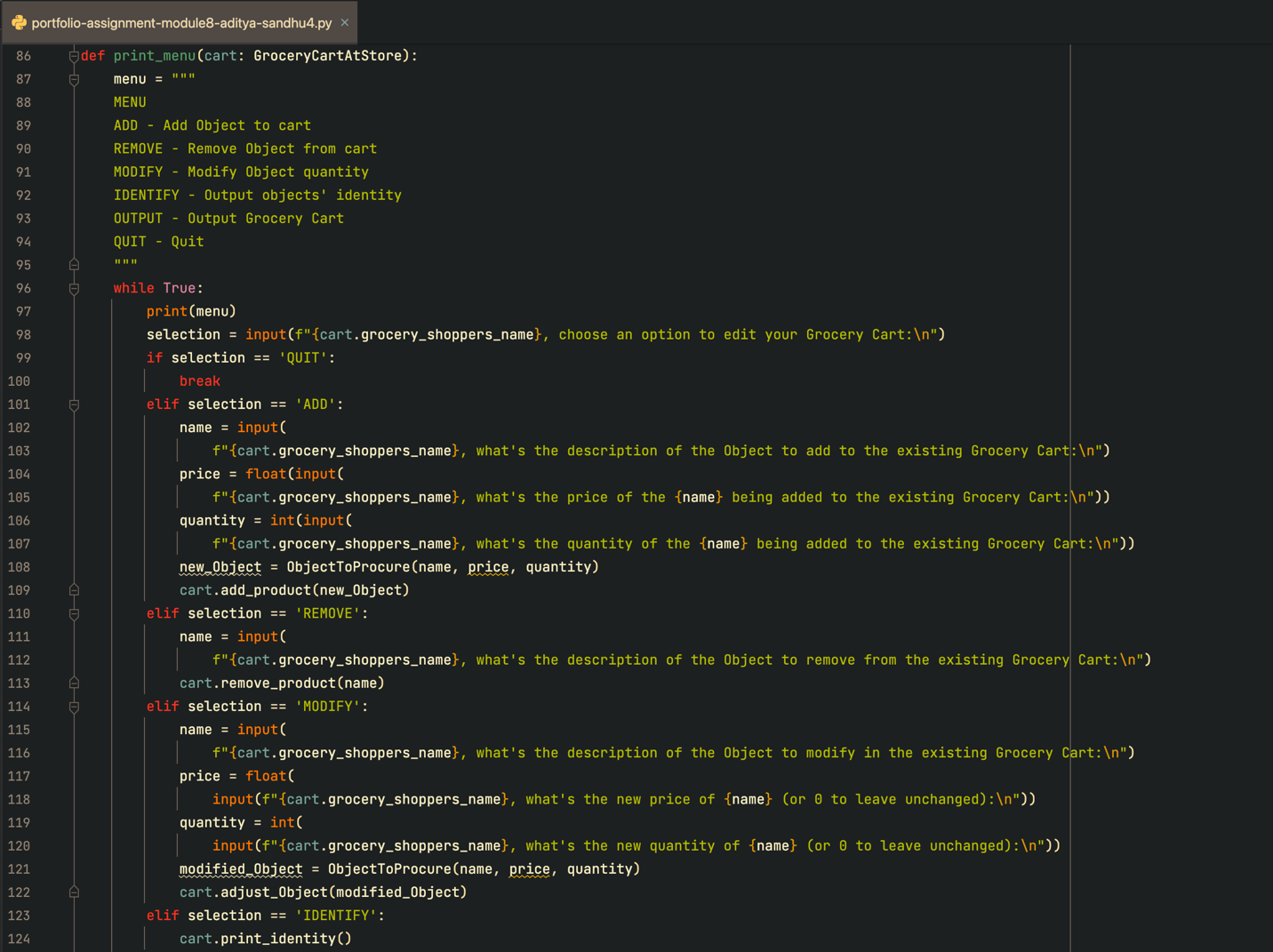
“Screenshot 4”, “Output on the Console of the Class ObjectToProcure”



