# Module 6: Portfolio Milestone

Maurice Henning

Colorado State University Global

CSC500-1: Principals of Programming

Dr. Steven Evans

April 30, 2024

**Online Shopping Cart**

This module is a continuation of module 4 which modifies the existing online cart

to include new parameters. In this second edition I’ve updated the online shopping cart with the

following data attributes and related methods; parameterized constructor, attributes, customer

name, customer date, cart items, methods, add an item, remove item, modify item, get num,

get cost, print total, print. Additionally, I’ve removed some code to improve code readability and

functionality.

Step 4: The foundation of the existing code for the online shopping cart remains intact however

I’ve modified the code to accommodate these new features.

The ItemToPurchase label which initializes the item name, price, and quantity remains but now

I’ve added a parameterized constructor by initializing the name and date attributes.

A parameterized constructor enables classes to jump-start their attributes with specific values.

It accepts one or more parameters and initializes the instance variables of the class with values

passed as arguments. Hence we create a class ShoppingCart and define customer name, date, and

items. The customer\_name label is initialized to “none” while the date is initialized to January

01, 2020. Continuing with the shopping cart I create the ability to add an item to the cart by

defining a variable label add\_item. This is followed by the ability to remove an item defining the

label remove\_item and creating an ‘if’ condition that stipulates that if an item is not in

the cart it displays ‘item not found in cart. Nothing removed’.

In the next series, I define cart modification with the label modify\_item. You can

alter the description, price, and quantity in this feature which includes a condition whereby if an

item is not found in the cart the application displays ‘Item not found in cart. Nothing modified.’

Next, I defined a variable for the number of items in the cart labeled get\_num\_items\_in\_cart.

Followed with a ‘for’ loop which itemizes and returns the number of items in the cart. This same

method is applied to the variables to define the cost of cart, the print total, and the print

description.

Step 5: Here we are asked to define the menu labeled print\_menu and follow it up with print

statements for “a - Add item to cart r - Remove item from cart, c - Change item quantity,

i - Output items' descriptions, o - Output shopping cart, q – Quit”.

Because this menu has so many options the ‘elif’ statement was used after the initial ‘if’

statement for adding an item and continues to the last state and final ‘else’ statement which prints

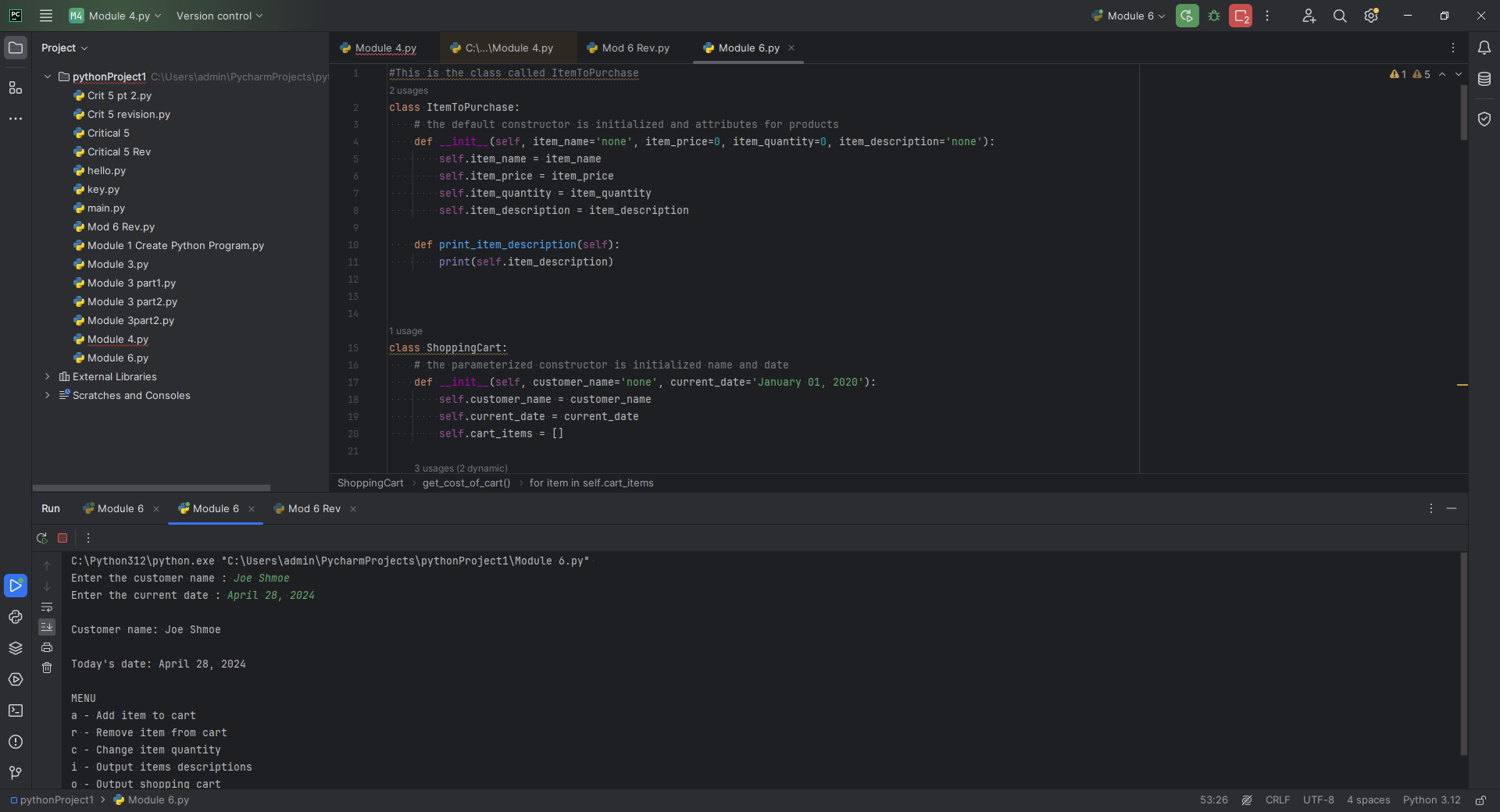
“Invalid choice. Please try again” if the user selects a letter that is not part of the menu selection.

