SREE RUTHIN

192324112

PROJECT 6

Step 1: Open the inventory program that was updated in Section 5: Creating an inventory Project.

In this step, we are continuing to build upon the inventory program that we started in Section 5. We will be adding new features to the program to make it more robust and user-friendly.

Step 2: Ask the user to enter the number of products they wish to add.

In this step, we need to ask the user to enter the number of products they want to add to the inventory. We will use a do-while loop to ensure that the user enters a valid positive integer. If the user enters a value less than zero, we will display an error message and prompt the user to enter a new value.

Here's the code for this step:

```
int maxsize = -1;

do {

    try {

        System.out.println("Enter the number of products you would like to add. Enter 0 (zero) if you do not wish to add products.");

        maxsize = scanner.nextInt();

        if (maxsize < 0) {

            System.out.println("Incorrect Value entered");

        }

    } catch (InputMismatchException e) {

        System.out.println("Incorrect data type entered!");

        scanner.next(); // clear the input buffer

    }
```

```
} while (maxsize < 0);
```

Step 3: Add error handling to deal with run-time errors in your code.

In this step, we need to add error handling to deal with run-time errors that may occur when the user enters invalid input. We will use a try-catch block to catch InputMismatchException, which is thrown when the user enters a non-numeric valu e.

Here's the code for this step:

```
try {
    maxsize = scanner.nextInt();
} catch (InputMismatchException e) {
    System.out.println("Incorrect data type entered!");
    scanner.next(); // clear the input buffer
}
```

Step 4: Modify the ProductTester class to handle multiple products using a single dimensional array.

In this step, we need to modify the ProductTester class to handle multiple products using a single dimensional array. If the user enters a value greater than zero, we w ll create an array of Product objects with the specified size.

Here's the code for this step:

```
if (maxsize == 0) {
    System.out.println("No products required!");
} else {
    Product[] products = new Product[maxsize];
    // ...
}
```

Step 5: Populate the array, getting the values from the user for each field in a product object.

In this step, we need to populate the array with user input. We will use a for loop to iterate through the array and prompt the user to enter the values for each field in a product object.

Here's the code for this step:

```
for (int i = 0; i < maxsize; i++) {
    scanner.nextLine(); // clear the input buffer
    System.out.println("Enter product name:");
    String name = scanner.nextLine();
    System.out.println("Enter product quantity:");
    int quantity = scanner.nextInt();
    scanner.nextLine(); // clear the input buffer
    System.out.println("Enter product price:");
    double price = scanner.nextDouble();
    scanner.nextLine(); // clear the input buffer
    System.out.println("Enter product item number:");
    int itemNumber = scanner.nextInt();
    scanner.nextLine(); // clear the input buffer


    products[i] = new Product(name, quantity, price, itemNumber);
}
```

Step 6: Use a for each loop to display the information for each individual product in the products array.

In this step, we need to display the information for each individual product in the products array. We will use a for-each loop to iterate through the array and print out the values for each product.

Here's the code for this step:

```
for (Product product : products) {
    System.out.println("Product Name: " + product.getName());
    System.out.println("Product Quantity: " + product.getQuantity());
    System.out.println("Product Price: " + product.getPrice());
```

```
    System.out.println("Product Item Number: " + product.getItemNumber());

    System.out.println();

}
```

Step 7: Remove any unnecessary code that's not used in this exercise.

In this step, we need to remove any unnecessary code that's not used in this exercise. We should review the code and remove any redundant or unused code to make the program more efficient.

CODE:

```
import java.util.InputMismatchException;

import java.util.Scanner;



public class ProductTester {



    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);



        int maxsize = -1;



        do {

            try {

                System.out.println("Enter the number of products you would like to add. Enter 0 (zero) if you do not wish to add products.");

                maxsize = scanner.nextInt();

                if (maxsize < 0) {

                    System.out.println("Incorrect Value entered");

                }

            } catch (InputMismatchException e) {
```

```
            System.out.println("Incorrect data type entered!");
            scanner.next(); // clear the input buffer
        }
    } while (maxsize < 0);

    if (maxsize == 0) {
        System.out.println("No products required!");
    } else {
        Product[] products = new Product[maxsize];

        for (int i = 0; i < maxsize; i++) {
            scanner.nextLine(); // clear the input buffer
            System.out.println("Enter product name:");
            String name = scanner.nextLine();
            System.out.println("Enter product quantity:");
            int quantity = scanner.nextInt();
            scanner.nextLine(); // clear the input buffer
            System.out.println("Enter product price:");
            double price = scanner.nextDouble();
            scanner.nextLine(); // clear the input buffer
            System.out.println("Enter product item number:");
            int itemNumber = scanner.nextInt();
            scanner.nextLine(); // clear the input buffer

            products[i] = new Product(name, quantity, price, itemNumber);
```

```
            }

            for (Product product : products) {
                System.out.println("Product Name: " + product.getName());
                System.out.println("Product Quantity: " + product.getQuantity());
                System.out.println("Product Price: " + product.getPrice());
                System.out.println("Product Item Number: " + product.getItemNumber());
                System.out.println();
            }
        }
    }
}
public class Product {
    private String name;
    private int quantity;
    private double price;
    private int itemNumber;

    public Product(String name, int quantity, double price, int itemNumber) {
        this.name = name;
        this.quantity = quantity;
        this.price = price;
        this.itemNumber = itemNumber;
    }
```

```
    public String getName() {
        return name;
    }

    public int getQuantity() {
        return quantity;
    }

    public double getPrice() {
        return price;
    }

    public int getItemNumber() {
        return itemNumber;
    }
}
```

```
Enter the number of products you would like to add. Enter 0 (zero) if you
3
Enter product name:
Product1
Enter product quantity:
10
Enter product price:
10.99
Enter product item number:
1234
Enter product name:
Product2
Enter product quantity:
20
Enter product price:
20.99
Enter product item number:
5678
Enter product name:
Product3
Enter product quantity:
30
Enter product price:
30.99
Enter product item number:
9012

9012

Product Name: Product1
Product Quantity: 10
Product Price: 10.99
Product Item Number: 1234

Product Name: Product2
Product Quantity: 20
Product Price: 20.99
Product Item Number: 5678

Product Name: Product3
Product Quantity: 30
Product Price: 30.99
Product Item Number: 9012
```