K.R. Mangalam University

Java programming

Assignment-3

Submitted by: SOPHIA SONI

Roll No: 2401201042

Course: BCA sp(AI&DS)

Section B

Submitted to: Dr. Manish Kumar

Problem Statement

Enhance the Student Management System by implementing exception handling and multithreading to ensure safe execution and responsiveness. The system should handle invalid input (such as marks outside the valid range or empty fields) using try-catch-finally blocks and custom exceptions like StudentNotFoundException. Additionally, the system should simulate a loading process when adding or saving student data by using multithreading. The program should utilize wrapper classes (such as Integer, Double) for data conversion and autoboxing where applicable, providing a robust and responsive user interface for managing student records.

CODE:

Welcome | ResultManager.java 9+

ResultManager.java > ResultManager.java > ...

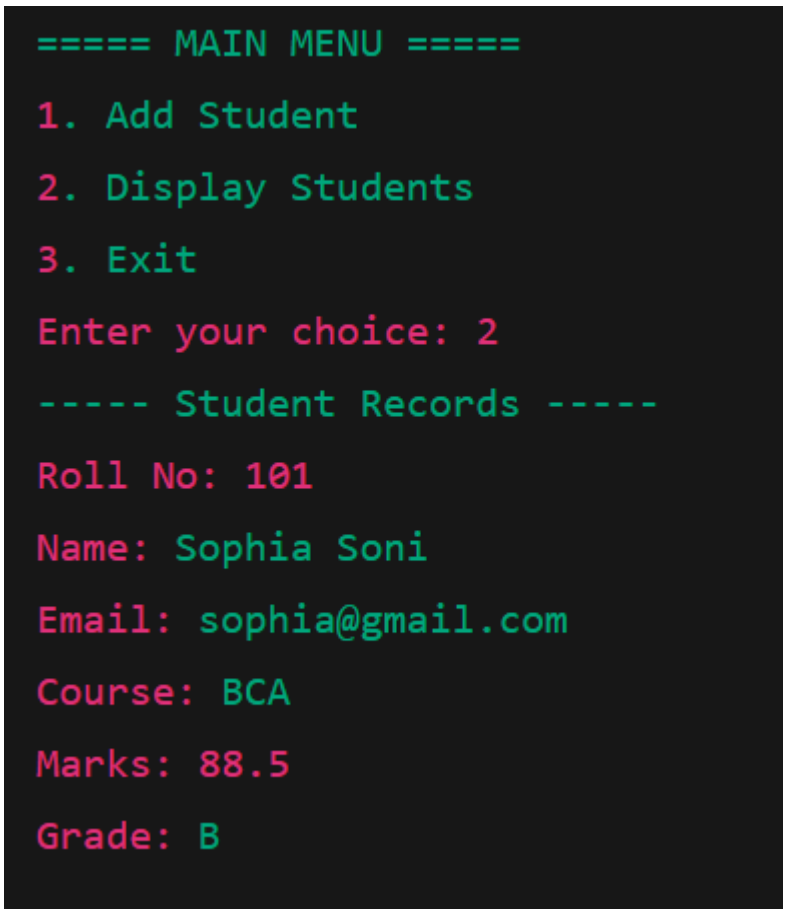
```java
// ================================================================
// Program: ResultManager.java
// Author: Sophia Soni
// Email: sophia@gmail.com
// College: K.R. Mangalam University
// Description: Student Management System with Exception Handling,
//              Multithreading & Wrapper Classes
// ================================================================

import java.util.*;

// Custom Exception for missing students
class StudentNotFoundException extends Exception {
    public StudentNotFoundException(String message) {
        super(message);
    }
}

// Interface for defining record actions
interface RecordActions {
    void addStudent();
    void displayStudents();
}

// Loader class to simulate loading (Multithreading)
class Loader implements Runnable {
    private String task;

    Loader(String task) {
```

```
        this.task = task;
    }

    @Override
    public void run() {
        System.out.print(task);
        try {
            for (int i = 0; i < 5; i++) {
                System.out.print(s: ".");
                Thread.sleep(millis: 500);
            }
        } catch (InterruptedException e) {
            System.out.println(x: "\nLoading interrupted!");
        }
        System.out.println(x: "\nLoading completed!\n");
    }
}

// Student class using wrapper classes (Integer, Double)
class Student {
    private Integer rollNo;
    private String name;
    private String email;
    private String course;
    private Double marks;

    public Student(Integer rollNo, String name, String email, String course, Double marks) {
        this.rollNo = rollNo;
        this.name = name;
        this.email = email;
        this.course = course;
        this.marks = marks;
    }

    public String calculateGrade() {
        if (marks >= 90) return "A";
        else if (marks >= 75) return "B";
        else if (marks >= 60) return "C";
        else if (marks >= 40) return "D";
        else return "F";
    }

    public void display() {
        System.out.println("Roll No: " + rollNo);
        System.out.println("Name: " + name);
        System.out.println("Email: " + email);
        System.out.println("Course: " + course);
        System.out.println("Marks: " + marks);
        System.out.println("Grade: " + calculateGrade());
    }
}