Step 6: The final touches are added when I implement the output shopping cart menu. This

displays the customer name, date, number of items along with the total cost of all items added

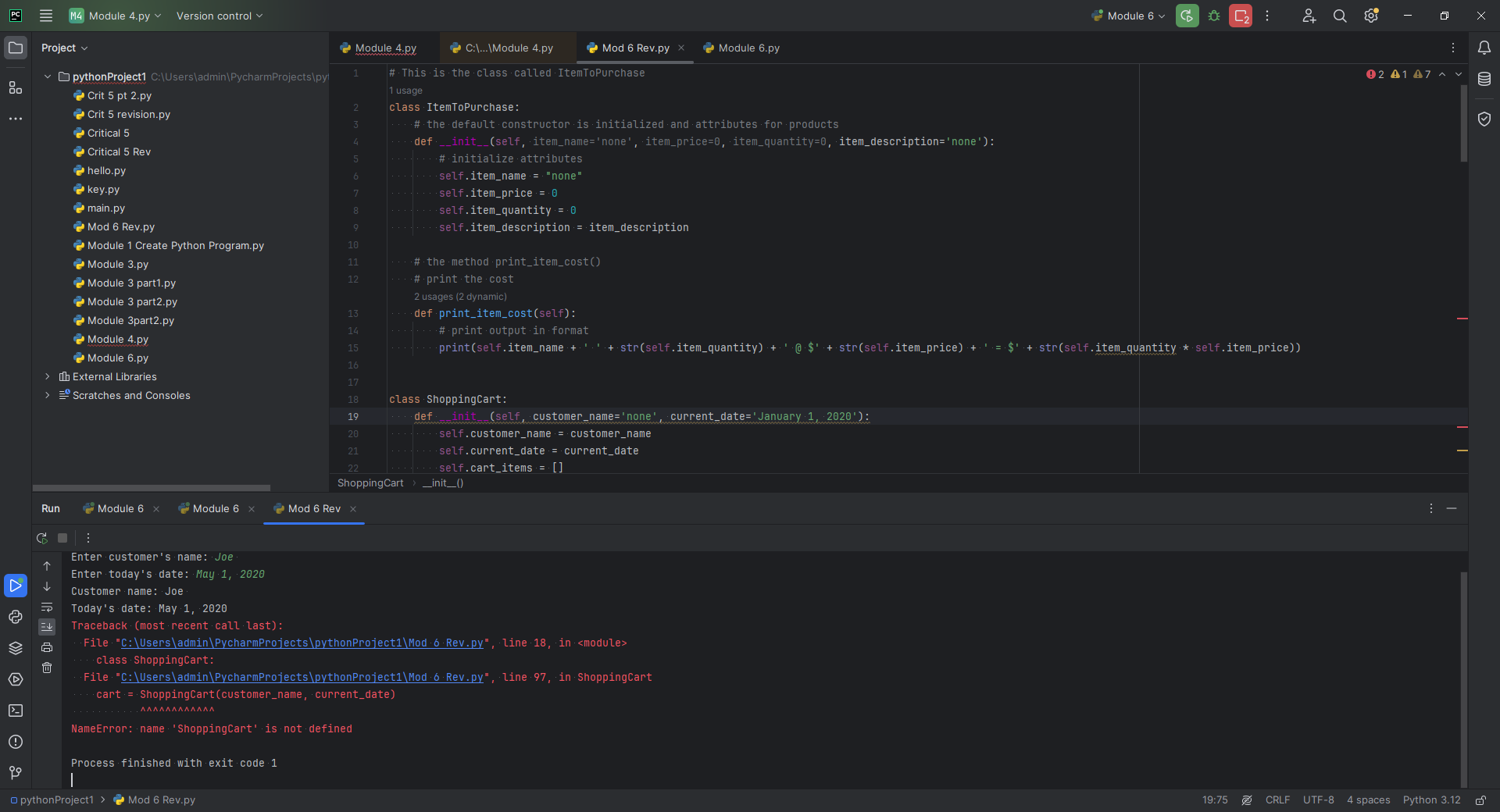
together resulting in a total cost.

**FINAL RESULT**



One of the challenges I encountered with updating this application was receiving errors in defining the shopping cart class. I made adjustments to the code eliminating some code and realignment to get around the error.

See below



**Pseudocode**

**Step 4:**

**1: INIT** class Item to Purchase **2: SET initialization** for item attributes; name, price, quantity, description  **3. INIT class Shopping Cart   
4. SET customer name, date, items all variables associated with shopping cart**

**Step 5:**

**1: DEFINE** and **DISPLAY** menu description and all variables associated **2. IF condition** followed by a series of **ELIF** statements

**Step 6:**

**1. SHOW** cart menu item(s) and total cost