Assignment 15 – Object-Oriented Programming

Techniques for Object-Oriented Design

A class in an object-oriented program typically defines an object that corresponds with an object, or entity, in the real world

Five Steps for Designing an OOP

1. Identify the data attributes
2. Subdivide each attribute into its smallest useful components
3. Identify the classes
4. Identify the methods
5. Refine the classes, attributes and, methods

Three-Tier (Layer) Architecture

1. Presentation-Tier
2. Business-Tier
3. Database-Tier (also referred to as the data layer)

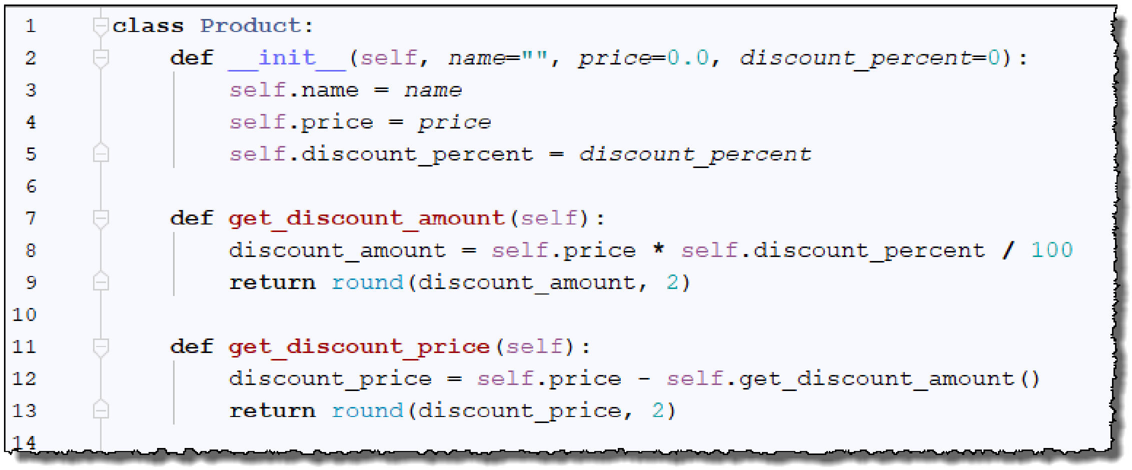
The Shopping Cart Program

In this application, we're going to create a shopping cart application where we utilize many of the oop skills and techniques we've covered so far. We'll break up the application into the three tiers (presentation, business, and data).

In this application, we'll have products that consist of three attributes (name, price, and discount percentage).

