CSA-0963 JAVA PROGRAMING FOR SYSTEM INTERFACES

SECTION 4: CREATING AN INVENTORY PROJECT.

B .Mahesh babu

192311189

Project-1.

PROGRAM:

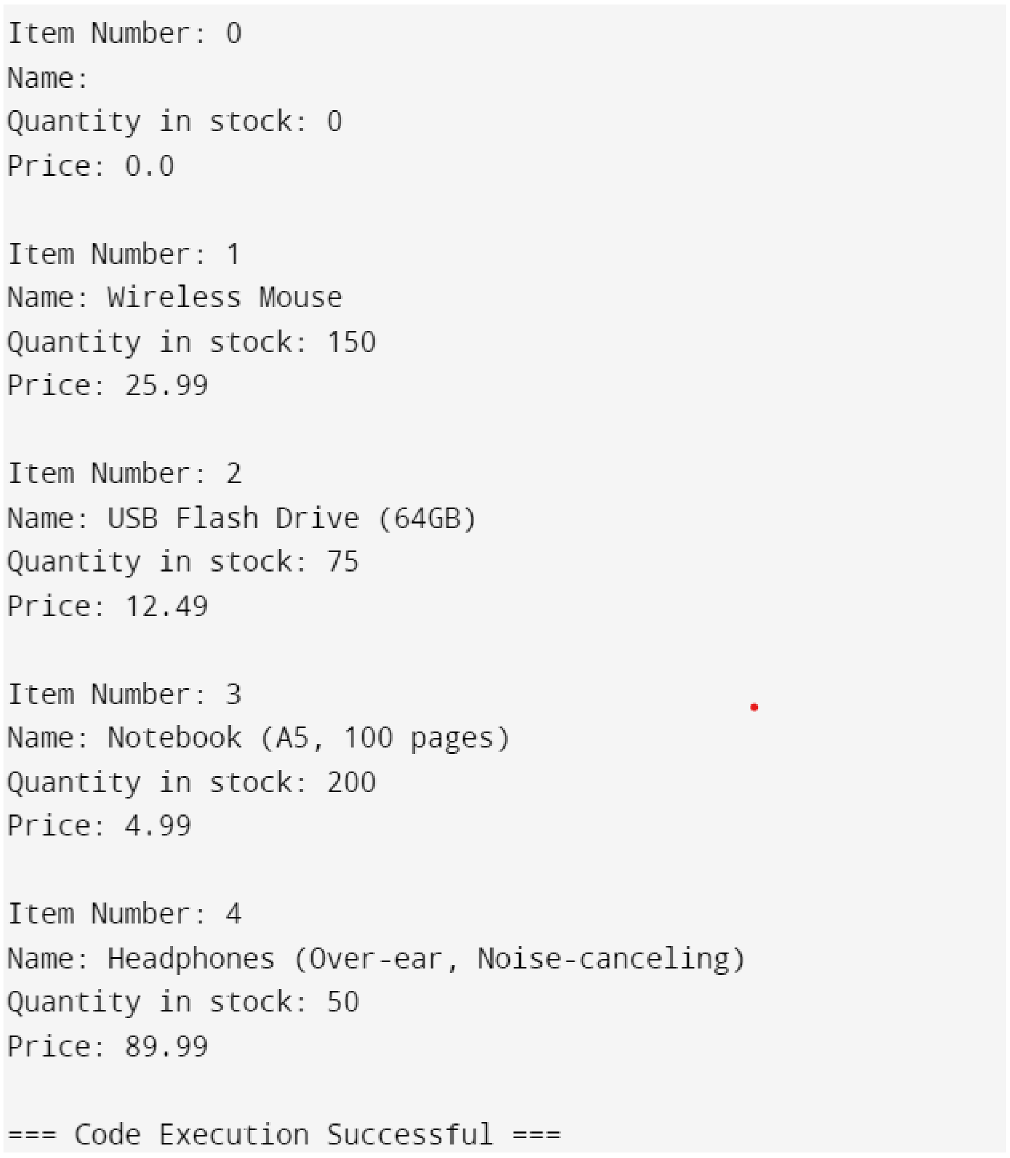
```
public class Product {
  // Instance field declarations
  private int itemNumber;
  private String name;
  private int numberOfUnitsInStock;
  private double price;




  public Product() {
    this.itemNumber = 0;
    this.name = "";
    this.numberOfUnitsInStock = 0;
    this.price = 0.0;
  }




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

```
   this.name = name;

   this.numberOfUnitsInStock = qty;

   this.price = price;

}


// Getter for itemNumber

// Returns the item number of the product

public int getItemNumber() {

   return itemNumber;

}


// Setter for itemNumber

// Sets the item number of the product

public void setItemNumber(int itemNumber) {

   this.itemNumber = itemNumber;

}





public String getName() {

   return name;

}


// Setter for name

// Sets the name of the product

public void setName(String name) {

   this.name = name;

}
```

```
// Returns the quantity of the product in stock
public int getNumberOfUnitsInStock() {
  return numberOfUnitsInStock;
}

// Setter for numberOfUnitsInStock
// Sets the quantity of the product in stock
public void setNumberOfUnitsInStock(int numberOfUnitsInStock) {
  this.numberOfUnitsInStock = numberOfUnitsInStock;
}

// Getter for price
// Returns the price of the product
public double getPrice() {
  return price;
}

// Setter for price
// Sets the price of the product
public void setPrice(double price) {
  this.price = price;
}

// Overrides the toString method to provide product details
@Override
public String toString() {
        return "Item Number: " + itemNumber +
            "\nName: " + name +
            "\nQuantity in stock: " + numberOfUnitsInStock +
            "\nPrice: " + price;
    }
}
// ProductTester.java
public class ProductTester {
    public static void main(String[] args) {
        // Creating Product objects
        Product product1 = new Product();  // Default constructor
        Product product2 = new Product();  // Default constructor
        Product product3 = new Product(1, "Wireless Mouse", 150, 25.99);
        Product product4 = new Product(2, "USB Flash Drive (64GB)", 75, 12.49);
        Product product5 = new Product(3, "Notebook (A5, 100 pages)", 200, 4.99);
        Product product6 = new Product(4, "Headphones (Over-ear, Noise-canceling)", 50,
89.99);

        // Displaying details of each product to the console
        System.out.println(product1.toString());
        System.out.println();
        System.out.println(product2.toString());
        System.out.println();
        System.out.println(product3.toString());
        System.out.println();
        System.out.println(product4.toString());
        System.out.println();
        System.out.println(product5.toString());
```

```
      System.out.println();

      System.out.println(product6.toString());

    }

}
```

Output:



```
Item Number: 0
Name:
Quantity in stock: 0
Price: 0.0

Item Number: 1
Name: Wireless Mouse
Quantity in stock: 150
Price: 25.99

Item Number: 2
Name: USB Flash Drive (64GB)
Quantity in stock: 75
Price: 12.49

Item Number: 3
Name: Notebook (A5, 100 pages)
Quantity in stock: 200
Price: 4.99

Item Number: 4
Name: Headphones (Over-ear, Noise-canceling)
Quantity in stock: 50
Price: 89.99

=== Code Execution Successful ===
```

Java Fundamentals

Section 5: Creating an Inventory Project

PROGRAM:

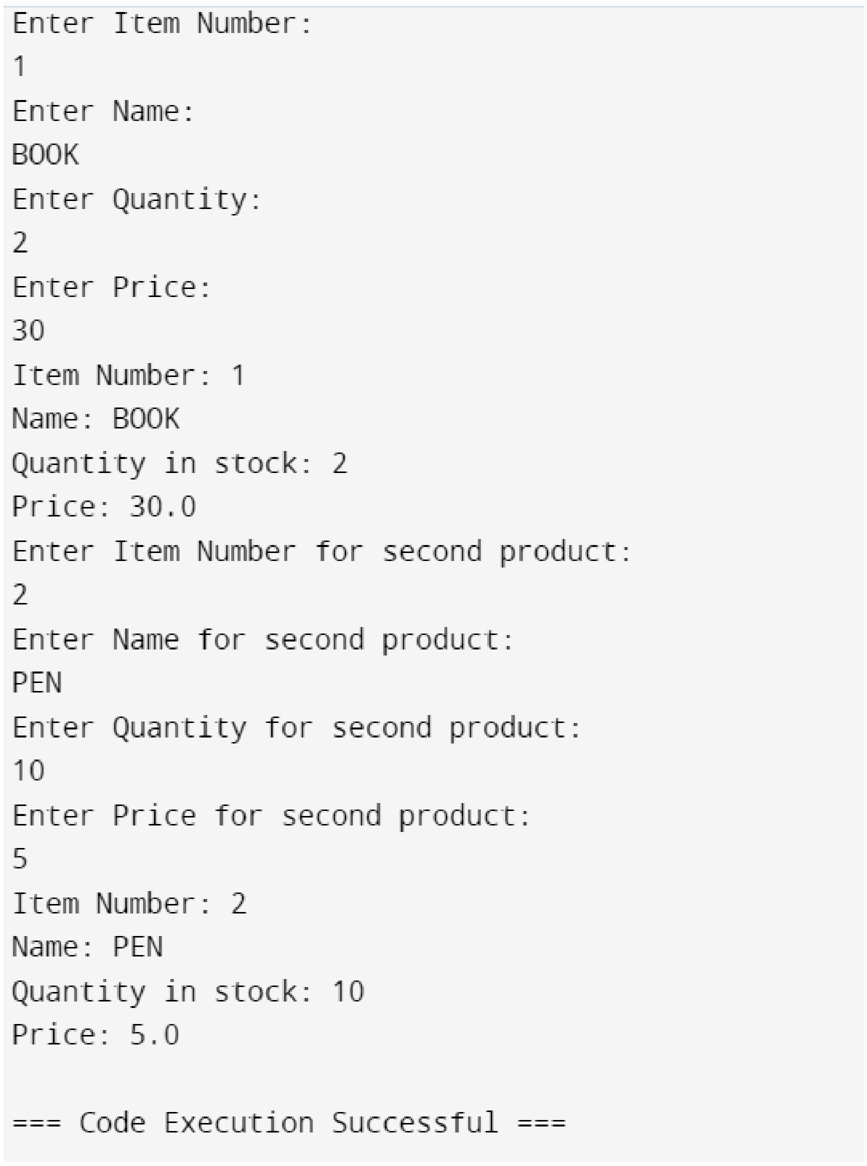
```
public class Product {
```

```
private int itemNumber;
private String name;
private int qty;
private double price;
private boolean active = true; // Default value is true

// Constructor with parameters
public Product(int itemNumber, String name, int qty, double price) {
    this.itemNumber = itemNumber;
    this.name = name;
    this.qty = qty;
    this.price = price;
}

// Getter and setter for active
public boolean isActive() {
    return active;
}

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

// Calculate inventory value
public double getInventoryValue() {
    return price * qty;
}
```

```
    // String representation of the Product
    @Override
    public String toString() {
        return "Item Number : " + itemNumber + "\n" +
               "Name : " + name + "\n" +
               "Quantity in stock: " + qty + "\n" +
               "Price : " + price + "\n" +
               "Stock Value : " + getInventoryValue() + "\n" +
               "Product Status : " + (active ? "Active (true)" : "Discontinued (false)");
    }
}
import java.util.Scanner;

public class ProductTester {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);

        // Temporary variables for product attributes
        int tempNumber;
        String tempName;
        int tempQty;
        double tempPrice;

        // Input for p1
        System.out.println("Enter Item Number: ");
        tempNumber = in.nextInt();

        // Clear the input buffer
```

```
in.nextLine();

System.out.println("Enter Name: ");
tempName = in.nextLine();

System.out.println("Enter Quantity: ");
tempQty = in.nextInt();

System.out.println("Enter Price: ");
tempPrice = in.nextDouble();

// Create p1
Product p1 = new Product(tempNumber, tempName, tempQty, tempPrice);
System.out.println(p1); // Display p1 information

// Clear the input buffer before getting values for p2
in.nextLine();

// Input for p2
System.out.println("Enter Item Number for second product: ");
tempNumber = in.nextInt();

// Clear the input buffer
in.nextLine();

System.out.println("Enter Name for second product: ");
tempName = in.nextLine();
```

```
        System.out.println("Enter Quantity for second product: ");
        tempQty = in.nextInt();

        System.out.println("Enter Price for second product: ");
        tempPrice = in.nextDouble();

        // Create p2
        Product p2 = new Product(tempNumber, tempName, tempQty, tempPrice);
        System.out.println(p2); // Display p2 information

        // Set active status for p2 to false
        p2.setActive(false);
        System.out.println(p2); // Display p2 with updated active status

        // Close Scanner
        in.close();
    }
}
```



```
Enter Item Number:
1
Enter Name:
BOOK
Enter Quantity:
2
Enter Price:
30
Item Number: 1
Name: BOOK
Quantity in stock: 2
Price: 30.0
Enter Item Number for second product:
2
Enter Name for second product:
PEN
Enter Quantity for second product:
10
Enter Price for second product:
5
Item Number: 2
Name: PEN
Quantity in stock: 10
Price: 5.0

