Assignment – 3

NAME – SHASHANK TYAGI

ROLL NO – 2401201033

COURSE- BCA(AI&DS)

CODE –

ResultManager.java 8

C: > Users > tyagi > OneDrive > Desktop > ResultManager.java > Java > InvalidMarksException

```java
import java.util.*;

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

class Student {
    private int roll;
    private String name;
    private int[] marks;

    public Student(int roll, String name, int[] marks) throws InvalidMarksException {
        this.roll = roll;
        this.name = name;
        this.marks = marks;
        validate();
    }

    private void validate() throws InvalidMarksException {
        if (name == null || name.trim().isEmpty())
            throw new IllegalArgumentException(s: "Name cannot be empty");
        if (marks.length != 3)
            throw new InvalidMarksException(msg: "Enter 3 marks");
        for (int m : marks)
            if (m < 0 || m > 100)
                throw new InvalidMarksException(msg: "Marks must be 0-100");
    }

    public int getRoll() { return roll; }

    public void display() {
        System.out.println("Roll: " + roll);
        System.out.println("Name: " + name);
        System.out.println("Marks: " + marks[0] + " " + marks[1] + " " + marks[2]);
        double avg = (marks[0] + marks[1] + marks[2]) / 3.0;
        System.out.println("Average: " + avg);
        System.out.println("Result: " + ((marks[0] >= 33 && marks[1] >= 33 && marks[2] >= 33) ? "Pass" : "Fail"));
```

ResultManager.java 8

C: > Users > tyagi > OneDrive > Desktop > ResultManager.java > Java > InvalidMarksException

```
class Student {
    public void display() {
    }
}

public class ResultManager {
    private Student[] list = new Student[100];
    private int count = 0;
    Scanner sc = new Scanner(System.in);

    private int readInt() {
        while (true) {
            try { return Integer.parseInt(sc.nextLine().trim()); }
            catch (Exception e) { System.out.print(s: "Enter valid number: "); }
        }
    }

    public void addStudent() throws InvalidMarksException {
        System.out.print(s: "Roll: ");
        int r = readInt();
        System.out.print(s: "Name: ");
        String n = sc.nextLine();
        int[] m = new int[3];
        for (int i = 0; i < 3; i++) {
            System.out.print("Marks " + (i + 1) + ": ");
            m[i] = readInt();
        }
        list[count++] = new Student(r, n, m);
        System.out.println(x: "Added.");
    }

    public void showStudent() {
        System.out.print(s: "Enter roll: ");
        int r = readInt();
        for (int i = 0; i < count; i++) {
```

ResultManager.java 8

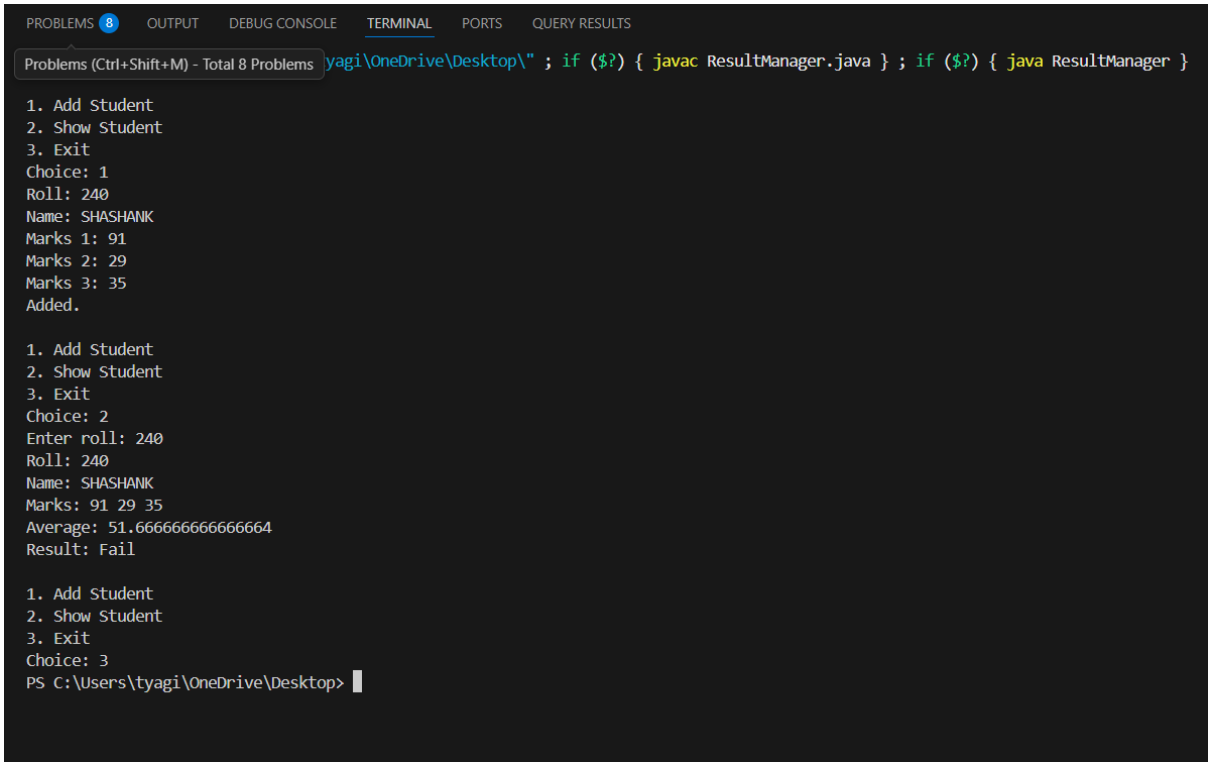
C: > Users > tyagi > OneDrive > Desktop > ResultManager.java > Java > InvalidMarksException

```
public class ResultManager {
    public void showStudent() {
        for (int i = 0; i < count; i++) {
            if (list[i].getRoll() == r) {
                list[i].display();
                return;
            }
        }
        System.out.println(x: "Not Found");
    }

    public void menu() {
        while (true) {
            System.out.println(x: "\n1. Add Student\n2. Show Student\n3. Exit");
            System.out.print(s: "Choice: ");
            int ch = readInt();
            try {
                if (ch == 1) addStudent();
                else if (ch == 2) showStudent();
                else if (ch == 3) return;
                else System.out.println(x: "Invalid choice");
            } catch (Exception e) {
                System.out.println(e.getMessage());
            }
        }
    }

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

OUTPUT –



EXPLANATION –

The program is a **Student Result Management System** that stores student details, validates input, and shows results with pass/fail status.

1. Custom Exception

A special exception class is created to handle invalid marks.
If marks are not between **0 and 100**, this exception is thrown.

2. Student Class

Represents one student and stores:

- Roll number
- Name
- Marks of 3 subjects

It also:

- Validates name and marks
- Calculates average
- Displays result (Pass if all marks ≥ 33)

3. ResultManager Class

Controls the whole program. It:

- Stores multiple students in an array
- Safely reads integer input
- Adds new students
- Searches and displays student details
- Shows a menu to choose actions
- Uses try–catch to prevent program crashes

4. Main Method

Starts the program by showing the menu.