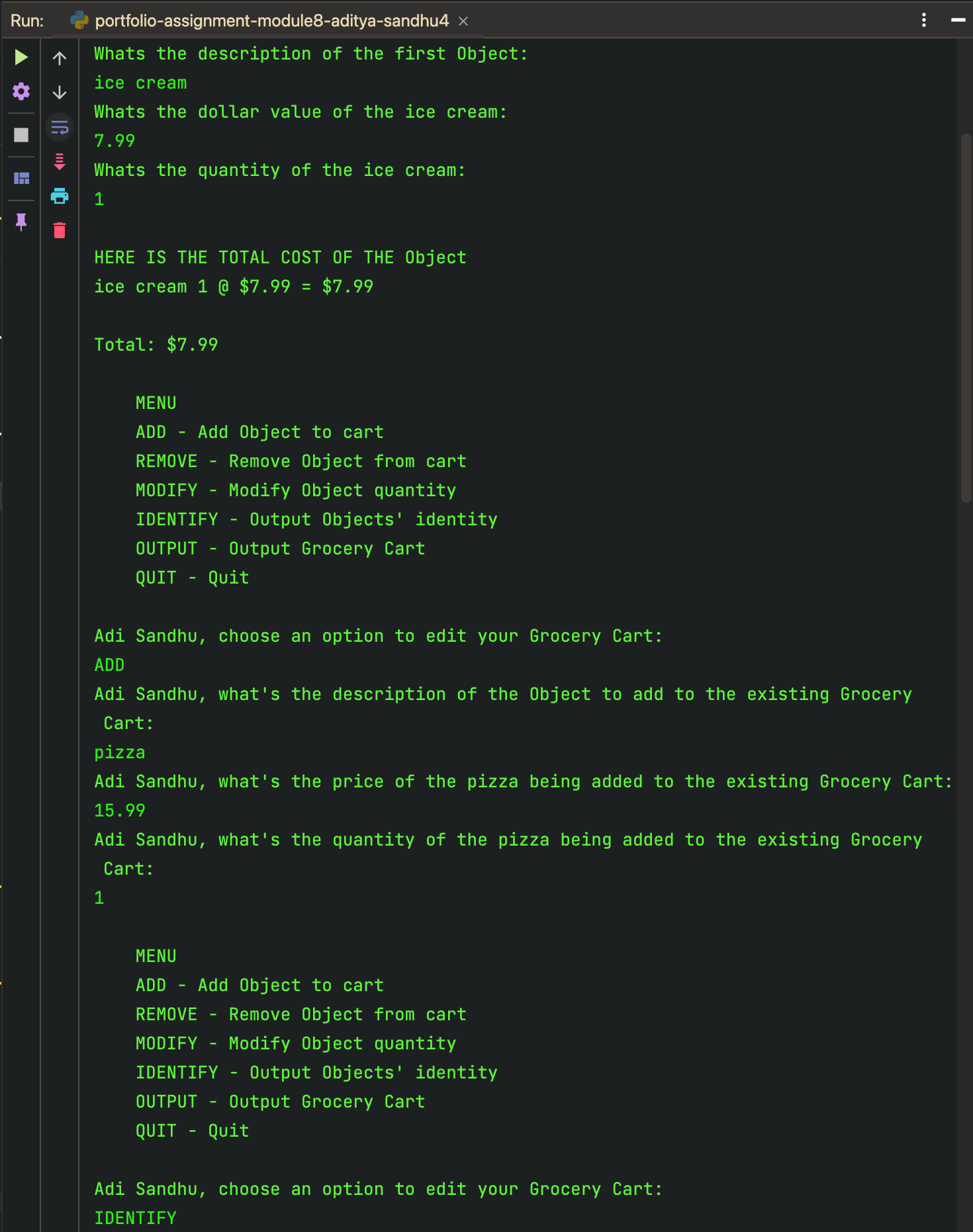
“Screenshot 5”, “The GroceryCartAtStore Class, has the add\_product, remove\_product, adjust\_object, get\_num\_objects\_in\_cart, get\_cost\_of\_cart, total\_output, print\_identity functions that are passed to print\_menu function to display ”



