JAVA ASSIGNMENT 3

NAME-BHAVYA RATTAN

ROLL NO-2401201004

COURSE-BCA(AI&DS

CODE:-

ResultManager.java > ...

```java
import java.util.*;

// Custom Exception Class
class InvalidMarksException extends Exception {
    public InvalidMarksException(String message) {
        super(message);
    }
}

// Student Class
class Student {
    private int rollNumber;
    private String studentName;
    private int[] marks = new int[3];

    public Student(int rollNumber, String studentName, int[] marks) throws InvalidMarksException {
        this.rollNumber = rollNumber;
        this.studentName = studentName;
        this.marks = marks;
        validateMarks();
    }

    // Validate each mark
    public void validateMarks() throws InvalidMarksException {
        for (int i = 0; i < marks.length; i++) {
            if (marks[i] < 0 || marks[i] > 100) {
                throw new InvalidMarksException("Invalid marks for subject " + (i + 1) + ": " + marks[i]);
            }
        }
    }

    // Calculate average
    public double calculateAverage() {
        int total = 0;
        for (int m : marks) {
            total += m;
        }
        return total / 3.0;
    }

    // Display details
    public void displayResult() {
        double avg = calculateAverage();
```

ResultManager.java > ...

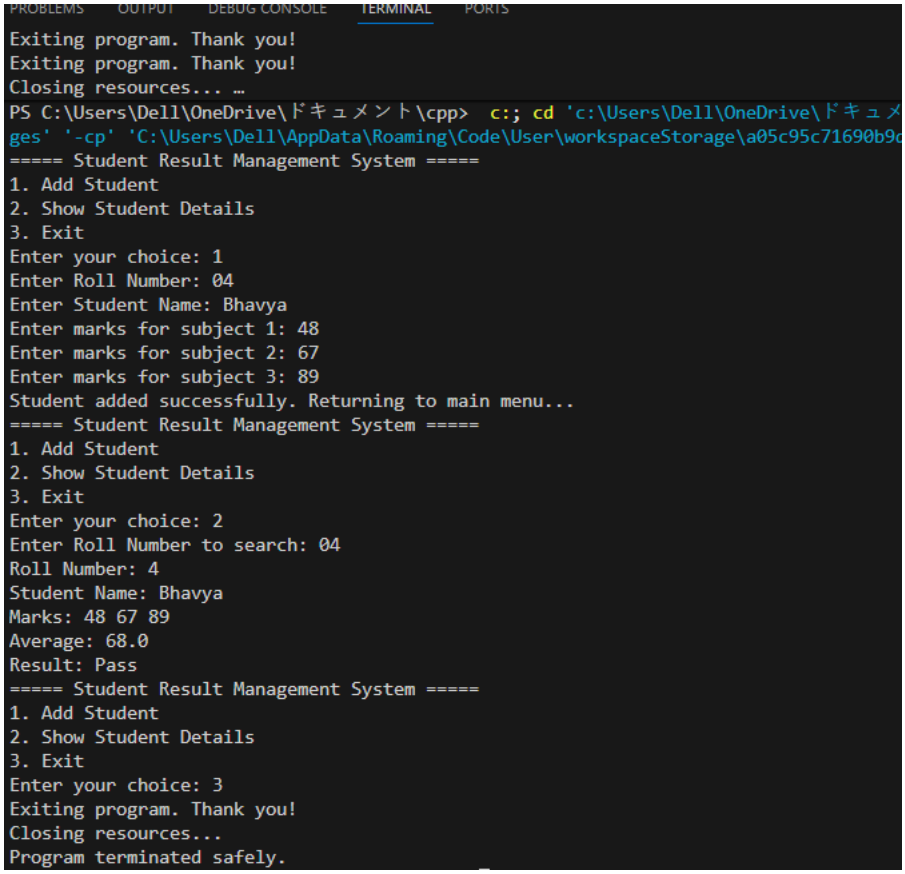
```
11  class Student {
42      public void displayResult() {
45
46          System.out.println("Roll Number: " + rollNumber);
47          System.out.println("Student Name: " + studentName);
48          System.out.print(s: "Marks: ");
49          for (int m : marks)
50              System.out.print(m + " ");
51          System.out.println();
52          System.out.println("Average: " + avg);
53          System.out.println("Result: " + status);
54      }
55
56      public int getRollNumber() {
57          return rollNumber;
58      }
59  }
60
61  // ResultManager Class
62  public class ResultManager {
63      private Student[] students;
64      private int count;
65      private Scanner sc;
66
67      public ResultManager() {
68          students = new Student[10]; // can store up to 10 students
69          count = 0;
70          sc = new Scanner(System.in);
71      }
72
73      // Add student details
74      public void addStudent() {
75          try {
76              System.out.print(s: "Enter Roll Number: ");
77              int roll = sc.nextInt();
78              sc.nextLine(); // consume newline
79
80              System.out.print(s: "Enter Student Name: ");
81              String name = sc.nextLine();
82
83              int[] marks = new int[3];
84              for (int i = 0; i < 3; i++) {
85                  System.out.print("Enter marks for subject " + (i + 1) + ": ");
86                  marks[i] = sc.nextInt();
```

```java
ResultManager.java > ...
public class ResultManager {
    public void addStudent() {
            }

            // Validate and create Student
            Student s = new Student(roll, name, marks);
            students[count++] = s;

            System.out.println(x: "Student added successfully. Returning to main menu...");

        } catch (InvalidMarksException e) {
            System.out.println("Error: " + e.getMessage());
        } catch (InputMismatchException e) {
            System.out.println(x: "Error: Invalid input type. Please enter correct data.");
            sc.nextLine(); // clear buffer
        } catch (Exception e) {
            System.out.println("Unexpected error: " + e.getMessage());
        }
    }

    // Show details of a specific student
    public void showStudentDetails() {
        try {
            System.out.print(s: "Enter Roll Number to search: ");
            int roll = sc.nextInt();

            boolean found = false;
            for (int i = 0; i < count; i++) {
                if (students[i].getRollNumber() == roll) {
                    students[i].displayResult();
                    found = true;
                    break;
                }
            }

            if (!found) {
                System.out.println("Student with Roll Number " + roll + " not found.");
            }

        } catch (InputMismatchException e) {
            System.out.println(x: "Error: Invalid roll number input.");
            sc.nextLine();
        } catch (Exception e) {
            System.out.println("Error: " + e.getMessage());
```

```
    public void showStudentDetails() {
            System.out.println("Error: " + e.getMessage());
        }
    }

    // Main Menu
    public void mainMenu() {
        int choice = 0;
        try {
            do {
                System.out.println("===== Student Result Management System =====");
                System.out.println("1. Add Student");
                System.out.println("2. Show Student Details");
                System.out.println("3. Exit");
                System.out.print("Enter your choice: ");

                choice = sc.nextInt();

                switch (choice) {
                    case 1:
                        addStudent();
                        break;
                    case 2:
                        showStudentDetails();
                        break;
                    case 3:
                        System.out.println("Exiting program. Thank you!");
                        break;
                    default:
                        System.out.println("Invalid choice! Please try again.");
                }

            } while (choice != 3);
        } catch (InputMismatchException e) {
            System.out.println("Error: Please enter a valid number choice.");
        } finally {
            // Resource cleanup
            System.out.println("Closing resources...");
            sc.close();
            System.out.println("Program terminated safely.");
        }
    }
}
```

```
    }

    // Main method
    Run | Debug
    public static void main(String[] args) {
        ResultManager manager = new ResultManager();
        manager.mainMenu();
    }
}
```