=== Code Execution Successful ===
```

PROJECT-3:

Java Fundamentals
Section 6: Creating an Inventory Project

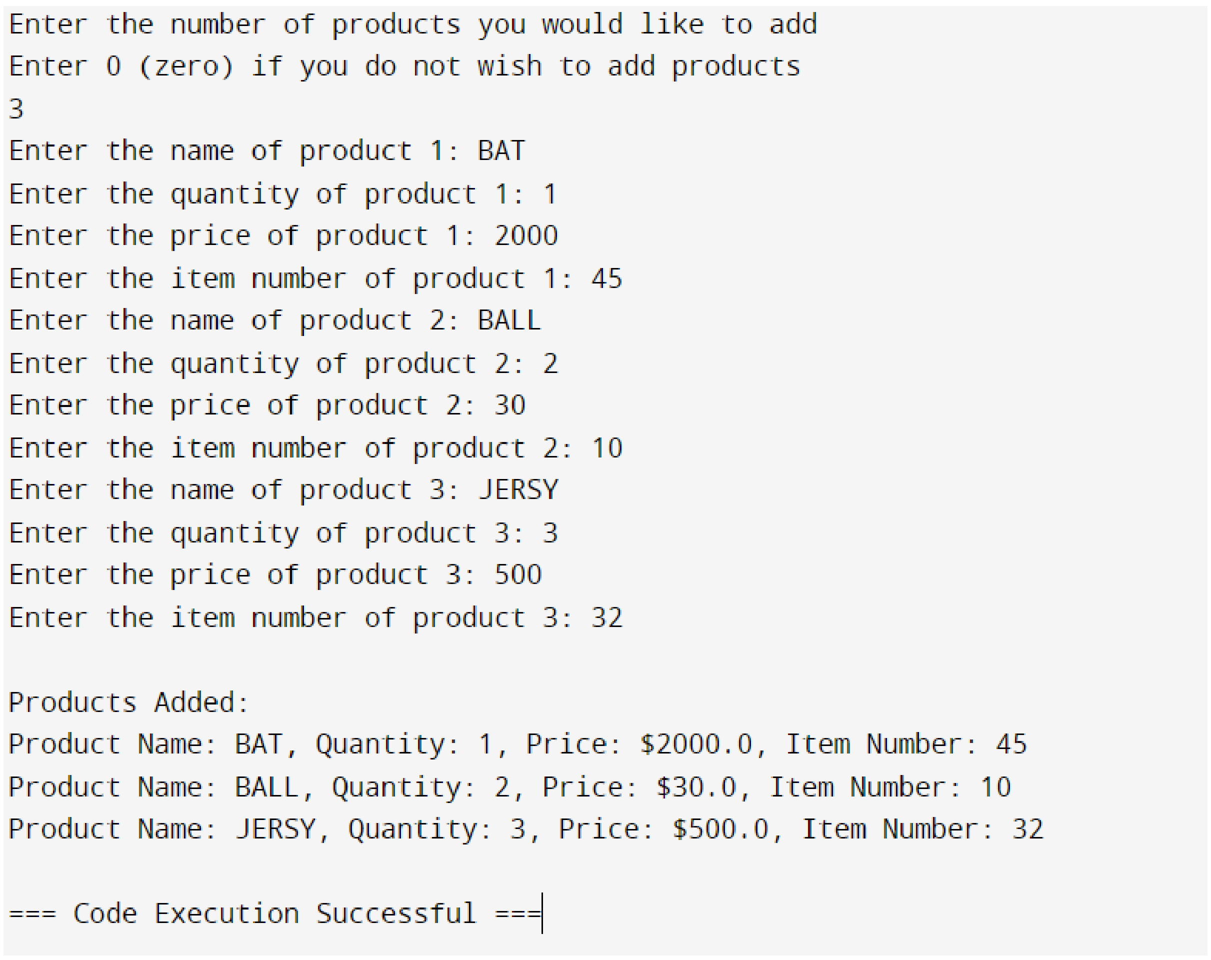
PROGRAM:

```
import java.util.Scanner;

import java.util.InputMismatchException;


class Product {

  private String name;
```

```
    private int quantity;

    private double price;

    private int itemNumber;


    // Constructor

    public Product(String name, int quantity, double price, int itemNumber) {

        this.name = name;

        this.quantity = quantity;

        this.price = price;

        this.itemNumber = itemNumber;

    }


    @Override

    public String toString() {

        return "Product Name: " + name + ", Quantity: " + quantity +

                ", Price: $" + price + ", Item Number: " + itemNumber;

    }

}


public class ProductTester {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        int maxSize = -1; // Initializing with a value to force a correct input later


        // Prompt for the number of products

        System.out.println("Enter the number of products you would like to add");

        System.out.println("Enter 0 (zero) if you do not wish to add products");
```

```
// Input loop
do {
  try {
    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
    // Continue the loop after clearing the buffer
  }
} while (maxSize < 0); // Exit on 0 or greater

// Handle the case of no products
if (maxSize == 0) {
  System.out.println("No products required!");
} else { // Handle positive maxSize
  // Create an array to store Product objects
  Product[] products = new Product[maxSize];

  // Populate the array with product details
  for (int i = 0; i < maxSize; i++) {
    scanner.nextLine(); // Clear the input buffer

    System.out.print("Enter the name of product " + (i + 1) + ": ");
```

```
            String name = scanner.nextLine();

            System.out.print("Enter the quantity of product " + (i + 1) + ": ");
            int quantity = scanner.nextInt();

            System.out.print("Enter the price of product " + (i + 1) + ": ");
            double price = scanner.nextDouble();

            System.out.print("Enter the item number of product " + (i + 1) + ": ");
            int itemNumber = scanner.nextInt();

            // Create a new product object and place it in the array
            products[i] = new Product(name, quantity, price, itemNumber);
          }

          // Display the products using a for-each loop
          System.out.println("\nProducts Added:");
          for (Product product : products) {
            System.out.println(product);
          }
        }

        // Close the scanner
        scanner.close();
      }
    }
```


```
Enter the number of products you would like to add
Enter 0 (zero) if you do not wish to add products
3
Enter the name of product 1: BAT
Enter the quantity of product 1: 1
Enter the price of product 1: 2000
Enter the item number of product 1: 45
Enter the name of product 2: BALL
Enter the quantity of product 2: 2
Enter the price of product 2: 30
Enter the item number of product 2: 10
Enter the name of product 3: JERSY
Enter the quantity of product 3: 3
Enter the price of product 3: 500
Enter the item number of product 3: 32

Products Added:
Product Name: BAT, Quantity: 1, Price: $2000.0, Item Number: 45
Product Name: BALL, Quantity: 2, Price: $30.0, Item Number: 10
Product Name: JERSY, Quantity: 3, Price: $500.0, Item Number: 32