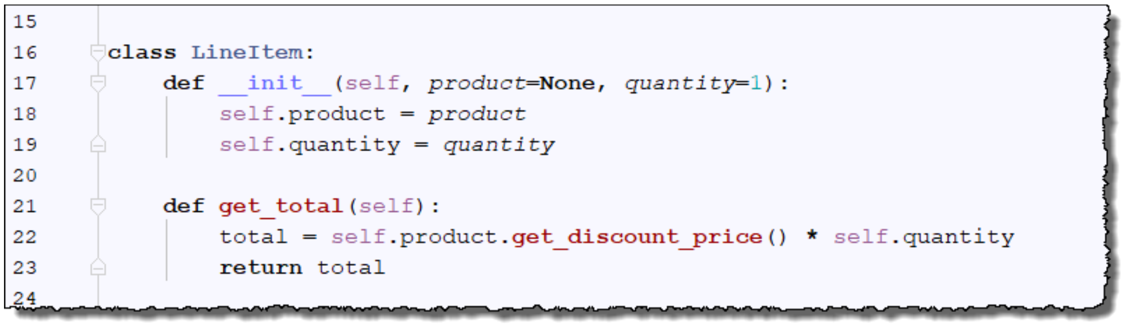
Creating the Business Tier

1. Create a new python file **business.py**.
2. Add a Product class as follows:



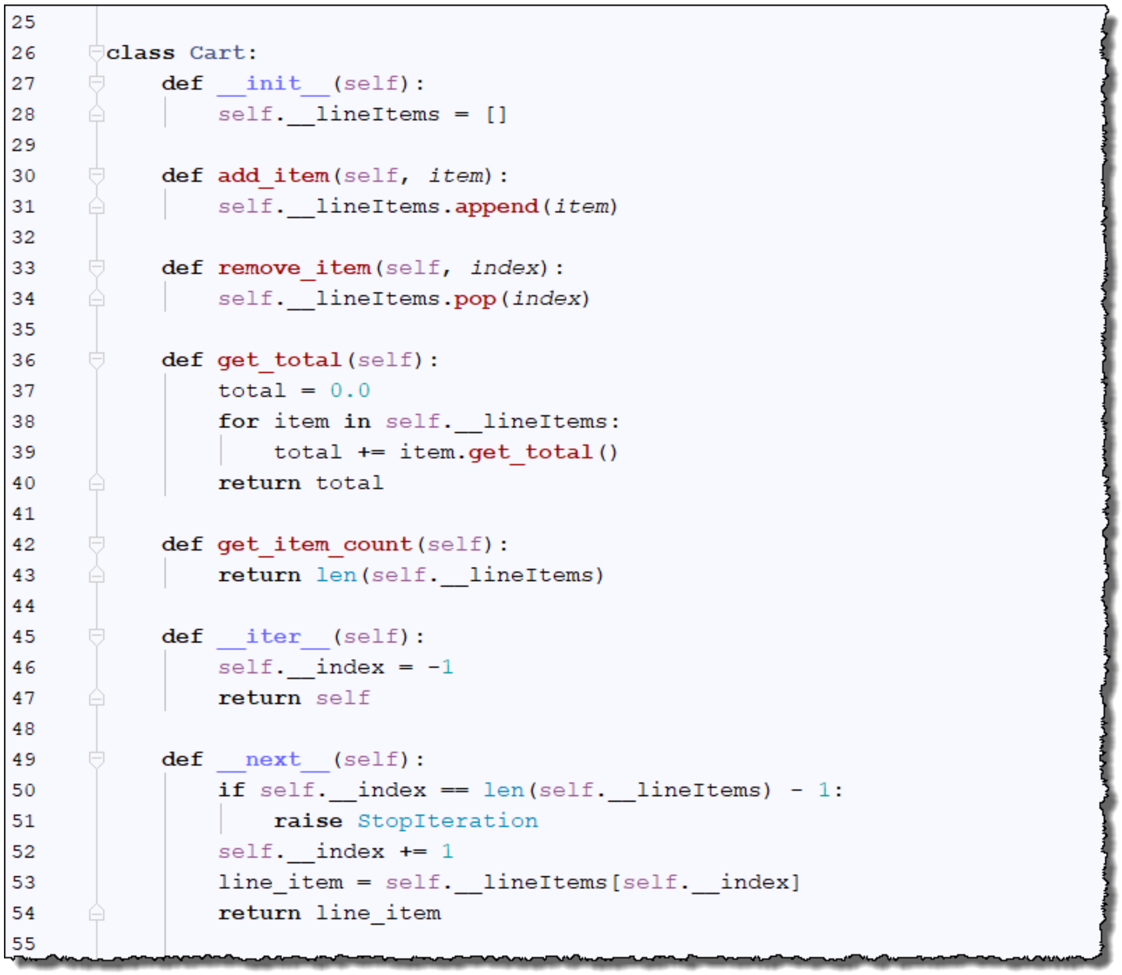
```python
class Product:
    def __init__(self, name="", price=0.0, discount_percent=0):
        self.name = name
        self.price = price
        self.discount_percent = discount_percent

    def get_discount_amount(self):
        discount_amount = self.price * self.discount_percent / 100
        return round(discount_amount, 2)

    def get_discount_price(self):
        discount_price = self.price - self.get_discount_amount()
        return round(discount_price, 2)

```

3. Add a LineItem class as follows:



```python

class LineItem:
    def __init__(self, product=None, quantity=1):
        self.product = product
        self.quantity = quantity

    def get_total(self):
        total = self.product.get_discount_price() * self.quantity
        return total

```

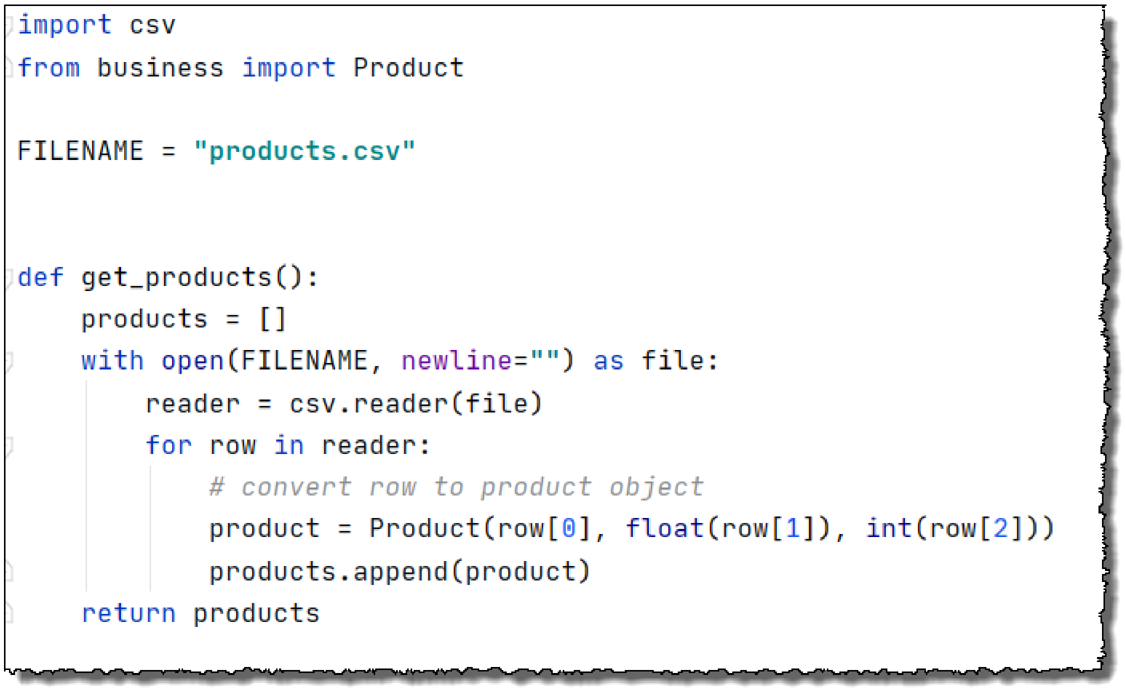
4. Add a Cart class as follows:



```
class Cart:
    def __init__(self):
        self.__lineItems = []

    def add_item(self, item):
        self.__lineItems.append(item)

    def remove_item(self, index):
        self.__lineItems.pop(index)

    def get_total(self):
        total = 0.0
        for item in self.__lineItems:
            total += item.get_total()
        return total

    def get_item_count(self):
        return len(self.__lineItems)

    def __iter__(self):
        self.__index = -1
        return self

    def __next__(self):
        if self.__index == len(self.__lineItems) - 1:
            raise StopIteration
        self.__index += 1
        line_item = self.__lineItems[self.__index]
        return line_item

