

ASSIGNMENT-6.4

2303A51595

B-10

TASK – 1

Prompt :

Write a Java program to accept day-wise product sales using Scanner and calculate daily totals and overall monthly sales.

Display a summary report and allow the user to repeat the process for another month using loops.

Code :

```
import java.util.Scanner;
```

```
class Student {  
    String name;  
    int rollNumber;  
    int marks;  
    Student(String name, int rollNumber, int marks) {  
        this.name = name;  
        this.rollNumber = rollNumber;  
        this.marks = marks;  
    }
```

```
    void displayDetails() {  
        System.out.println("Name: " + name);  
        System.out.println("Roll Number: " + rollNumber);  
        System.out.println("Marks: " + marks);  
    }
```

```
    String checkPerformance(int average) {  
        if (marks > average)  
            return "Above Class Average";  
        else  
            return "Below Class Average";  
    }  
}
```

```
public class StudentTest {
```

```

public static void main(String[] args) {

    Scanner sc = new Scanner(System.in);

    System.out.print("Enter Name: ");
    String name = sc.nextLine();

    System.out.print("Enter Roll Number: ");
    int roll = sc.nextInt();

    System.out.print("Enter Marks: ");
    int marks = sc.nextInt();

    System.out.print("Enter Class Average: ");
    int avg = sc.nextInt();

    Student s = new Student(name, roll, marks);

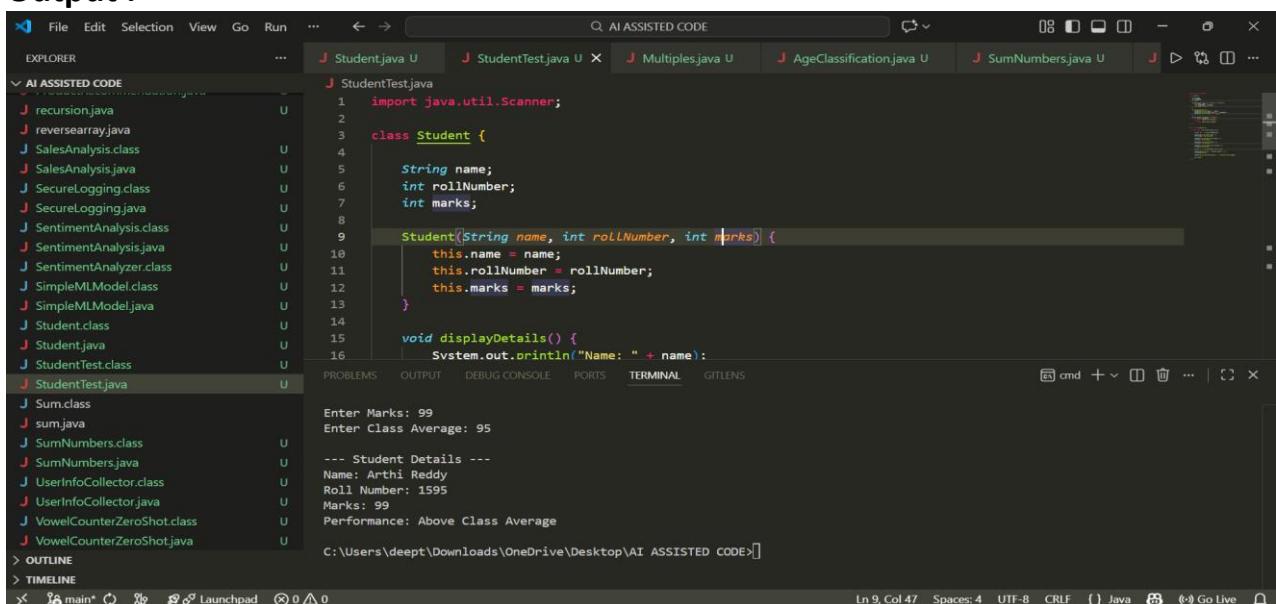
    System.out.println("\n--- Student Details ---");
    s.displayDetails();

    System.out.println("Performance: " + s.checkPerformance(avg));

    sc.close();
}