=== Code Execution Successful ===
```

PROJECT-4:

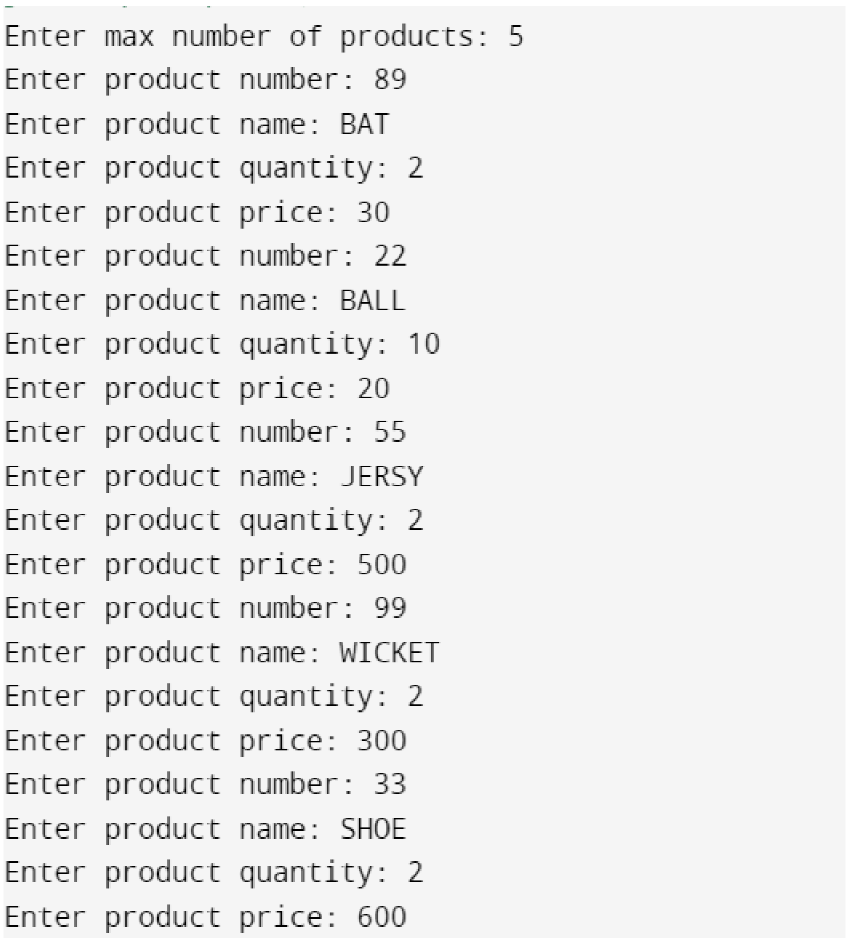
Java Fundamentals
Section 7 Part 1: Creating an Inventory Project.

```
import java.util.Scanner;


class Product {
  private int number;
  private String name;
  private int quantity;
  private double price;


  // Constructor
  public Product(int number, String name, int quantity, double price) {
    this.number = number;

    this.name = name;

    this.quantity = quantity;

    this.price = price;

}


// Getters

public String getName() {

    return name;

}


public int getQuantity() {

    return quantity;

}


// Method to add quantity

public void addToInventory(int quantity) {

    if (quantity > 0) {

        this.quantity += quantity;

    } else {

        System.out.println("Quantity must be greater than zero.");

    }

}


// Method to deduct quantity

public void deductFromInventory(int quantity) {

    if (quantity > 0 && quantity <= this.quantity) {

        this.quantity -= quantity;
        } else {
            System.out.println("Invalid quantity for deduction.");
        }
    }
}

public class ProductTester {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int maxSize = getNumProducts(scanner);
        Product[] products = new Product[maxSize];

        addToInventory(products, scanner);
        displayInventory(products);

        int option;
        do {
            option = getMenuOption(scanner);
            switch (option) {
                case 1:
                    displayInventory(products);
                    break;
                case 2:
                    addInventory(products, scanner);
                    break;
                case 3:
                    deductInventory(products, scanner);
                    break;
                case 4:
                    discontinueProduct(products, scanner);
                    break;
            }
        } while (option != 0);

        scanner.close();
    }

    public static void displayInventory(Product[] products) {
        System.out.println("Current Inventory:");
        for (int i = 0; i < products.length; i++) {
            if (products[i] != null) {
                System.out.println(i + ": " + products[i].getName() + " - Quantity: " +
products[i].getQuantity());
            }
        }
    }

    public static void addToInventory(Product[] products, Scanner scanner) {
        int tempNumber;
        String tempName;
        int tempQty;
        double tempPrice;

        for (int i = 0; i < products.length; i++) {
            System.out.print("Enter product number: ");
            tempNumber = scanner.nextInt();
            System.out.print("Enter product name: ");
            tempName = scanner.next();

            System.out.print("Enter product quantity: ");
            tempQty = scanner.nextInt();

            System.out.print("Enter product price: ");
            tempPrice = scanner.nextDouble();

            products[i] = new Product(tempNumber, tempName, tempQty, tempPrice);
        }
    }

    public static int getNumProducts(Scanner scanner) {
        int maxSize;
        do {
            System.out.print("Enter max number of products: ");
            maxSize = scanner.nextInt();
        } while (maxSize <= 0);
        return maxSize;
    }

    public static int getMenuOption(Scanner scanner) {
        int option = -1;
        while (option < 0 || option > 4) {
            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: ");
            try {
                option = scanner.nextInt();
            } catch (Exception e) {
                System.out.println("Invalid input. Please enter a number between 0 and 4.");
                scanner.next(); // Clear the invalid input
            }
        }
        return option;
    }