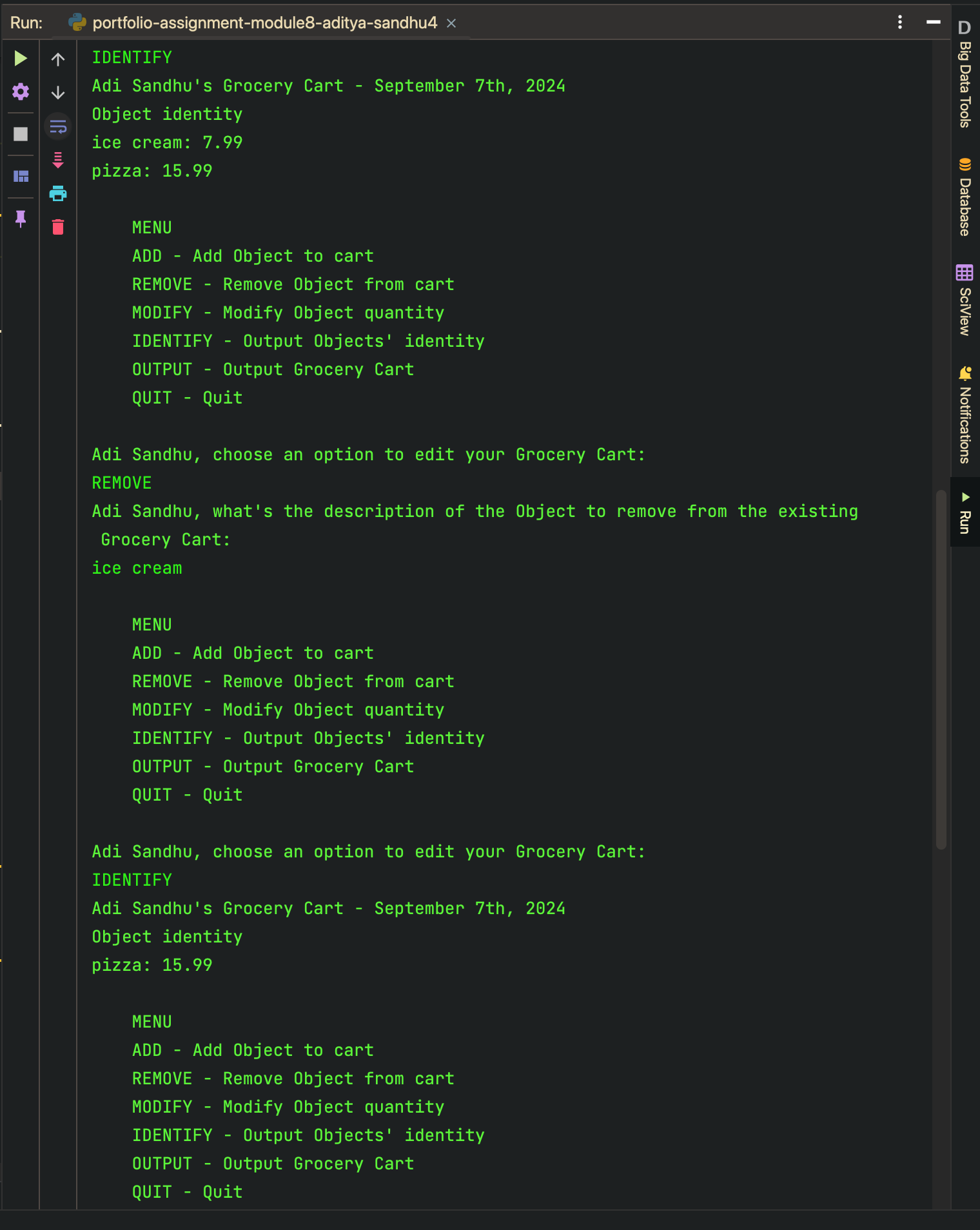
### “Screenshot 6”, “The print\_menu function with the menu, and the ‘if, elif control flow that calls the add\_product, remove\_product, adjust\_product, print\_identity ,and total\_output function’s’ from the GroceryCartAtStore Class”