OUTPUT:-



Code explanation:-

Objective:

The main goal of this project is to design and implement a **Java application** that:

- Collects **student details** and **marks**.
- Calculates their **average** and **result status (Pass/Fail)**.
- Demonstrates **exception handling** using:
    - Built-in exceptions
    - Custom exceptions

Class Structure Overview:

The program consists of **three main classes**:

1. **InvalidMarksException** → Custom Exception Class
2. **Student** → Model Class for student data
3. **ResultManager** → Main Class (handles user interface and logic)

---

1. InvalidMarksException Class

Purpose:

- To handle **domain-specific errors**, i.e., when student marks are not within 0–100.

Description:

```
class InvalidMarksException extends Exception {
    public InvalidMarksException(String message) {
        super(message);
    }
}
```

- his class **extends the built-in Exception class**, making it a **checked exception**.
- It takes a **custom error message** that describes what went wrong.
- Example: if a student enters marks = -10 → program throws InvalidMarksException("Invalid marks for subject 1: -10");

---

2. Student Class

Purpose:

To store and manage data of an individual student.

Attributes:

- rollNumber – Unique student ID (Integer)
- studentName – Student's name (String)
- marks[] – Array to store marks for 3 subjects

Important Methods:

a. validateMarks()

- Checks if each mark is between **0 and 100**.
- If not, it **throws** an InvalidMarksException.

. ResultManager Class

Purpose:

Acts as the **main driver class** that:

- Provides the **menu-driven interface**,

- Handles user input,
- Stores multiple students,
- Catches and handles exceptions properly.

Attributes:

- students[] – Array to store up to 10 students.
- count – To keep track of the number of students added.
- Scanner sc – For user input.

Important Methods:

a. addStudent()

- Prompts the user to enter roll number, name, and marks.
- Uses try-catch to handle:
    - **InvalidMarksException** (custom)
    - **InputMismatchException** (if input type is wrong)
    - **General Exception**
- If data is valid, creates a new Student object and adds it to the array.

Demonstrates:
throw (inside Student class)
throws (in Student constructor)
try-catch (inside ResultManager)

b. showStudentDetails()

- Asks the user for a roll number.
- If not found → prints a message.
- Uses try-catch for invalid