    public static int getProductNumber(Product[] products, Scanner scanner) {
        int productChoice = -1;
        while (productChoice < 0 || productChoice >= products.length) {
            System.out.println("Select a product by number:");
            for (int i = 0; i < products.length; i++) {
                if (products[i] != null) {
                    System.out.println(i + ": " + products[i].getName());
                }
            }
            try {
                productChoice = scanner.nextInt();
            } catch (Exception e) {
                System.out.println("Invalid input. Please enter a valid product number.");
                scanner.next(); // Clear the invalid input
```

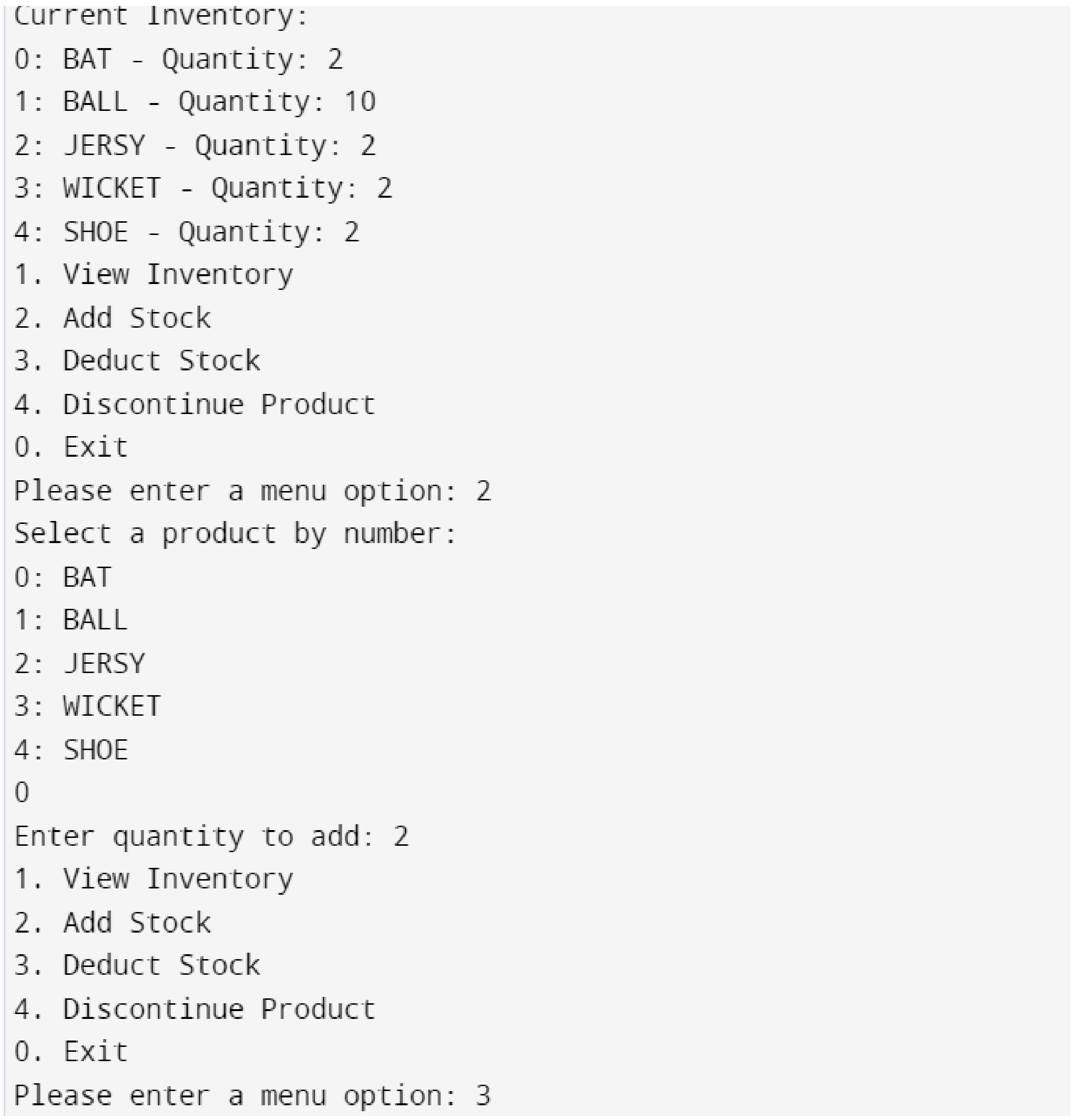
```
      }
    }
    return productChoice;
  }

  public static void addInventory(Product[] products, Scanner scanner) {
    int productChoice;
    int updateValue = -1;

    productChoice = getProductNumber(products, scanner);

    while (updateValue < 0) {
      System.out.print("Enter quantity to add: ");
      updateValue = scanner.nextInt();
    }

    products[productChoice].addToInventory(updateValue);
  }

  public static void deductInventory(Product[] products, Scanner scanner) {
    int productChoice;
    int updateValue = -1;

    productChoice = getProductNumber(products, scanner);

    while (updateValue < 0) {
      System.out.print("Enter quantity to deduct: ");
      updateValue = scanner.nextInt();
```

```
        }

        products[productChoice].deductFromInventory(updateValue);
    }

    public static void discontinueProduct(Product[] products, Scanner scanner) {
        int productChoice = getProductNumber(products, scanner);
        products[productChoice] = null; // Setting the product to null to discontinue it
        System.out.println("Product discontinued.");
    }
}
```

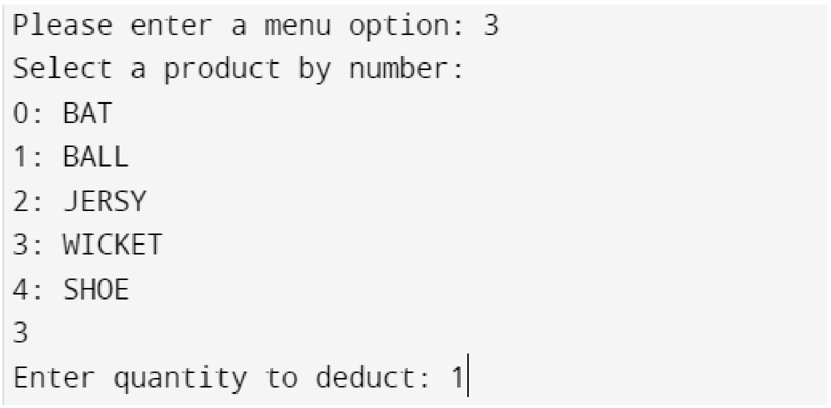
OUTPUT:



```
Enter max number of products: 5
Enter product number: 89
Enter product name: BAT
Enter product quantity: 2
Enter product price: 30
Enter product number: 22
Enter product name: BALL
Enter product quantity: 10
Enter product price: 20
Enter product number: 55
Enter product name: JERSY
Enter product quantity: 2
Enter product price: 500
Enter product number: 99
Enter product name: WICKET
Enter product quantity: 2
Enter product price: 300
Enter product number: 33
Enter product name: SHOE
Enter product quantity: 2
Enter product price: 600
```



```
Current Inventory:
0: BAT - Quantity: 2
1: BALL - Quantity: 10
2: JERSY - Quantity: 2
3: WICKET - Quantity: 2
4: SHOE - Quantity: 2
1. View Inventory
2. Add Stock
3. Deduct Stock
4. Discontinue Product
0. Exit
Please enter a menu option: 2
Select a product by number:
0: BAT
1: BALL
2: JERSY
3: WICKET
4: SHOE
0
Enter quantity to add: 2
1. View Inventory
2. Add Stock
3. Deduct Stock
4. Discontinue Product
0. Exit
Please enter a menu option: 3
```



```
Please enter a menu option: 3
Select a product by number:
0: BAT
1: BALL
2: JERSY
3: WICKET
4: SHOE
3
Enter quantity to deduct: 1
```

PROJECT-5:

PROGRAM FOR CD AND DVD:

```
import java.util.ArrayList;

import java.util.Scanner;
```

```
class Product {
  protected String name;
  protected double price;
  protected int quantity;
  protected int itemNumber;
  protected String status = "Available";