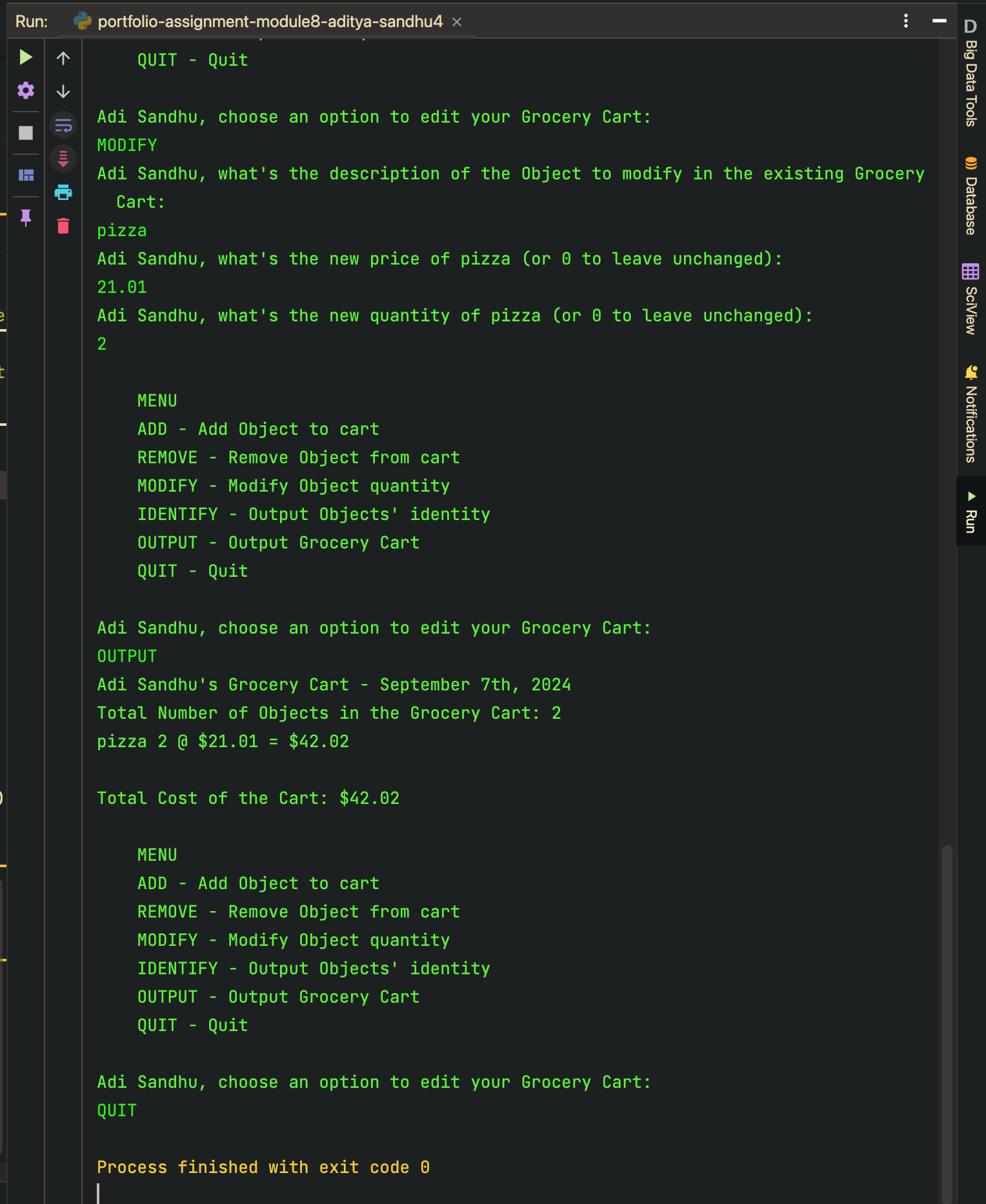
“Screenshot 7”, “Output Console, First Object Input request, input is ice cream, at price 7.99, then print out of cost of object entered, and the price, with quantity 1, and Total of 7.99, Then Menu Displayed With Options, User input ADD, and pizza object added, with price 15.99 and quantity 1, then MENU Again displayed and user input IDENTIFY Option”