```

Creating the Database Tier

5. Create a new python file **db.py**.
6. Code as follows:

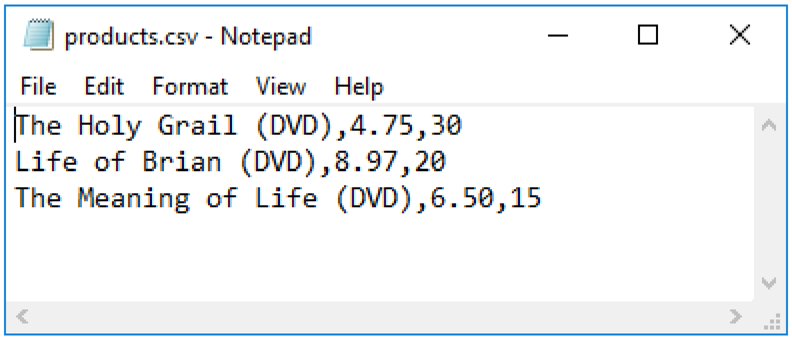


```python
import csv
from business import Product

FILENAME = "products.csv"


def get_products():
    products = []
    with open(FILENAME, newline="") as file:
        reader = csv.reader(file)
        for row in reader:
            # convert row to product object
            product = Product(row[0], float(row[1]), int(row[2]))
            products.append(product)
    return products
```

Creating the Database File

Note: If you did the Extra Credit above, skip to step 9.

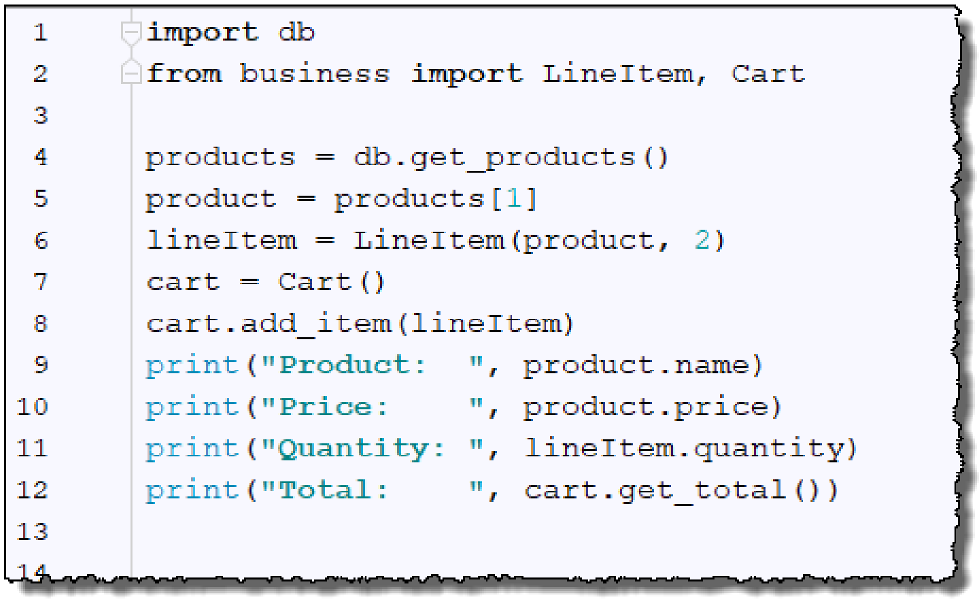
7. In the same folder as the python files, create a products.csv file
8. Populate as follows:



```
The Holy Grail (DVD),4.75,30
Life of Brian (DVD),8.97,20
The Meaning of Life (DVD),6.50,15
```