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

  public double calculateInventoryValue() {
    return price * quantity;
  }

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

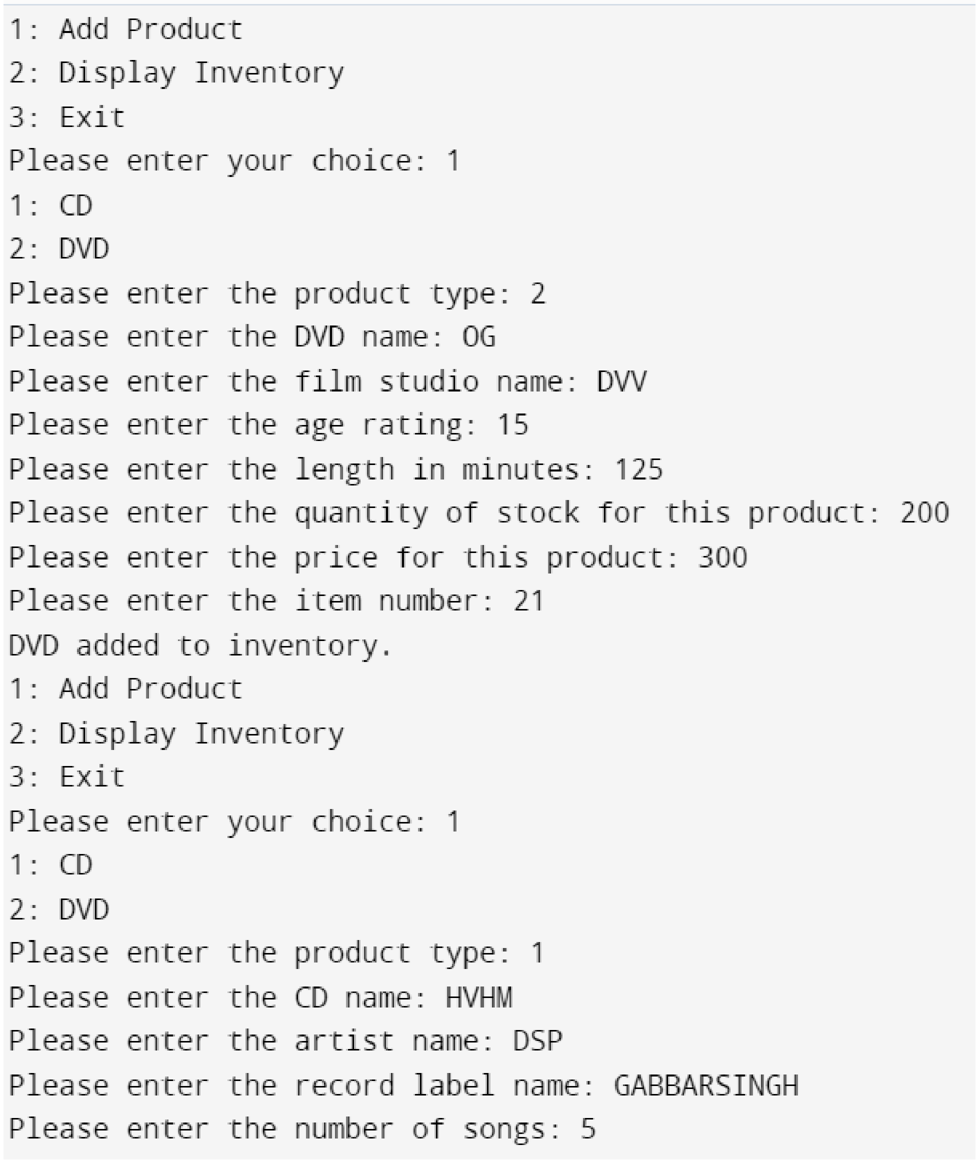
```
}


class DVD extends Product {
   private int length;
   private int ageRating;
   private String filmStudio;


   public DVD(String name, double price, int quantity, int itemNumber, int length, int
ageRating, String filmStudio) {
      super(name, price, quantity, itemNumber);
      this.length = length;
      this.ageRating = ageRating;
      this.filmStudio = filmStudio;
   }


   @Override
   public String toString() {
      return super.toString() + "\n" +
            "Movie Length: " + length + " minutes\n" +
            "Age Rating: " + ageRating + "\n" +
            "Film Studio: " + filmStudio;
   }
}


class CD extends Product {
   private String artist;
   private int numSongs;
   private String label;
```

```
  public CD(String name, double price, int quantity, int itemNumber, String artist, int
numSongs, String label) {
    super(name, price, quantity, itemNumber);
    this.artist = artist;
    this.numSongs = numSongs;
    this.label = label;
  }

  @Override
  public String toString() {
    return super.toString() + "\n" +
        "Artist: " + artist + "\n" +
        "Songs on Album: " + numSongs + "\n" +
        "Record Label: " + label;
  }
}

Public class ProductTester {
  private ArrayList<Product> products = new ArrayList<>();
  private Scanner scanner = new Scanner(System.in);

  public void addToInventory() {
    int stockChoice = -1;

    while (stockChoice != 1 && stockChoice != 2) {
      System.out.println("1: CD\n2: DVD");
      System.out.print("Please enter the product type: ");
```