“Screenshot 8”, “Console Output, user input IDENTIFY, and the output of the Grocery Cart with data and objects pizzam, and ice cream in it is displayed with price, and MENU Displayed again, User selects REMOVE Option and selects ice cream, MENU Displayed again and user inputs IDENTIFY to confirm that only pizza left in grocery cart. Menu displayed again.”



“Screenshot 9”, “Console Output, User inputs MODIFY and then enters in the prompts, pizza, at a new price of 21.01 and a quantity of 2, then the Menu is displayed, again and the user selects OUTPUT and gets a view of the Shoppers Grocery Cart, with the date, the total number of objects, their total quantity and list of total price of pizza at 42.02, and the Grocery cart total as well. Then the Menu is displayed and the user selects QUIT to end the program with a 0 exit code”



This Python program models a grocery shopping experience using object-oriented programming principles. It defines two main classes: `ObjectToProcure` and `GroceryCartAtStore`. The `ObjectToProcure` class represents individual items to be added to the grocery cart, with attributes for product name, price, and quantity. A method `console\_log\_product\_cost()` calculates and logs the total cost of each item based on its price and quantity.

The `GroceryCartAtStore` class manages a grocery cart with attributes for the shopper's name, date, and a list of products. It provides methods to add, remove, and modify items in the cart. The cart also tracks the total number of products and calculates the total cost using the `get\_num\_objects\_in\_cart()` and `get\_cost\_of\_cart()` methods. The method `total\_output()` displays a summary of all items in the cart, including their individual and total costs.

In the `main()` function, the program collects user input for the shopper’s name and date, then initializes an instance of `GroceryCartAtStore`. It adds the first product to the cart by calling `initial\_product\_to\_cart()`, which prompts the user for details of the item. The `print\_menu()` function allows the user to add, remove, modify, or view the items in the cart through a command-line menu, providing a basic user interface for interacting with the cart.

Key concepts in this script include encapsulation (hiding data within objects) and methods to modify, log, and manage the cart’s contents. The use of getter and setter methods ensures that attributes are accessed and modified in a controlled manner. This program simulates a simple but functional shopping cart system, illustrating object-oriented design and user interaction.

GITHUB LINK - <https://github.com/65AR645ASAN/csc_500_prin_of_prg/blob/main/MODULE8/final-portfolio-assignment-module-8-aditya-sandhu.docx>

APA citations:

1. Python-Course.eu. (n.d.). Inheritance in Python. Python-Course.eu. Retrieved September 7, 2024, from <https://python-course.eu/oop/inheritance.php#:~:text=The%20class%20from%20which%20a,heir%20class%20or%20child%20class>.
2. GeeksforGeeks. (2024). How to read from a file in Python. GeeksforGeeks. Retrieved September 7, 2024, from <https://www.geeksforgeeks.org/how-to-read-from-a-file-in-python/>
3. W3Schools. (n.d.). Python regex. W3Schools. Retrieved September 7, 2024, from <https://www.w3schools.com/python/python_regex.asp>
4. W3Schools. (n.d.). Python lambda. W3Schools. Retrieved September 7, 2024, from <https://www.w3schools.com/python/python_lambda.asp>
5. Python Software Foundation. (n.d.). Regular expression operations. Python.org. Retrieved September 7, 2024, from <https://docs.python.org/3/howto/regex.html>