```

Output :



The screenshot shows a Java development environment with multiple tabs open. The active tab is 'StudentTest.java'. The code in the editor is as follows:

```

import java.util.Scanner;
class Student {
    String name;
    int rollNumber;
    int marks;

    Student(String name, int rollNumber, int marks) {
        this.name = name;
        this.rollNumber = rollNumber;
        this.marks = marks;
    }

    void displayDetails() {
        System.out.println("Name: " + name);
    }
}

```

The 'TERMINAL' tab shows the execution of the code. The user enters '99' for marks and '95' for average. The program outputs the student details, including the calculated performance.

```

Enter Marks: 99
Enter Class Average: 95
--- Student Details ---
Name: Arthi Reddy
Roll Number: 1595
Marks: 99
Performance: Above Class Average

```

Analysis :

This program takes product sales details for each day and calculates daily and monthly totals using loops.

It shows the final sales summary and allows the user to repeat the process for the next month.

TASK-2**Prompt :**

Write a Java program to loop through sensor readings, identify even numbers, calculate their square, and print the result clearly.

Code :

```
import java.util.*;  
  
public class SensorMonitoring {  
  
    public static void main(String[] args) {  
  
        Scanner sc = new Scanner(System.in);  
  
        System.out.print("Enter number of readings: ");  
  
        int n = sc.nextInt();  
  
        int[] readings = new int[n];  
  
        System.out.println("Enter sensor readings:");  
  
        for(int i = 0; i < n; i++) {  
  
            readings[i] = sc.nextInt();  
  
            // Copilot: check each reading, if even calculate square and print nicely  
  
            for(int value : readings) {  
  
                if(value % 2 == 0) {  
  
                    int square = value * value;  
  
                    System.out.println("Even: " + value + " -> Square: " + square)  
  
                    sc.close();  
                }  
            }  
        }  
    }  
}
```

Output :

The screenshot shows a Java code editor interface with the following details:

- File Explorer:** Shows multiple Java files including Student.java, StudentTest.java, SensorMonitoring.java, Multiples.java, and AgeClassification.java.
- Code Editor:** Displays the content of SensorMonitoring.java:

```
1 import java.util.*;
2 +
3 public class SensorMonitoring {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter number of readings: ");
8         int n = sc.nextInt();
9
10        int[] readings = new int[n];
11
12        System.out.println("Enter sensor readings:");
13        for(int i = 0; i < n; i++) {
14            readings[i] = sc.nextInt();
15        }
16    }
}
```
- Terminal:** Shows the command-line output of running the Java code:

```
C:\Users\deeps\Downloads\OneDrive\Desktop\AI ASSISTED CODE>javac SensorMonitoring.java
C:\Users\deeps\Downloads\OneDrive\Desktop\AI ASSISTED CODE>java SensorMonitoring
Enter number of readings: 5
Enter sensor readings:
2 5 8 3 6
Even: 2 -> Square: 4
Even: 8 -> Square: 64
Even: 6 -> Square: 36
```
- Status Bar:** Shows the current line (Ln 3, Col 14), character count (17 selected), spaces (Spaces: 4), encoding (UTF-8), and file type (CRLF, Java).

Analysis :

The loop checks each reading using `% 2 == 0` to find even numbers. If even, it calculates the square and prints the result clearly.

TASK-3

Prompt :

Write a Java class BankAccount with accountHolder and balance. Add methods for deposit, withdraw (with insufficient balance check), and display balance using if-else conditions.