Creating a Testing Module

9. Create a new python file **test.py**.
10. Code as follows:

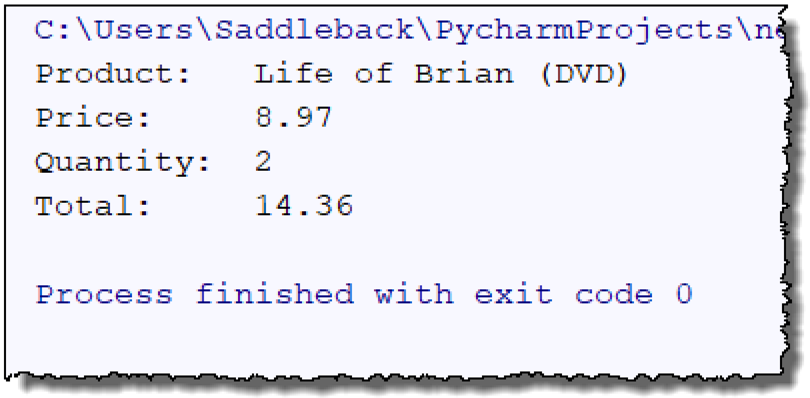


```python
import db
from business import LineItem, Cart

products = db.get_products()
product = products[1]
lineItem = LineItem(product, 2)
cart = Cart()
cart.add_item(lineItem)
print("Product:  ", product.name)
print("Price:    ", product.price)
print("Quantity: ", lineItem.quantity)
print("Total:    ", cart.get_total())
```

Testing the Business and Database Tiers

11. Run the test.py

Screen Capture #1 (2 points)



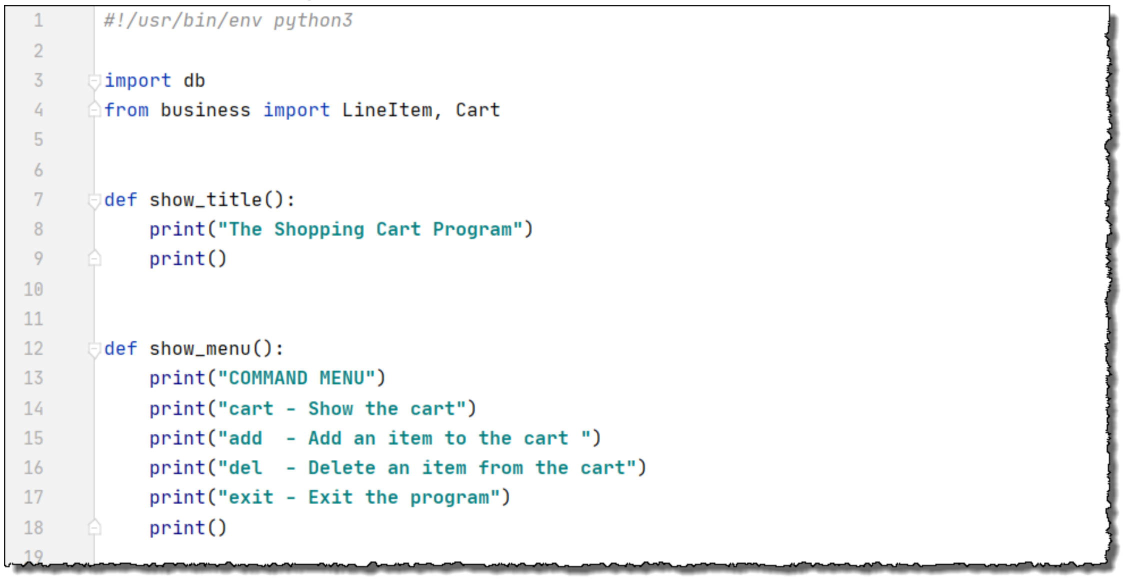
```
C:\Users\Saddleback\PycharmProjects\n
Product:   Life of Brian (DVD)
Price:     8.97
Quantity:  2
Total:     14.36

Process finished with exit code 0
```

Creating the Presentation Tier

12. Create a new python file **shopping_cart.py**.
13. The title and menu should look like:

```
The Shopping Cart Program

COMMAND MENU
cart - Show the cart
add  - Add an item to the cart
del  - Delete an item from the cart
exit - Exit the program
```

