ASSIGNMENT-3

Name – Nandini Kumari

Roll no.- 2401201085

Course- BCA(AI&DS)

Input-

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


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


class Student {

    private Integer rollNumber;
    private String studentName;
    private Integer[] marks = new Integer[3];

    public Student(Integer rollNumber, String studentName, Integer[] marks)
            throws InvalidMarksException {

        this.rollNumber = rollNumber;
        this.studentName = studentName;
        this.marks = marks;

        validateMarks();
    }


```

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

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

    public String getResultStatus() {
        for (int m : marks) {
            if (m < 35) return "Fail";
        }
        return "Pass";
    }

    public Integer getRollNumber() {
        return rollNumber;
    }
    }

    public void displayResult() {
        System.out.println("Roll Number : " + rollNumber);
        System.out.println("Student Name: " + studentName);

        System.out.print(s: "Marks       : ");
        for (int m : marks) {
            System.out.print(m + " ");
        }

        System.out.println("\nAverage     : " + calculateAverage());
        System.out.println("Result      : " + getResultStatus());
    }
}


public class ResultManager {

    private Student[] students = new Student[50];
    private int count = 0;
    private Scanner sc = new Scanner(System.in);


    public void addStudent() {
        try {
```

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

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

            Integer[] marks = new Integer[3];

            for (int i = 0; i < 3; i++) {
                System.out.print("Enter marks for subject " + (i + 1) + ": ");
                marks[i] = sc.nextInt();
            }


            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 numeric values.");
            sc.nextLine();
        } catch (Exception e) {
            System.out.println("Unexpected error: " + e.getMessage());
        }
    }


    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(x: "Student not found.");
            }

        } catch (InputMismatchException e) {
            System.out.println(x: "Error: Invalid roll number format.");
```

```
            System.out.println("Error: Invalid roll number format.");
            sc.nextLine();
        } catch (Exception e) {
            System.out.println("Unexpected error: " + e.getMessage());
        }
    }

    public void mainMenu() {

        try {
            int choice;

            do {
                System.out.println("\n===== 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. Try again.");
                }

            } while (choice != 3);

        } finally {
            sc.close();
            System.out.println("Scanner closed. Program terminated.");
        }
    }

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

```
===== Student Result Management System =====
1. Add Student
2. Show Student Details
3. Exit
Enter your choice: 1
Enter Roll Number: 85
Enter Student Name: Nandini Kumari
Enter marks for subject 1: 80
Enter marks for subject 2: 75
Enter marks for subject 3: 65
Student added successfully. Returning to main menu...
```

```
===== Student Result Management System =====
1. Add Student
2. Show Student Details
3. Exit
Enter your choice: 2
Enter Roll Number to search: 85
Roll Number : 85
Student Name: Nandini Kumari
Marks       : 80 75 65
Average     : 73.33333333333333
Result      : Pass
```

```
===== Student Result Management System =====
1. Add Student
2. Show Student Details
3. Exit
Enter your choice: 3
Exiting program. Thank you!
Scanner closed. Program terminated.
```

EXPLANATNON

Short Explanation of the Student Result Management System
This program is a simple Java application that stores student details, validates their marks, calculates their average, and displays whether they passed or failed.
It uses three main components:

---

1. Custom Exception (InvalidMarksException)
A custom checked exception used to detect invalid marks (marks below 0 or above 100).
If invalid marks are entered, the program throws this exception and stops adding the student.

---

2. Student Class
This class stores:
Roll number
Student name
Marks for 3 subjects
It also:
Validates marks using the custom exception
Calculates the average
Determines pass/fail
Displays all student details

---

3. ResultManager (Main Menu System)
This class:
Stores multiple students in an array
Provides a menu with 3 choices:
Add Student
Show Student Details
Exit
Handles errors like invalid input and invalid marks using try–catch blocks
Uses a finally block to close the Scanner

Overall Function
The program repeatedly shows a menu, takes user input, validates data, handles exceptions, and displays results. It demonstrates Java concepts like exception handling, custom exceptions, throw/throws, try–catch–finally, and object-oriented programming.