JAVA FUNDAMENTALS SECTION-7 :

CREATING AN INVENTORY PROJECT

- B.RISHITHA

192324130

Topics:

- Modifying Programs
- Creating Static Methods
- Using parameters in a method
- Return a value from a method
- Adding methods(behaviours) to an existing class
- Implementing a user interface.

Problem Statement:

- Create an inventory program that can be used for a range of different products.

Code:

```
 import java. util.Scanner;

import java.util.InputMismatchException;

 class Product {

    private String name;

    private int inventory;

    private boolean active;


    public Product(String name, int inventory) {

        this.name = name;

        this.inventory = inventory;

        this.active = true;

    }


    public String getName() {

        return name;

    }
    public int getInventory() {
        return inventory;
    }

    public void addToInventory(int quantity) {
        inventory += quantity;
    }

    public void deductFromInventory(int quantity) {
        if (quantity <= inventory) {
            inventory -= quantity;
        } else {
            System.out.println("Not enough inventory to fulfill request.");
        }
    }

    public void setActive(boolean active) {
        this.active = active;
    }

    public boolean isActive() {
        return active;
    }
}



 class ProductTester {
    public static void main(String[] args) {
```

```
    Scanner in = new Scanner(System.in);
    int maxSize, menuChoice;

    maxSize = getNumProducts(in);
    if (maxSize == 0) {
        System.out.println("No products required!");
    } else {
        Product[] products = new Product[maxSize];
        addToInventory(products, in);
        do {
            menuChoice = getMenuOption(in);
            executeMenuChoice(menuChoice, products, in);
        } while (menuChoice != 4);
    }
}

public static void displayInventory(Product[] products) {
    for (int i = 0; i < products.length; i++) {
        System.out.println((i + 1) + ". " + products[i].getName() + " - " + products[i].getInventory());
    }
}

public static void addToInventory(Product[] products, Scanner in) {
    for (int i = 0; i < products.length; i++) {
        System.out.print("Enter name for product " + (i + 1) + ": ");
        String name = in.next();
        System.out.print("Enter initial inventory for product " + (i + 1) + ": ");
        int inventory = in.nextInt();
        products[i] = new Product(name, inventory);
    }
```