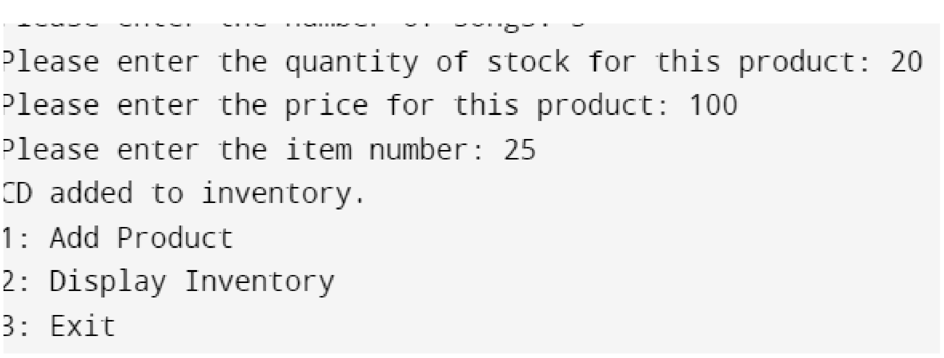
```
        stockChoice = scanner.nextInt();
        scanner.nextLine(); // Consume newline

        if (stockChoice != 1 && stockChoice != 2) {
          System.out.println("Only numbers 1 or 2 allowed!");
        }
      }

      if (stockChoice == 1) {
        addCDToInventory();
      } else {
        addDVDToInventory();
      }
    }

    private void addCDToInventory() {
      System.out.print("Please enter the CD name: ");
      String name = scanner.nextLine();

      System.out.print("Please enter the artist name: ");
      String artist = scanner.nextLine();

      System.out.print("Please enter the record label name: ");
      String label = scanner.nextLine();

      System.out.print("Please enter the number of songs: ");
      int numSongs = scanner.nextInt();
```

```
        System.out.print("Please enter the quantity of stock for this product: ");
        int quantity = scanner.nextInt();

        System.out.print("Please enter the price for this product: ");
        double price = scanner.nextDouble();

        System.out.print("Please enter the item number: ");
        int itemNumber = scanner.nextInt();

        CD cd = new CD(name, price, quantity, itemNumber, artist, numSongs, label);
        products.add(cd);
        System.out.println("CD added to inventory.");
    }

    private void addDVDToInventory() {
        System.out.print("Please enter the DVD name: ");
        String name = scanner.nextLine();

        System.out.print("Please enter the film studio name: ");
        String filmStudio = scanner.nextLine();

        System.out.print("Please enter the age rating: ");
        int ageRating = scanner.nextInt();

        System.out.print("Please enter the length in minutes: ");
        int length = scanner.nextInt();

        System.out.print("Please enter the quantity of stock for this product: ");
        int quantity = scanner.nextInt();


        System.out.print("Please enter the price for this product: ");
        double price = scanner.nextDouble();


        System.out.print("Please enter the item number: ");
        int itemNumber = scanner.nextInt();


        DVD dvd = new DVD(name, price, quantity, itemNumber, length, ageRating,
filmStudio);
        products.add(dvd);
        System.out.println("DVD added to inventory.");
    }


    public void displayInventory() {
        for (Product product : products) {
            System.out.println(product);
            System.out.println("\n" + "=".repeat(40) + "\n");
        }
    }


    public static void main(String[] args) {
        ProductTester tester = new ProductTester();


        while (true) {
            System.out.println("1: Add Product\n2: Display Inventory\n3: Exit");
            System.out.print("Please enter your choice: ");
            int choice = tester.scanner.nextInt();
            tester.scanner.nextLine(); // Consume newline


            if (choice == 1) {
                tester.addToInventory();
            } else if (choice == 2) {
                tester.displayInventory();
            } else if (choice == 3) {
                break;
            } else {
                System.out.println("Invalid choice. Please try again.");
            }
        }
        tester.scanner.close();
    }
}
```

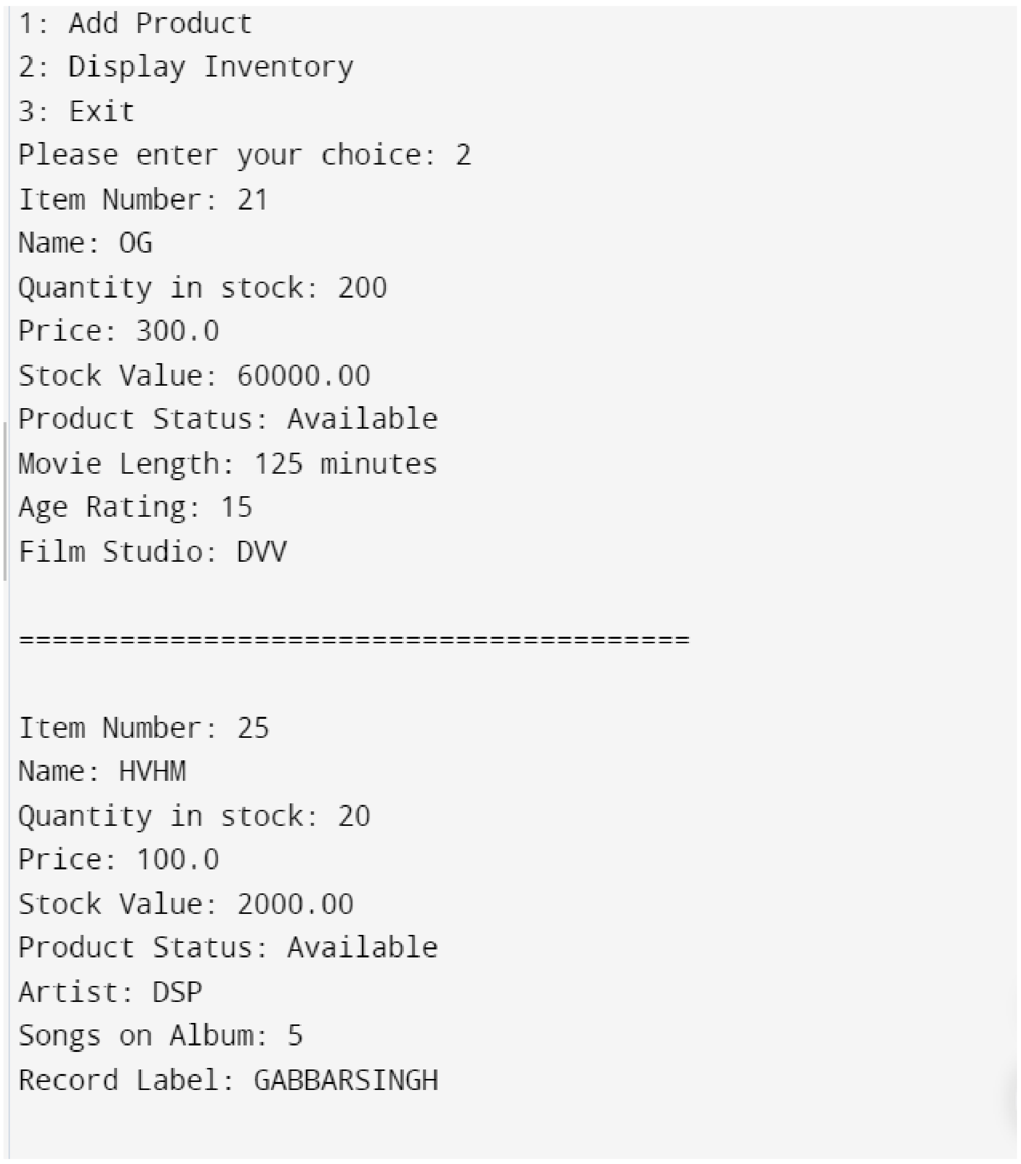
Output:



```
1: Add Product
2: Display Inventory
3: Exit
Please enter your choice: 1
1: CD
2: DVD
Please enter the product type: 2
Please enter the DVD name: OG
Please enter the film studio name: DVV
Please enter the age rating: 15
Please enter the length in minutes: 125
Please enter the quantity of stock for this product: 200
Please enter the price for this product: 300
Please enter the item number: 21
DVD added to inventory.
1: Add Product
2: Display Inventory
3: Exit
Please enter your choice: 1
1: CD
2: DVD
Please enter the product type: 1
Please enter the CD name: HVHM
Please enter the artist name: DSP
Please enter the record label name: GABBARSINGH
Please enter the number of songs: 5
```



```
Please enter the quantity of stock for this product: 20
Please enter the price for this product: 100
Please enter the item number: 25
CD added to inventory.
1: Add Product
2: Display Inventory
3: Exit
```



```
1: Add Product
2: Display Inventory
3: Exit
Please enter your choice: 2
Item Number: 21
Name: OG
Quantity in stock: 200
Price: 300.0
Stock Value: 60000.00
Product Status: Available
Movie Length: 125 minutes
Age Rating: 15
Film Studio: DVV

========================================

Item Number: 25
Name: HVHM
Quantity in stock: 20
Price: 100.0
Stock Value: 2000.00
Product Status: Available
Artist: DSP
Songs on Album: 5
Record Label: GABBARSINGH
```