Code :

```
import java.util.Scanner;
class BankAccount {
    String accountHolder;
    double balance;
    BankAccount(String accountHolder, double balance) {
        this.accountHolder = accountHolder;
        this.balance = balance;
    }
    void deposit(double amount) {
        balance += amount;
```

```

        System.out.println("Deposited: " + amount);
    }

    void withdraw(double amount) {
        if (amount <= balance) {
            balance -= amount;
            System.out.println("Withdrawn: " + amount);
        } else {
            System.out.println("Insufficient balance! Withdrawal failed.");
        }
    }

    public class BankSimulation {
        public static void main(String[] args) {
            Scanner sc = new Scanner(System.in);
            System.out.print("Enter account holder name: ");
            String name = sc.nextLine();
            System.out.print("Enter initial balance: ");
            double bal = sc.nextDouble();
            BankAccount acc = new BankAccount(name, bal);
            System.out.print("Enter deposit amount: ");
            acc.deposit(sc.nextDouble());
            System.out.print("Enter withdrawal amount: ");
            acc.withdraw(sc.nextDouble());
            acc.checkBalance();
            sc.close();
        }
    }
}

```

Output :

The screenshot shows a Java development environment with the following details:

- File Explorer:** Shows various Java files in the project, including Student.java, StudentTest.java, SensorMonitoring.java, BankSimulation.java, and Multiples.java.
- Code Editor:** Displays the BankSimulation.java code. The cursor is currently at line 31, which defines the main method.
- Terminal:** Shows the command-line output of running the code. It includes the command `C:\Users\deept\Downloads\OneDrive\Desktop\AI ASSISTED CODE>javac BankSimulation.java`, followed by the program's interaction with the user to enter account holder name, initial balance, deposit amount, and withdrawal amount, and finally the error message "Insufficient balance! Withdrawal failed.".
- Status Bar:** Provides information about the current file (main.java), line count (Ln 32, Col 45), character count (Spaces: 4, UTF-8, CRLF), and other development tools like Java and Go Live.

Analysis :

The class stores account details and updates balance using deposit and withdraw methods. If withdrawal is more than balance, an if-else check prevents it and shows a friendly message.

Task-4**Prompt :**

Write a Java program that stores student names and scores, use a while loop to check each student, and print names of students scoring more than 75.

Code :

```
import java.util.Scanner;

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

        Scanner sc = new Scanner(System.in);

        System.out.print("Enter number of students: ");
        int n = sc.nextInt();
        sc.nextLine();

        String[] names = new String[n];
        int[] scores = new int[n];

        // initialize student data
        for (int i = 0; i < n; i++) {
            System.out.print("Enter name: ");
            names[i] = sc.nextLine();

            System.out.print("Enter score: ");
            scores[i] = sc.nextInt();
            sc.nextLine();
        }

        int i = 0;
        System.out.println("\nEligible Students:");
        while (i < n) {
```

```

        if (scores[i] > 75) {
            System.out.println(names[i]);
        }
        i++;
    }
    sc.close();
}

```

Output :

The screenshot shows a Java development environment with the following details:

- File Bar:** File, Edit, Selection, View, Go, Run, ...
- Toolbar:** Standard icons for file operations.
- Code Editor:** Displays the `ScholarshipCheck.java` file content.
- Output Terminal:** Shows the command-line output of running the Java program.
- Explorer:** Shows a list of Java files in the project.
- Right Sidebar:** Contains icons for Java, cmd, and cmd.
- Bottom Status Bar:** Shows the current file path (C:\Users\deeps\Downloads\OneDrive\Desktop\AI ASSISTED CODE), line count (Ln 11, Col 1), space count (Spaces: 4), encoding (UTF-8), and file type (Java).

```

File Edit Selection View Go Run ...
PROBLEMS OUTPUT DEBUG CONSOLE PORTS TERMINAL GITLENS
Microsoft Windows [Version 10.0.26200.7705]
(c) Microsoft Corporation. All rights reserved.

C:\Users\deeps\Downloads\OneDrive\Desktop\AI ASSISTED CODE>javac ScholarshipCheck.java
C:\Users\deeps\Downloads\OneDrive\Desktop\AI ASSISTED CODE>java ScholarshipCheck
Enter number of students: 5
Enter name: Arthi reddy
Enter score: 90
Enter name: Varsha
Enter score: 95
Enter name: Rashmitha
Enter score: 98
Enter name: Madhuri
Enter score: 98
Enter name: Nithya
Enter score: 95
Eligible Students:
Arthi reddy
Varsha
Rashmitha
Madhuri
Nithya

```

Analysis :

The while loop checks each student one by one using an index. If score is greater than 75, the student name is printed as eligible.