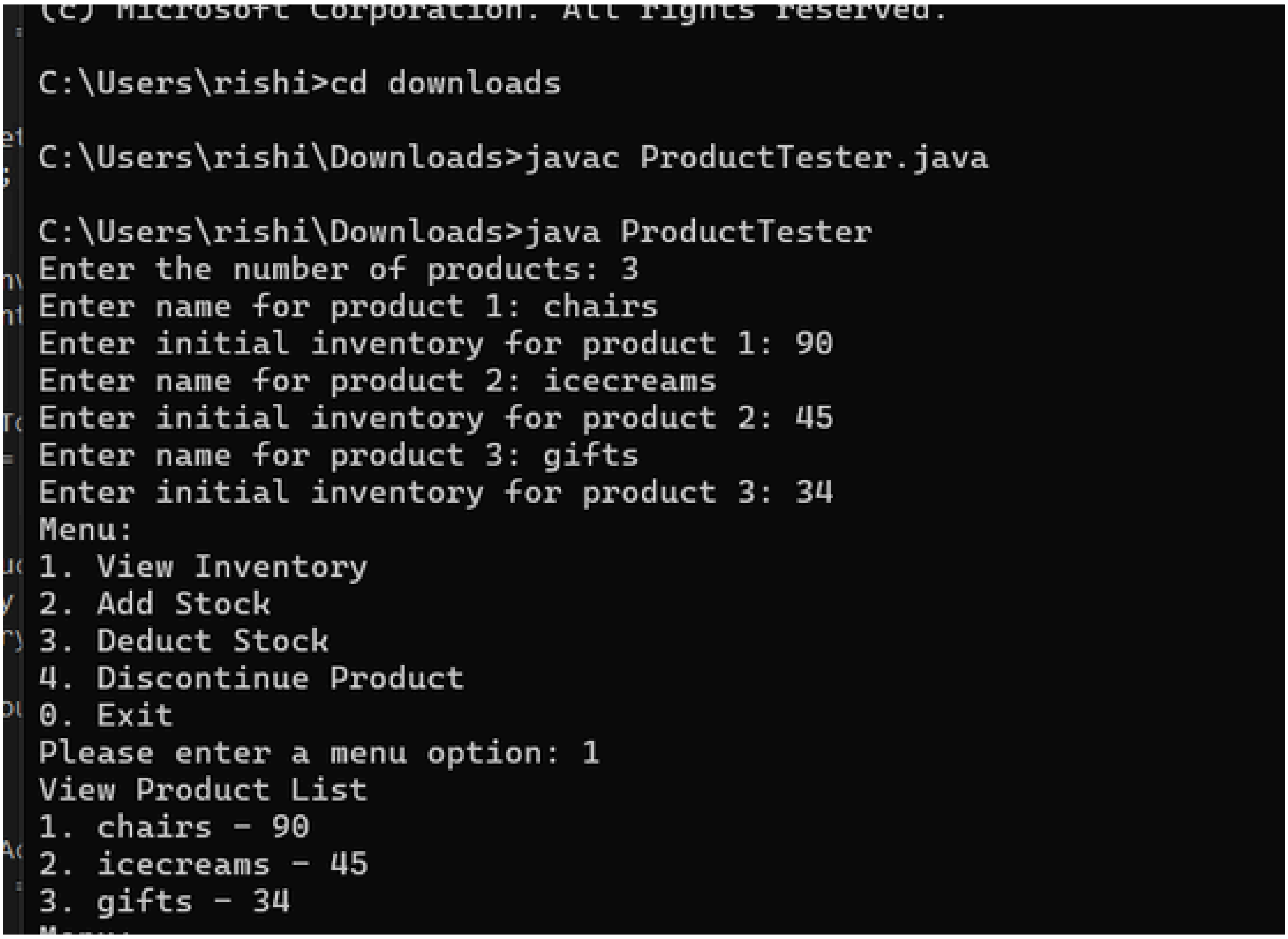
```java
}

public static int getMenuOption(Scanner in) {
   int menuChoice = -1;
   do {
      try {
         System.out.println("Menu:");
         System.out.println("1. View Inventory");
         System.out.println("2. Add Stock");
         System.out.println("3. Deduct Stock");
         System.out.println("4. Discontinue Product");
         System.out.println("0. Exit");
         System.out.print("Please enter a menu option: ");
         menuChoice = in.nextInt();
         if (menuChoice < 0 || menuChoice > 4) {
            System.out.println("Invalid choice. Please choose again.");
         }
      } catch (InputMismatchException e) {
         System.out.println("Incorrect data type entered! Please enter a valid integer.");
         in.next();
      } catch (Exception e) {
         System.out.println("An error occurred: " + e.getMessage());
         in.next();
      }
   } while (menuChoice < 0 || menuChoice > 4);
   return menuChoice;
}

public static int getProductNumber(Product[] products, Scanner in) {
   int productChoice = -1;
```

```
    do {
        try {
            System.out.println("Please enter the product number:");
            for (int i = 0; i < products.length; i++) {
                System.out.println((i + 1) + ". " + products[i].getName());
            }
            productChoice = in.nextInt();
            if (productChoice < 1 || productChoice > products.length) {
                System.out.println("Incorrect Value entered. Please enter a valid product number.");
            }
        } catch (InputMismatchException e) {
            System.out.println("Incorrect data type entered! Please enter a valid integer.");
            in.next();
        } catch (Exception e) {
            System.out.println("An error occurred: " + e.getMessage());