14. So we'll code the program title and menu as follows:

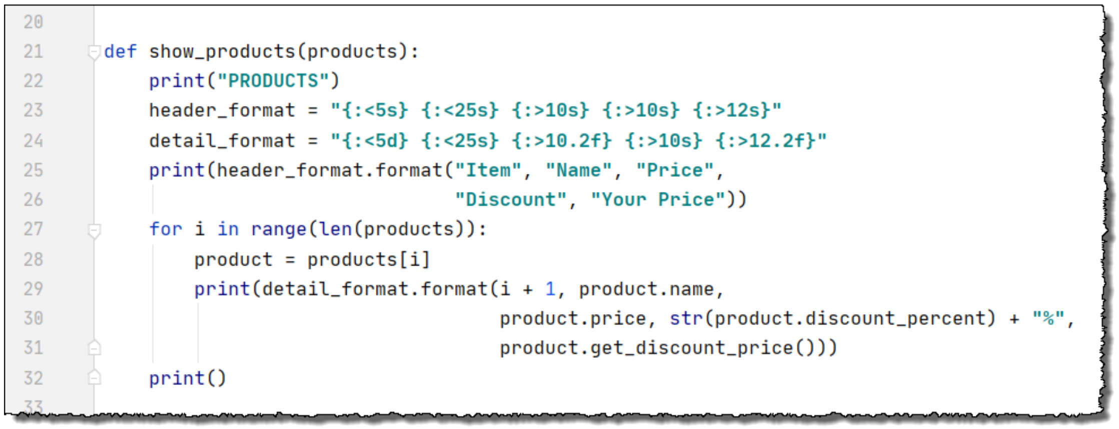


```python
#!/usr/bin/env python3

import db
from business import LineItem, Cart


def show_title():
    print("The Shopping Cart Program")
    print()


def show_menu():
    print("COMMAND MENU")
    print("cart - Show the cart")
    print("add  - Add an item to the cart ")
    print("del  - Delete an item from the cart")
    print("exit - Exit the program")
    print()
```

15. We want the product listing to appear as…

```
PRODUCTS
Item  Name                          Price   Discount    Your Price
1     The Holy Grail (DVD)           4.75        30%          3.75
2     Life of Brian (DVD)            8.97        20%          6.97
3     The Meaning of Life (DVD)      6.50        15%          5.50
```

16. …so to achieve, code as follows:

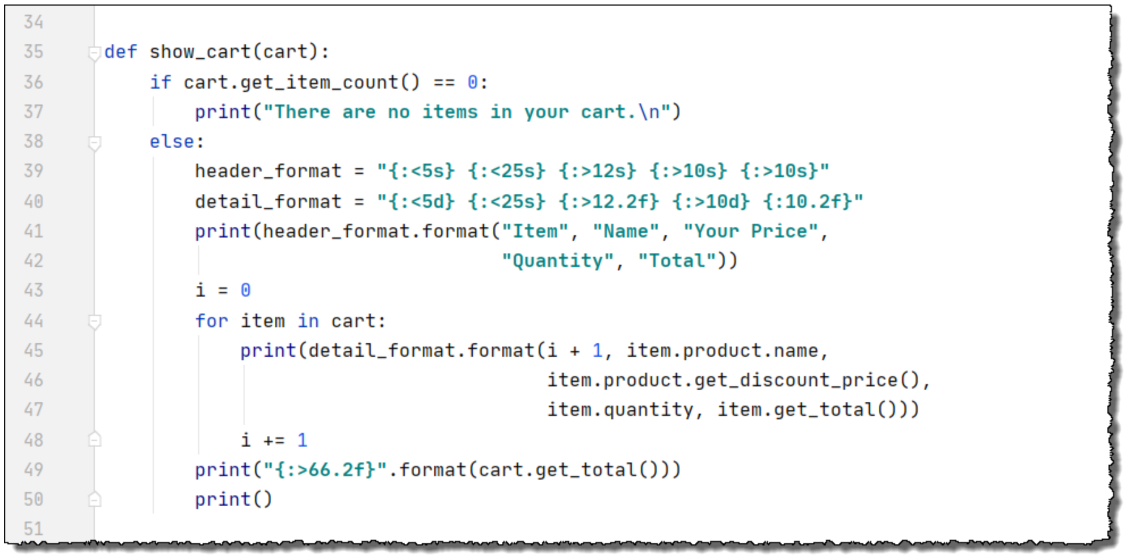


```python
def show_products(products):
    print("PRODUCTS")
    header_format = "{:<5s} {:<25s} {:>10s} {:>10s} {:>12s}"
    detail_format = "{:<5d} {:<25s} {:>10.2f} {:>10s} {:>12.2f}"
    print(header_format.format("Item", "Name", "Price",
                               "Discount", "Your Price"))
    for i in range(len(products)):
        product = products[i]
        print(detail_format.format(i + 1, product.name,
                                   product.price, str(product.discount_percent) + "%",
                                   product.get_discount_price()))
    print()
```