Final project:

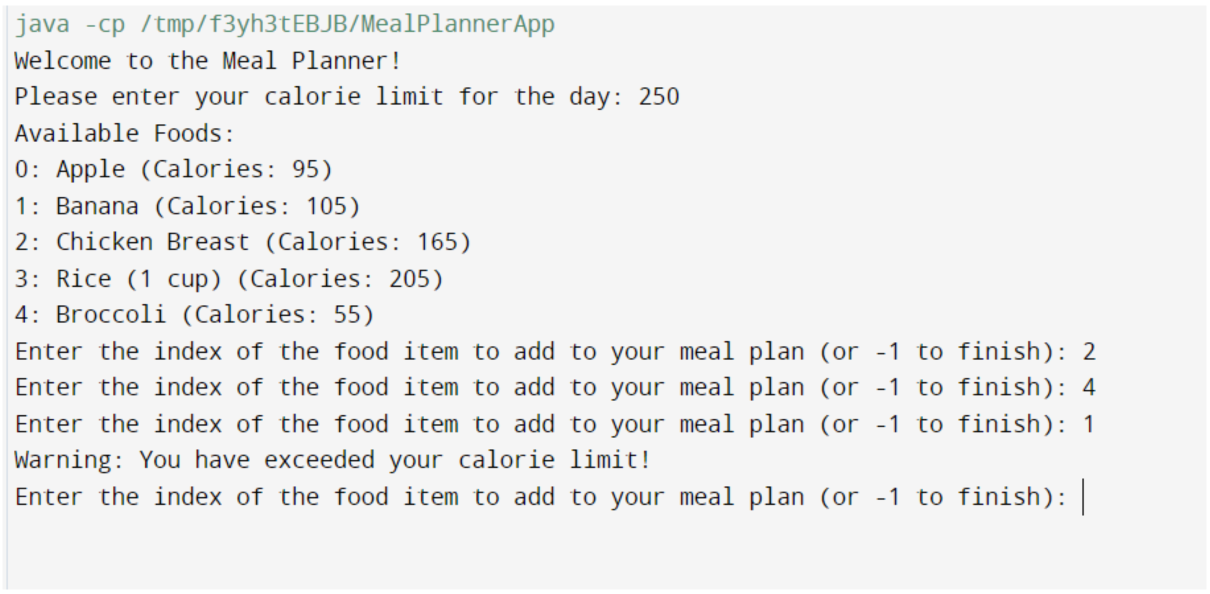
```
import java.util.ArrayList;

import java.util.InputMismatchException;

import java.util.Scanner;


public class MealPlannerApp {


  public static void main(String[] args) {

    Scanner scanner = new Scanner(System.in);

    MealPlanner mealPlanner = new MealPlanner();
```

```java
    System.out.println("Welcome to the Meal Planner!");

    int calorieLimit = 0;
    while (true) {
      System.out.print("Please enter your calorie limit for the day: ");
      try {
        calorieLimit = scanner.nextInt();
        if (calorieLimit <= 0) {
          System.out.println("Calorie limit must be a positive number. Try again.");
        } else {
          break;
        }
      } catch (InputMismatchException e) {
        System.out.println("Invalid input. Please enter a number.");
        scanner.next(); // Clear invalid input
      }
    }

    mealPlanner.displayFoodOptions();

    while (true) {
      System.out.print("Enter the index of the food item to add to your meal plan (or -1
  to finish): ");
      int inputIndex = scanner.nextInt();
      if (inputIndex == -1) {
        break;
      }
```

```
            if (inputIndex >= 0 && inputIndex < mealPlanner.foodList.size()) {
                mealPlanner.addFoodToMealPlan(mealPlanner.getFood(inputIndex));
                if (!mealPlanner.isUnderCalorieLimit(calorieLimit)) {
                    System.out.println("Warning: You have exceeded your calorie limit!");
                }
            } else {
                System.out.println("Invalid index. Please try again.");
            }
        }

        mealPlanner.displayMealPlan();
        System.out.println("Total calories: " + mealPlanner.totalCalories());

        scanner.close();
    }

    static class Food {
        private String name;
        private int calories;

        public Food(String name, int calories) {
            this.name = name;
            this.calories = calories;
        }

        public String getName() {
            return name;
        }
    public int getCalories() {
        return calories;
    }

    public String getDescription() {
        return name + " (Calories: " + calories + ")";
    }
}

static class MealPlanner {
    private ArrayList<Food> foodList;
    private ArrayList<Food> mealPlan;

    public MealPlanner() {
        foodList = new ArrayList<>();
        mealPlan = new ArrayList<>();
        initializeFoodList();
    }

    private void initializeFoodList() {
        foodList.add(new Food("Apple", 95));
        foodList.add(new Food("Banana", 105));
        foodList.add(new Food("Chicken Breast", 165));
        foodList.add(new Food("Rice (1 cup)", 205));
        foodList.add(new Food("Broccoli", 55));
    }
```

```
public void addFoodToMealPlan(Food food) {
    mealPlan.add(food);
}

public void displayMealPlan() {
    System.out.println("Your Meal Plan:");
    for (Food food : mealPlan) {
        System.out.println(food.getDescription());
    }
}

public int totalCalories() {
    int total = 0;
    for (Food food : mealPlan) {
        total += food.getCalories();
    }
    return total;
}

public boolean isUnderCalorieLimit(int limit) {
    return totalCalories() <= limit;
}

public void displayFoodOptions() {
    System.out.println("Available Foods:");
    for (int i = 0; i < foodList.size(); i++) {
        System.out.println(i + ": " + foodList.get(i).getDescription());
    }
    }

    public Food getFood(int index) {
      return foodList.get(index);
    }
  }
}
```

Output:



```
java -cp /tmp/f3yh3tEBJB/MealPlannerApp
Welcome to the Meal Planner!
Please enter your calorie limit for the day: 250
Available Foods:
0: Apple (Calories: 95)
1: Banana (Calories: 105)
2: Chicken Breast (Calories: 165)
3: Rice (1 cup) (Calories: 205)
4: Broccoli (Calories: 55)
Enter the index of the food item to add to your meal plan (or -1 to finish): 2
Enter the index of the food item to add to your meal plan (or -1 to finish): 4
Enter the index of the food item to add to your meal plan (or -1 to finish): 1
Warning: You have exceeded your calorie limit!
Enter the index of the food item to add to your meal plan (or -1 to finish): 
```