

## JAVA ASSIGNMENT 3

NAME-BHAVYA RATTAN

ROLL NO-2401201004

COURSE-BCA(AI&DS

CODE:-

```
ResultManager.java  1 Welcome

J ResultManager.java > ...
1  import java.util.*;
2
3  // Custom Exception Class
4  class InvalidMarksException extends Exception {
5      public InvalidMarksException(String message) {
6          super(message);
7      }
8  }
9
10 // Student Class
11 class Student {
12     private int rollNumber;
13     private String studentName;
14     private int[] marks = new int[3];
15
16     public Student(int rollNumber, String studentName, int[] marks) throws InvalidMarksException {
17         this.rollNumber = rollNumber;
18         this.studentName = studentName;
19         this.marks = marks;
20         validateMarks();
21     }
22
23     // Validate each mark
24     public void validateMarks() throws InvalidMarksException {
25         for (int i = 0; i < marks.length; i++) {
26             if (marks[i] < 0 || marks[i] > 100) {
27                 throw new InvalidMarksException("Invalid marks for subject " + (i + 1) + ": " + marks[i]);
28             }
29         }
30     }
31
32     // Calculate average
33     public double calculateAverage() {
34         int total = 0;
35         for (int m : marks) {
36             total += m;
37         }
38         return total / 3.0;
39     }
40
41     // Display details
42     public void displayResult() {
43         double avg = calculateAverage();
44         System.out.println("Roll No: " + rollNumber + " Student Name: " + studentName + " Average: " + avg);
45     }
46 }
```

```
ResultManager.java > ...
11  class Student {
12      public void displayResult() {
13
14          System.out.println("Roll Number: " + rollNumber);
15          System.out.println("Student Name: " + studentName);
16          System.out.print(s: "Marks: ");
17          for (int m : marks)
18              System.out.print(m + " ");
19          System.out.println();
20          System.out.println("Average: " + avg);
21          System.out.println("Result: " + status);
22      }
23
24
25      public int getRollNumber() {
26          return rollNumber;
27      }
28  }
29
30
31 // ResultManager Class
32 public class ResultManager {
33     private Student[] students;
34     private int count;
35     private Scanner sc;
36
37     public ResultManager() {
38         students = new Student[10]; // can store up to 10 students
39         count = 0;
40         sc = new Scanner(System.in);
41     }
42
43     // Add student details
44     public void addStudent() {
45         try {
46             System.out.print(s: "Enter Roll Number: ");
47             int roll = sc.nextInt();
48             sc.nextLine(); // consume newline
49
50             System.out.print(s: "Enter Student Name: ");
51             String name = sc.nextLine();
52
53             int[] marks = new int[3];
54             for (int i = 0; i < 3; i++) {
55                 System.out.print("Enter marks for subject " + (i + 1) + ": ");
56                 marks[i] = sc.nextInt();
57             }
58
59             Student student = new Student();
60             student.rollNumber = roll;
61             student.studentName = name;
62             student.marks = marks;
63             student.avg = calculateAverage(marks);
64             student.status = determineStatus(student.avg);
65             students[count] = student;
66             count++;
67         } catch (Exception e) {
68             e.printStackTrace();
69         }
70     }
71
72     public void displayResults() {
73         for (Student student : students) {
74             student.displayResult();
75         }
76     }
77
78     public static void main(String[] args) {
79         ResultManager manager = new ResultManager();
80         manager.addStudent();
81         manager.addStudent();
82         manager.addStudent();
83         manager.addStudent();
84         manager.addStudent();
85         manager.addStudent();
86         manager.addStudent();
87         manager.addStudent();
88         manager.addStudent();
89         manager.addStudent();
90     }
91 }
```

```
J ResultManager.java > ...
62 √ public class ResultManager {
74 √     public void addStudent() {
87     }
88
89         // Validate and create Student
90         Student s = new Student(roll, name, marks);
91         students[count++] = s;
92
93         System.out.println(x: "Student added successfully. Returning to main menu...");
94
95     } catch (InvalidMarksException e) {
96         System.out.println("Error: " + e.getMessage());
97     } catch (InputMismatchException e) {
98         System.out.println(x: "Error: Invalid input type. Please enter correct data.");
99         sc.nextLine(); // clear buffer
100    } catch (Exception e) {
101        System.out.println("Unexpected error: " + e.getMessage());
102    }
103}
104
105 // Show details of a specific student
106 public void showStudentDetails() {
107     try {
108         System.out.print(s: "Enter Roll Number to search: ");
109         int roll = sc.nextInt();
110
111         boolean found = false;
112         for (int i = 0; i < count; i++) {
113             if (students[i].getRollNumber() == roll) {
114                 students[i].displayResult();
115                 found = true;
116                 break;
117             }
118         }
119
120         if (!found) {
121             System.out.println("Student with Roll Number " + roll + " not found.");
122         }
123
124     } catch (InputMismatchException e) {
125         System.out.println(x: "Error: Invalid roll number input.");
126         sc.nextLine();
127     } catch (Exception e) {
128         System.out.println("Error: " + e.getMessage());
```

```

106     public void showStudentDetails() {
107         System.out.println("Error: " + e.getMessage());
108     }
109 }
110
111 // Main Menu
112 public void mainMenu() {
113     int choice = 0;
114     try {
115         do {
116             System.out.println(x: "===== Student Result Management System =====");
117             System.out.println(x: "1. Add Student");
118             System.out.println(x: "2. Show Student Details");
119             System.out.println(x: "3. Exit");
120             System.out.print(s: "Enter your choice: ");
121
122             choice = sc.nextInt();
123
124             switch (choice) {
125                 case 1:
126                     addStudent();
127                     break;
128                 case 2:
129                     showStudentDetails();
130                     break;
131                 case 3:
132                     System.out.println(x: "Exiting program. Thank you!");
133                     break;
134                 default:
135                     System.out.println(x: "Invalid choice! Please try again.");
136             }
137
138         } while (choice != 3);
139     } catch (InputMismatchException e) {
140         System.out.println(x: "Error: Please enter a valid number choice.");
141     } finally {
142         // Resource cleanup
143         System.out.println(x: "Closing resources...");
144         sc.close();
145         System.out.println(x: "Program terminated safely.");
146     }
147 }
148
149 }
150
151 // Main method
152 public static void main(String[] args) {
153     ResultManager manager = new ResultManager();
154     manager.mainMenu();
155 }
156
157 }
158
159 }
160
161 }
162
163 }
164
165 }
166
167 }
168
169 }
170
171 }
172
173 }
174
175 }
176

```

OUTPUT:-

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS
Exiting program. Thank you!
Exiting program. Thank you!
Closing resources...
PS C:\Users\DELL\OneDrive\ドキュメント\cpp> cd 'c:\Users\DELL\OneDrive\ドキュメント' & -cp 'C:\Users\DELL\AppData\Roaming\Code\User\workspaceStorage\aa05c95c71690b9d'
===== Student Result Management System =====
1. Add Student
2. Show Student Details
3. Exit
Enter your choice: 1
Enter Roll Number: 04
Enter Student Name: Bhavya
Enter marks for subject 1: 48
Enter marks for subject 2: 67
Enter marks for subject 3: 89
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: 04
Roll Number: 4
Student Name: Bhavya
Marks: 48 67 89
Average: 68.0
Result: Pass
===== Student Result Management System =====
1. Add Student
2. Show Student Details
3. Exit
Enter your choice: 3
Exiting program. Thank you!
Closing resources...
Program terminated safely.
```

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);  
    }  
}
```

- This 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