17. We want the product listing to appear as…

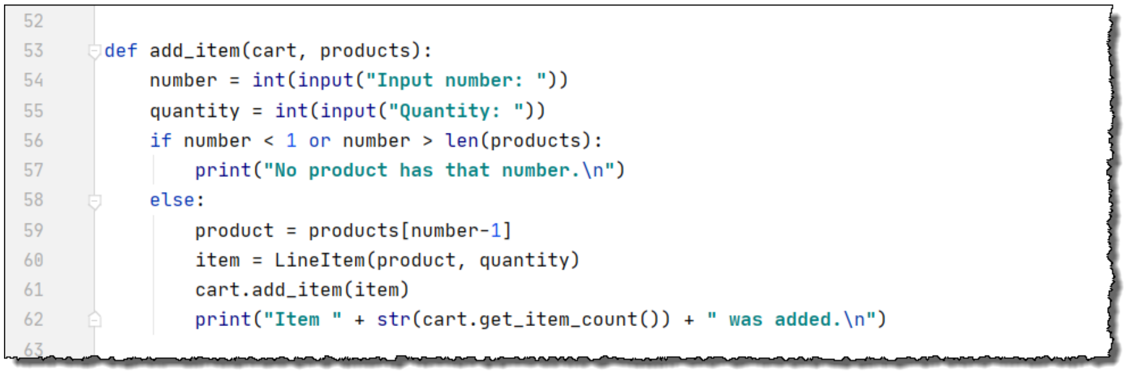
```
Command: cart
Item  Name                        Your Price   Quantity      Total
1     Life of Brian (DVD)               6.97          3      20.91
                                                             20.91
```

18. … so to achieve that, code as follows:



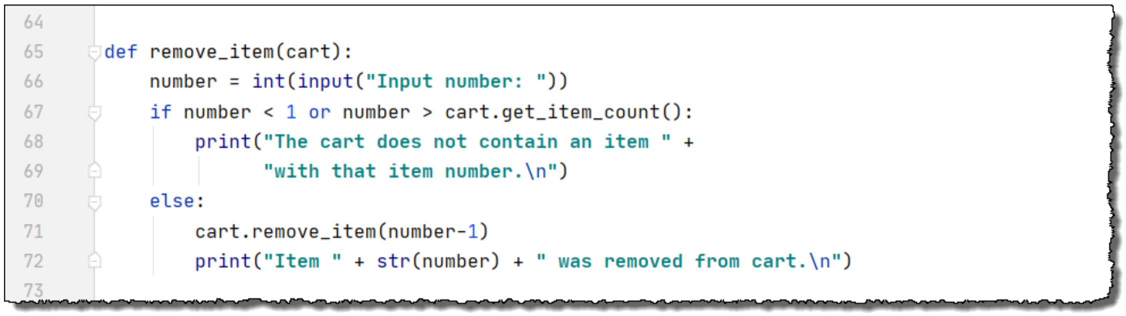
```python
def show_cart(cart):
    if cart.get_item_count() == 0:
        print("There are no items in your cart.\n")
    else:
        header_format = "{:<5s} {:<25s} {:>12s} {:>10s} {:>10s}"
        detail_format = "{:<5d} {:<25s} {:>12.2f} {:>10d} {:10.2f}"
        print(header_format.format("Item", "Name", "Your Price",
                                   "Quantity", "Total"))
        i = 0
        for item in cart:
            print(detail_format.format(i + 1, item.product.name,
                                       item.product.get_discount_price(),
                                       item.quantity, item.get_total()))
            i += 1
        print("{:>66.2f}".format(cart.get_total()))
        print()
```

19. We'll need to add the ability to items to the cart, code as follows:



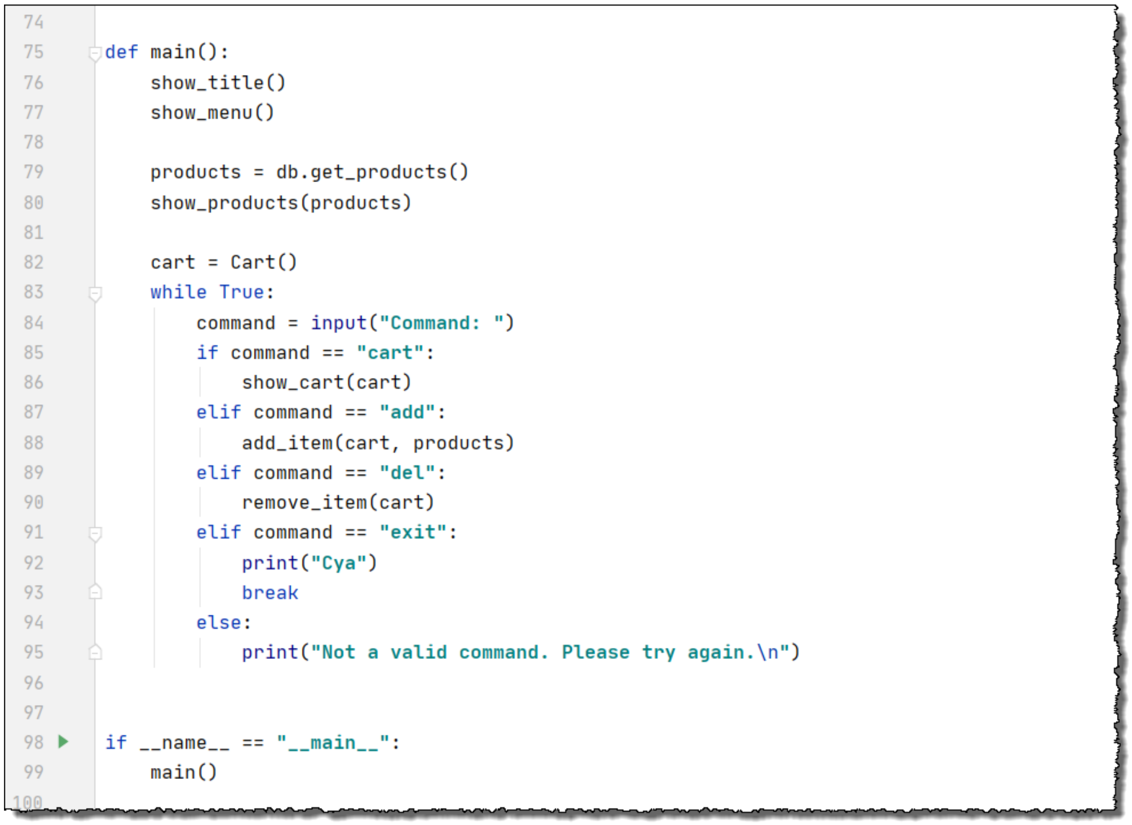
```python

def add_item(cart, products):
    number = int(input("Input number: "))
    quantity = int(input("Quantity: "))
    if number < 1 or number > len(products):
        print("No product has that number.\n")
    else:
        product = products[number-1]
        item = LineItem(product, quantity)
        cart.add_item(item)
        print("Item " + str(cart.get_item_count()) + " was added.\n")