            in.next();
        }
    } while (productChoice < 1 || productChoice > products.length);
    return productChoice - 1;
}


public static void addInventory(Product[] products, Scanner in) {
    int productChoice = getProductNumber(products, in);
    int updateValue=0;
    do {
        try {
            System.out.print("How many products do you want to add? ");
            updateValue = in.nextInt();
            if (updateValue < 0) {
                System.out.println("Incorrect Value entered. Please enter a positive integer.");
            } else {
                products[productChoice].addToInventory(updateValue);
                            }
        } catch (InputMismatchException e) {
            System.out.println("Incorrect data type entered! Please enter a valid integer.");
            in.next();
        } catch (Exception e) {
            System.out.println("An error occurred: " + e.getMessage());
            in.next();
        }
    } while (updateValue < 0);
}

public static void deductInventory(Product[] products, Scanner in) {
    int productChoice = getProductNumber(products, in);
    int updateValue=0;
    do {
        try {
            System.out.print("How many products do you want to deduct? ");
            updateValue = in.nextInt();
            if (updateValue < 0) {
                System.out.println("Incorrect Value entered. Please enter a positive integer.");
            } else if (updateValue > products[productChoice].getInventory()) {
                System.out.println("Not enough inventory to fulfill request.");
            } else {
                products[productChoice].deductFromInventory(updateValue);
            }
        } catch (InputMismatchException e) {
            System.out.println("Incorrect data type entered! Please enter a valid integer.");
            in.next();
```