TASK-5

Prompt :

Write a Java class ShoppingCart with an item list. Add methods to add items, remove items, calculate total bill using loops, and apply discount if total exceeds a limit.

Code :

```

import java.util.*;
class Item {
    String name;
    double price;
}

```

```
int quantity;

Item(String name, double price, int quantity) {
    this.name = name;
    this.price = price;
    this.quantity = quantity;
}
}

class ShoppingCart {
    ArrayList<Item> items = new ArrayList<>();

    // add item to cart
    void addItem(String name, double price, int quantity) {
        items.add(new Item(name, price, quantity));
        System.out.println("Item added to cart.");
    }

    // remove item from cart
    void removeItem(String name) {
        items.removeIf(item -> item.name.equalsIgnoreCase(name));
        System.out.println("Item removed from cart.");
    }

    // calculate total with discount
    void calculateBill() {
        double total = 0;

        for (Item item : items) {
            total += item.price * item.quantity;
        }

        System.out.println("Total before discount: " + total);

        if (total > 1000) {
            total = total * 0.9; // 10% discount
            System.out.println("Discount applied (10%)");
        }

        System.out.println("Final Bill: " + total);
    }
}

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

```

Scanner sc = new Scanner(System.in);
ShoppingCart cart = new ShoppingCart();
System.out.print("Enter number of items: ");
int n = sc.nextInt();
sc.nextLine();

for (int i = 0; i < n; i++) {
    System.out.print("Item name: ");
    String name = sc.nextLine();

    System.out.print("Price: ");
    double price = sc.nextDouble();

    cart.addItem(name, price, qty);
}

System.out.print("Enter item name to remove: ");
cart.removeItem(sc.nextLine());
cart.calculateBill();
sc.close();
}
}

```

Output :

The screenshot shows a Java IDE interface with the ShoppingCartDemo.java file open in the editor. The code defines a ShoppingCart class with methods to add items and calculate a bill. The main method prompts the user for the number of items, loops to get each item's name and price, adds them to the cart, and then removes one item before calculating the total bill.

```

File Edit Selection View Go Run ... ← → Q AI ASSISTED CODE 08 □ - ⌂ ⌂ ...
EXPLORER J SensorMonitoring.java J ShoppingCartDemo.java U J BankSimulation.java U J ScholarshipCheck.java U J ShoppingCartDemo.java U ...
AI ASSISTED CODE
J reversearray.java
J SalesAnalysis.class
J SalesAnalysis.java
J ScholarshipCheck.class
J ScholarshipCheck.java
J SecureLogging.class
J SecureLogging.java
J SensorMonitoring.class
J SensorMonitoring.java
J SentimentAnalysis.class
J SentimentAnalysis.java
J SentimentAnalyzer.class
J ShoppingCart.class
J ShoppingCartDemo.class
J ShoppingCartDemo.java U
J ShoppingCartDemo.java
J SimpleMLModel.class
J SimpleMLModel.java
J Student.class
J Student.java
J StudentTest.class
J StudentTest.java
J Sum.class
J sum.java
OUTLINE
TIMELINE
C:\Users\deepthi\Downloads\OneDrive\Desktop\AI ASSISTED CODE>java ShoppingCartDemo
Enter number of items: 2
Item name: rice
Price: 50
Quantity: 1
Item added to cart.
Item name: soap
Price: 40
Quantity: 3
Item added to cart.
Enter item name to remove: rice
Item removed from cart.
Total before discount: 120.0
Final Bill: 120.0

```

Analysis :

Items are stored in an ArrayList and processed using loops to calculate the total. If the bill exceeds the limit, an if-condition applies a discount automatically.