CSA0961 – JAVA

JF PROJECT 5

1. Open the inventory program that was created in Section 4: Creating an inventory Project.

2. Modify the ProductTester class.

a. Add a Scanner called in to the beginning of your main method.

b. Create local variables that will store values for the for each of the attributes of the Product class. Name the variables tempNumber, tempName, tempQty and tempPrice.

c. Ask the user to input values for the for each of the attributes of the Product class. Ask for the name, quantity, price and item number, store the values in temporary local variables that you have just created.

d. Use the values that were entered by the user to create the p1 object. This means that you will be using the constructor that takes 4 parameters instead of the default constructor.

3. You are going to get the user to provide you with values for p2.

a. Use the same local variables as before to get input from the user to create the p2 object. Copy and paste the code after the line that makes the p1 object.

b. Run the program and identify where an error has occurred.

c. The program doesn't appear to ask you for a value for name. This is because the last value entered was a numeric value and it has left some special characters in the input buffer. To clear the input buffer, add the following statement before you ask for any values for p2: in.nextLine(); This takes in any values stored in the buffer and discards them leaving an empty buffer.

d. Run the program now, it should be error free and display all the values including the user entered ones in the console.

e. Close the Scanner object when you are finished with it.

4. You want to be able to mark your products as active or discontinued. If a product is discontinued it means that the remaining stock will be the last of it, and no more orders are to be made. a. Add a Boolean instance field to your Product class called active that has a default value of true. b. Create getter/setter methods for this new field. c. Add the value of this new field to the toString() method so that the output matches the following:

Item Number : 1

Name : Greatest Hits

Quantity in stock:25

Price : 9.99

Product Status : true

5. When you run the code, you get back a printed value for active as either true or false. This is not user friendly and would be better if the output stated either Active (true) or Discontinued (false). Add a ternary operator in your toString() method to achieve this

6. Call the setter from the driver class and set the active value to false for the p6 object before you display the values to screen. Run and test your code.

7. Create a method in the Product class that will return the inventory value for each item. Use the product price multiplied by the quantity of stock to calculate the inventory value. Do not use any local variables in this method simply return the value in a single line of code.

8. Update the toString() method in the Product class to include a method call to the getInventoryValue() method that you have just created so that the output is as follows:

Item Number : 1

Name : Greatest Hits

Quantity in stock: 25

Price : 9.99

Stock Value : 249.75

Product Status : true

9. Save your project.

1. Modify the ProductTester Class

a.Add a Scanner

PROGRAM :

```
import java.util.Scanner;


public class ProductTester {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int tempNumber;
        String tempName;
        int tempQty;
        double tempPrice;
        System.out.print("Enter item number for Product 1: ");
        tempNumber = in.nextInt();
        in.nextLine();
        System.out.print("Enter name for Product 1: ");
        tempName = in.nextLine();
        System.out.print("Enter quantity for Product 1: ");
        tempQty = in.nextInt();
        System.out.print("Enter price for Product 1: ");
```

```
        tempPrice = in.nextDouble();
        Product p1 = new Product(tempNumber, tempName, tempQty, tempPrice);
        System.out.print("Enter item number for Product 2: ");
        tempNumber = in.nextInt();
        in.nextLine();
        System.out.print("Enter name for Product 2: ");
        tempName = in.nextLine();
        System.out.print("Enter quantity for Product 2: ");
        tempQty = in.nextInt();
        System.out.print("Enter price for Product 2: ");
        tempPrice = in.nextDouble();
        Product p2 = new Product(tempNumber, tempName, tempQty, tempPrice);
        in.close();
        System.out.println("\nProduct 1:");
        System.out.println(p1);
        System.out.println("\nProduct 2:");
        System.out.println(p2);
    }
}
```

2. Modifying the ‘Product ‘ Class

a. Add ‘active ‘ Field

PROGRAM :

```
public class Product {
    private int itemNumber;
    private String name;
    private int quantity;
    private double price;
    private boolean active = true; // default value
    public Product(int itemNumber, String name, int quantity, double price) {
        this.itemNumber = itemNumber;
        this.name = name;
        this.quantity = quantity;
```

```
        this.price = price;
    }
    public boolean isActive() {
        return active;
    }

    public void setActive(boolean active) {
        this.active = active;
    }
    public double getInventoryValue() {
        return price * quantity;
    }

    @Override
    public String toString() {
        return String.format(
            "Item Number      : %d\n" +
            "Name             : %s\n" +
            "Quantity in stock: %d\n" +
            "Price            : %.2f\n" +
            "Stock Value      : %.2f\n" +
            "Product Status   : %s",
            itemNumber,
            name,
            quantity,
            price,
            getInventoryValue(),
            active ? "Active" : "Discontinued"
        );
    }
}
```