```

20. We'll also need the ability to remove items from the cart, so code as follows:



```python

def remove_item(cart):
    number = int(input("Input number: "))
    if number < 1 or number > cart.get_item_count():
        print("The cart does not contain an item " +
              "with that item number.\n")
    else:
        cart.remove_item(number-1)
        print("Item " + str(number) + " was removed from cart.\n")

```

21. And finally, we'll need to add the main program flow, so code as follows:

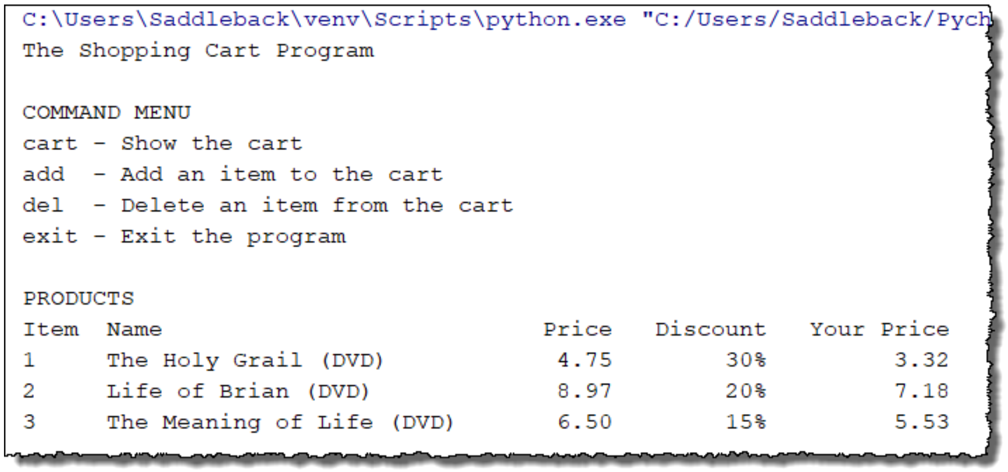


```python

def main():
    show_title()
    show_menu()

    products = db.get_products()
    show_products(products)

    cart = Cart()
    while True:
        command = input("Command: ")
        if command == "cart":
            show_cart(cart)
        elif command == "add":
            add_item(cart, products)
        elif command == "del":
            remove_item(cart)
        elif command == "exit":
            print("Cya")
            break
        else:
            print("Not a valid command. Please try again.\n")


if __name__ == "__main__":
    main()