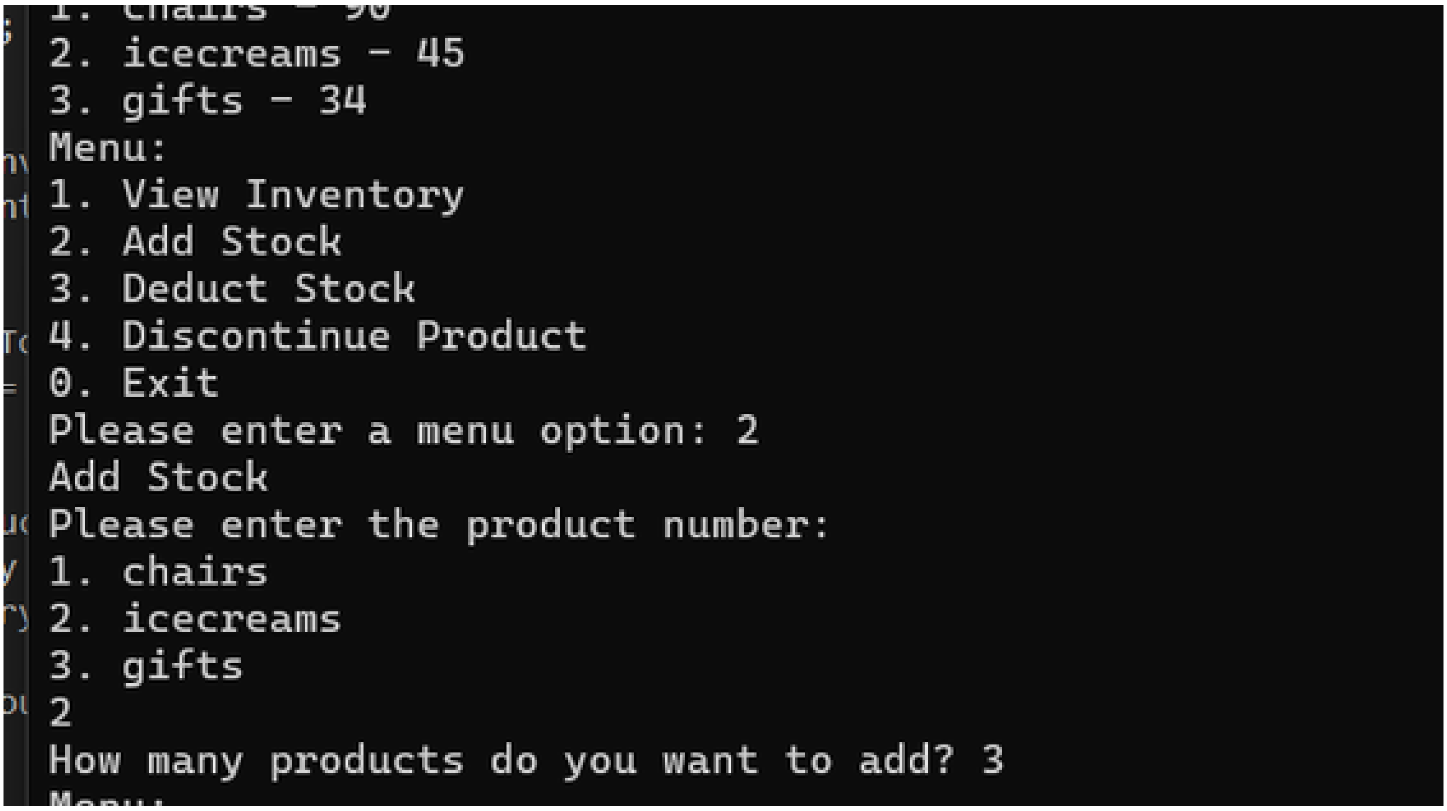
```
      } catch (Exception e) {
        System.out.println("An error occurred: " + e.getMessage());
        in.next();
      }
    } while (updateValue < 0 || updateValue > products[productChoice].getInventory());
  }

  public static void discontinueInventory(Product[] products, Scanner in) {
    int productChoice = getProductNumber(products, in);
    products[productChoice].setActive(false);
  }

  public static void executeMenuChoice(int menuChoice, Product[] products, Scanner in) {
    switch (menuChoice) {
      case 1:
        System.out.println("View Product List");
        displayInventory(products);
        break;
      case 2:
        System.out.println("Add Stock");
        addInventory(products, in);
        break;
      case 3:
        System.out.println("Deduct Stock");
        deductInventory(products, in);
        break;
      case 4:
        System.out.println("Discontinue Stock");
        discontinueInventory(products, in);
        break;
```

```
            default:
                System.out.println("Invalid choice. Please choose again.");
        }
    }

    public static int getNumProducts(Scanner in) {
        int maxSize = -1;
        do {
            try {
                System.out.print("Enter the number of products: ");
                maxSize = in.nextInt();
                if (maxSize < 0) {
                    System.out.println("Incorrect Value entered. Please enter a positive integer.");
                }
            } catch (InputMismatchException e) {
                System.out.println("Incorrect data type entered! Please enter a valid integer.");
                in.next();
            } catch (Exception e) {
                System.out.println("An error occurred: " + e.getMessage());
                in.next();
            }
        } while (maxSize < 0);
        return maxSize;
    }
}
```