```

Test

22. Run the application.
23. Display the title, header and product listing

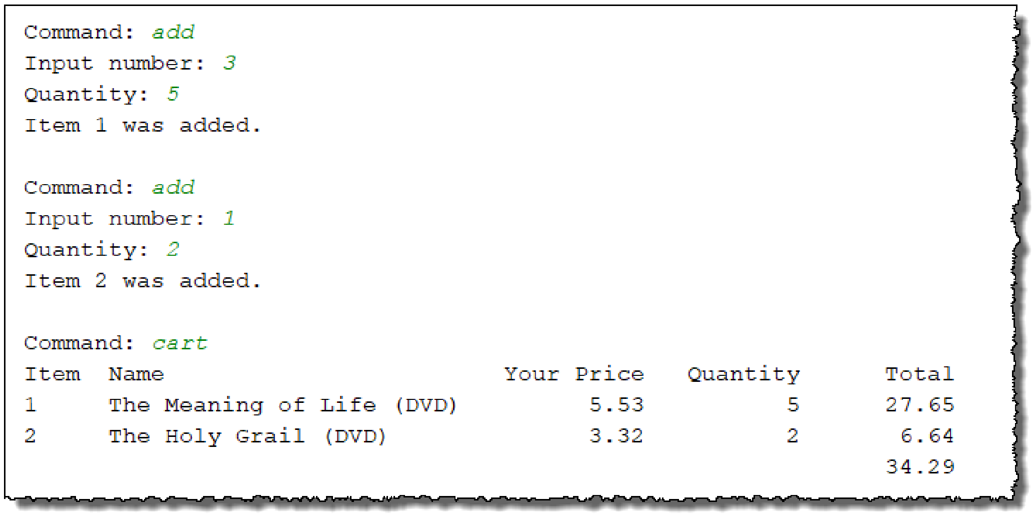
    Screen Capture #2 (1 point)



```
C:\Users\Saddleback\venv\Scripts\python.exe "C:/Users/Saddleback/Pych
The Shopping Cart Program

COMMAND MENU
cart - Show the cart
add  - Add an item to the cart
del  - Delete an item from the cart
exit - Exit the program

PRODUCTS
Item  Name                         Price   Discount   Your Price
1     The Holy Grail (DVD)          4.75        30%         3.32
2     Life of Brian (DVD)           8.97        20%         7.18
3     The Meaning of Life (DVD)     6.50        15%         5.53
```

24. Add 2 items as follows and display the items in the cart:

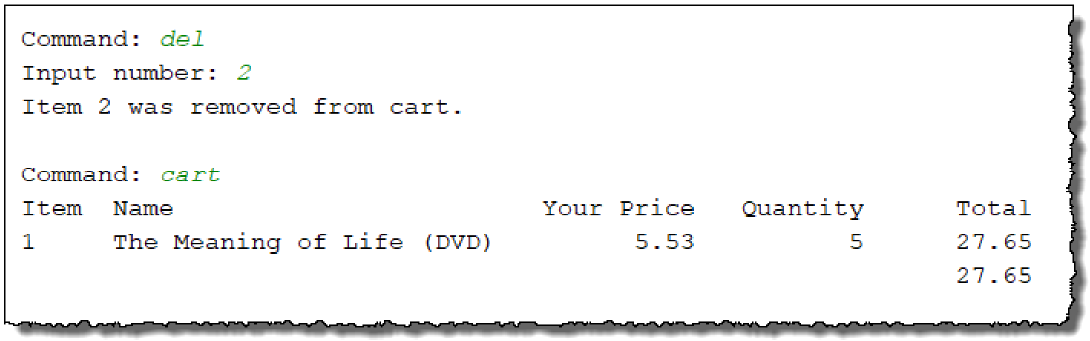
    Screen Capture #3 (2 point)



```
Command: add
Input number: 3
Quantity: 5
Item 1 was added.

Command: add
Input number: 1
Quantity: 2
Item 2 was added.

Command: cart
Item  Name                        Your Price   Quantity      Total
1     The Meaning of Life (DVD)         5.53          5      27.65
2     The Holy Grail (DVD)              3.32          2       6.64
                                                             34.29
```

25. Remove item 2 from the cart

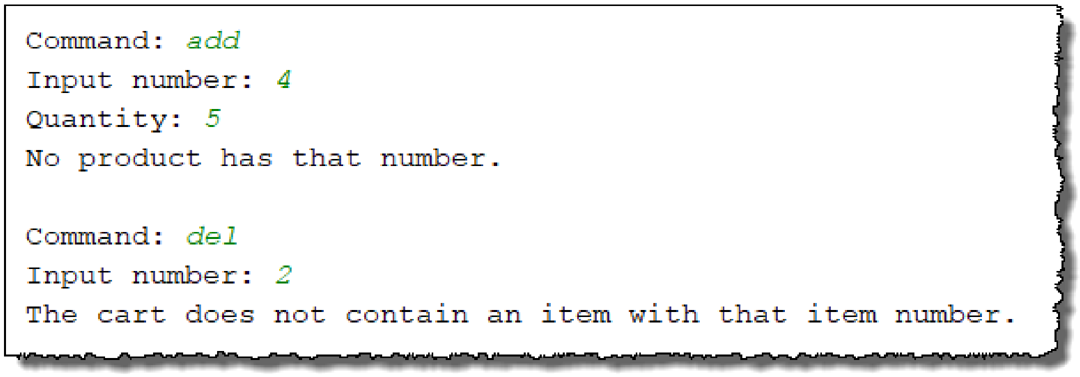
Screen Capture #4 (1 point)



```
Command: del
Input number: 2
Item 2 was removed from cart.

Command: cart
Item  Name                            Your Price   Quantity      Total
1     The Meaning of Life (DVD)             5.53          5      27.65
                                                                 27.65
```

26. Try and add an invalid product, then try and remove an invalid item from the cart

Screen Capture #5 (2 point)



```
Command: add
Input number: 4
Quantity: 5
No product has that number.

Command: del
Input number: 2
The cart does not contain an item with that item number.
```

Code Validation for business.py, db.py, test.py & the shopping_cart.py (12 points)