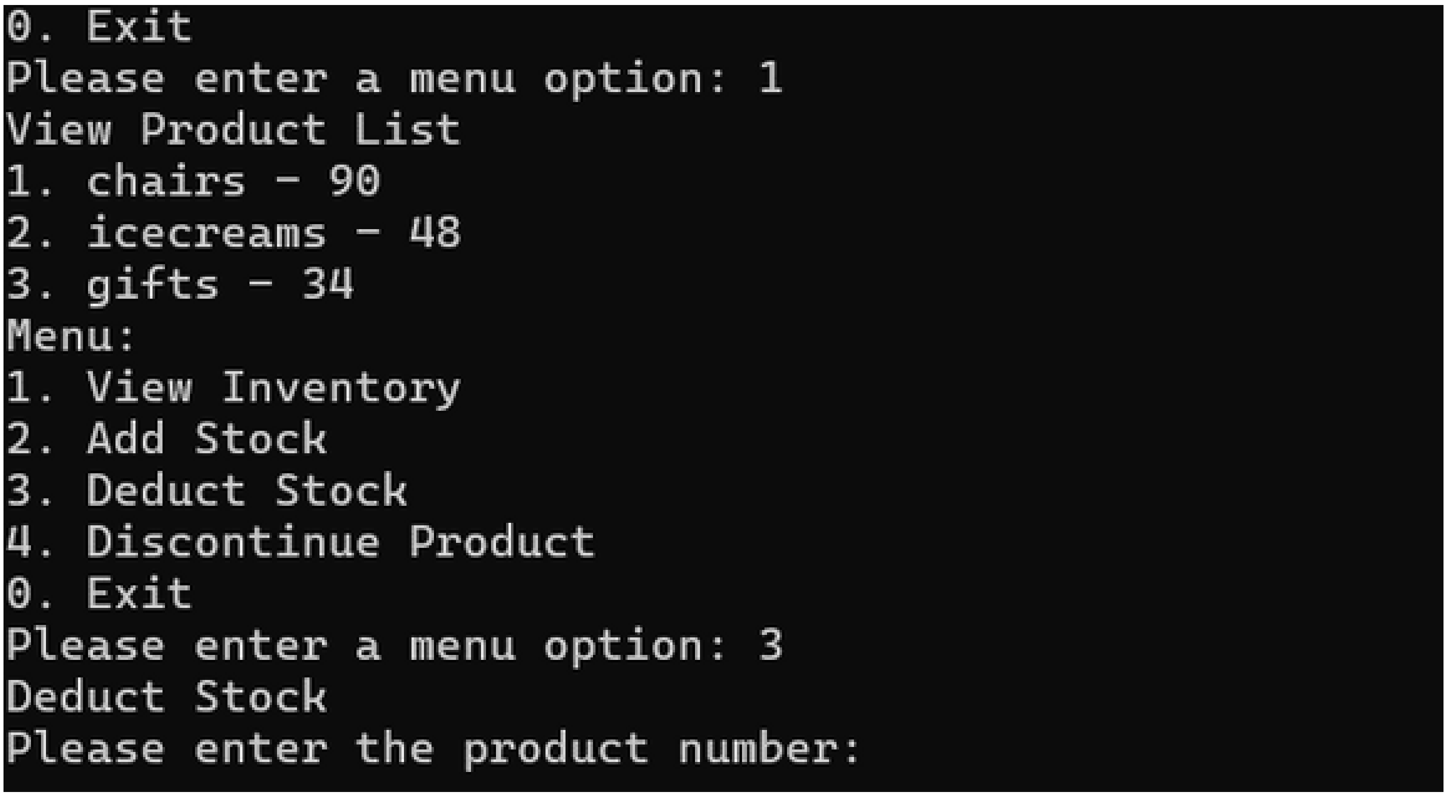
Output:



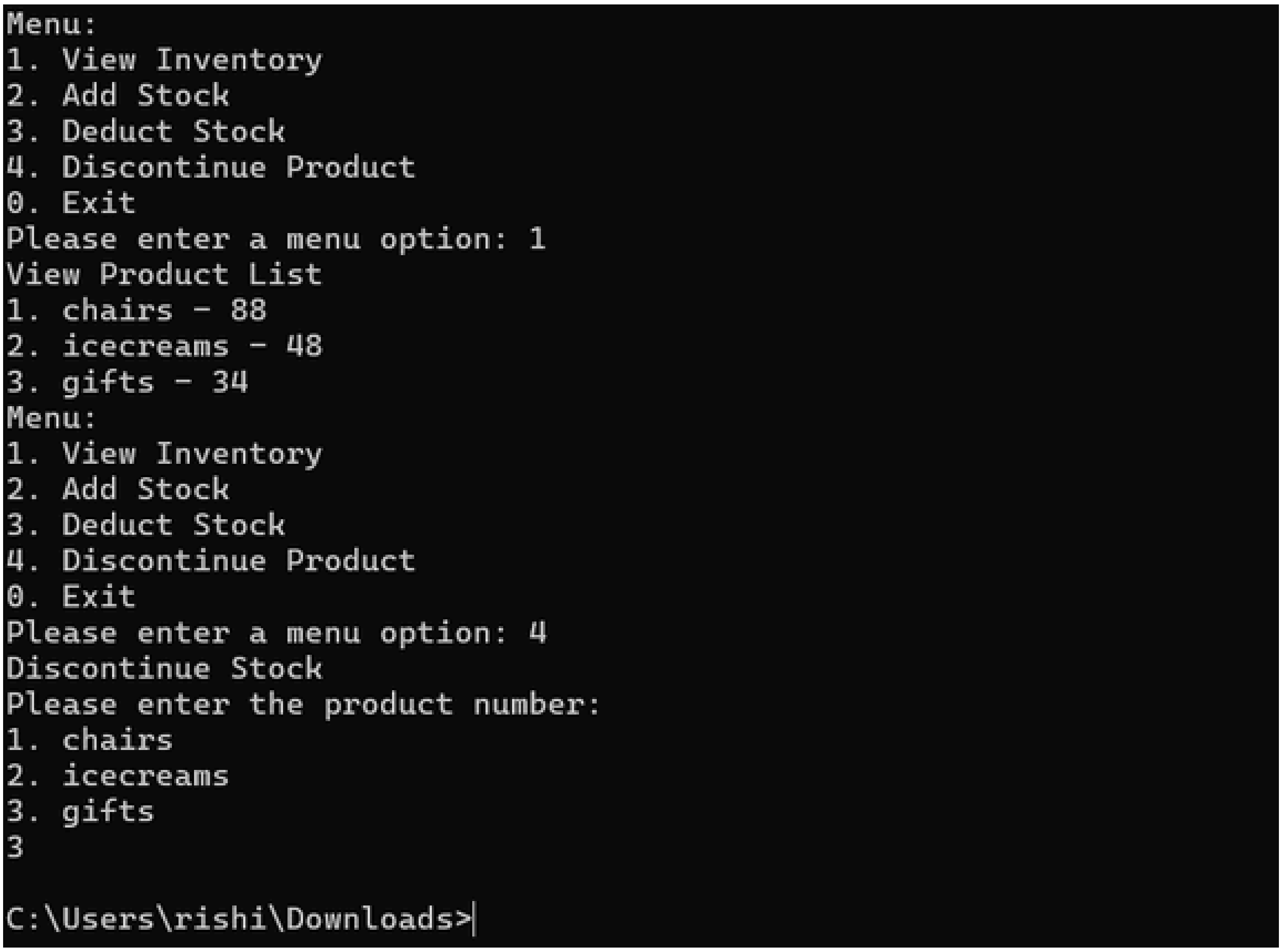
```
(c) Microsoft Corporation. All rights reserved.

C:\Users\rishi>cd downloads

C:\Users\rishi\Downloads>javac ProductTester.java

C:\Users\rishi\Downloads>java ProductTester
Enter the number of products: 3
Enter name for product 1: chairs
Enter initial inventory for product 1: 90
Enter name for product 2: icecreams
Enter initial inventory for product 2: 45
Enter name for product 3: gifts
Enter initial inventory for product 3: 34
Menu:
1. View Inventory
2. Add Stock
3. Deduct Stock
4. Discontinue Product
0. Exit
Please enter a menu option: 1
View Product List
1. chairs - 90
2. icecreams - 45
3. gifts - 34
```



```
1. chairs - 90
2. icecreams - 45
3. gifts - 34
Menu:
1. View Inventory
2. Add Stock
3. Deduct Stock
4. Discontinue Product
0. Exit
Please enter a menu option: 2
Add Stock
Please enter the product number:
1. chairs
2. icecreams
3. gifts
2
How many products do you want to add? 3
```



```
0. Exit
Please enter a menu option: 1
View Product List
1. chairs - 90
2. icecreams - 48
3. gifts - 34
Menu:
1. View Inventory
2. Add Stock
3. Deduct Stock
4. Discontinue Product
0. Exit
Please enter a menu option: 3
Deduct Stock
Please enter the product number:
```



```
Menu:
1. View Inventory
2. Add Stock
3. Deduct Stock
4. Discontinue Product
0. Exit
Please enter a menu option: 1
View Product List
1. chairs - 88
2. icecreams - 48
3. gifts - 34
Menu:
1. View Inventory
2. Add Stock
3. Deduct Stock
4. Discontinue Product
0. Exit
Please enter a menu option: 4
Discontinue Stock
Please enter the product number:
1. chairs
2. icecreams
3. gifts
3

C:\Users\rishi\Downloads>
```