```

```java

// Manager class implementing interface
class StudentManager implements RecordActions {
    private List<Student> students = new ArrayList<>();
    private Scanner sc = new Scanner(System.in);

    @Override
    public void addStudent() {
        try {
            System.out.print(s: "Enter Roll No (Integer): ");
            Integer rollNo = Integer.parseInt(sc.nextLine());

            System.out.print(s: "Enter Name: ");
            String name = sc.nextLine();
            if (name.isEmpty()) throw new IllegalArgumentException(s: "Name cannot be empty!");

            System.out.print(s: "Enter Email: ");
            String email = sc.nextLine();
            if (email.isEmpty()) throw new IllegalArgumentException(s: "Email cannot be empty!");

            System.out.print(s: "Enter Course: ");
            String course = sc.nextLine();
            if (course.isEmpty()) throw new IllegalArgumentException(s: "Course cannot be empty!");

            System.out.print(s: "Enter Marks: ");
            Double marks = Double.parseDouble(sc.nextLine());
            if (marks < 0 || marks > 100)
                throw new IllegalArgumentException(s: "Marks must be between 0 and 100!");

            // Simulate loading with thread
            Thread loaderThread = new Thread(new Loader(task: "Loading"));
            loaderThread.start();
            loaderThread.join();

            students.add(new Student(rollNo, name, email, course, marks));
            System.out.println(x: "Student added successfully!\n");
        } catch (NumberFormatException e) {
            System.out.println(x: "Invalid input format! Please enter correct numeric values.");
        } catch (IllegalArgumentException e) {
            System.out.println("Error: " + e.getMessage());
        } catch (InterruptedException e) {
            System.out.println(x: "Thread interrupted during loading.");
        } finally {
            System.out.println(x: "Data input process completed.\n");
        }
    }

    @Override
    public void displayStudents() {
        try {
            if (students.isEmpty())
                throw new StudentNotFoundException(message: "No student records found!");
```

```
133
134             System.out.println(x: "----- Student Records -----");
135             for (Student s : students) {
136                 s.display();
137                 System.out.println();
138             }
139         } catch (StudentNotFoundException e) {
140             System.out.println(e.getMessage());
141         }
142     }
143 }
144
145 // Main class – matches file name
146 public class ResultManager {
        Run main | Debug main | Run | Debug
147     public static void main(String[] args) {
148         System.out.println(x: "================================================");
149         System.out.println(x: "        STUDENT MANAGEMENT SYSTEM - RESULT MANAGER");
150         System.out.println(x: "        Developed by: Sophia Soni");
151         System.out.println(x: "        Email: sophia@gmail.com");
152         System.out.println(x: "        K.R. Mangalam University");
153         System.out.println(x: "================================================\n");
154
155         StudentManager manager = new StudentManager();
156         Scanner sc = new Scanner(System.in);
157
158         while (true) {
159             System.out.println(x: "===== MAIN MENU =====");
160             System.out.println(x: "1. Add Student");
161             System.out.println(x: "2. Display Students");
162             System.out.println(x: "3. Exit");
163             System.out.print(s: "Enter your choice: ");
164
165             int choice;
166             try {
167                 choice = Integer.parseInt(sc.nextLine());
168             } catch (NumberFormatException e) {
169                 System.out.println(x: "Please enter a valid number!\n");
170                 continue;
171             }
172
173             switch (choice) {
174                 case 1:
175                     manager.addStudent();
176                     break;
177                 case 2:
178                     manager.displayStudents();
179                     break;
180                 case 3:
181                     System.out.println(x: "Exiting system... Goodbye!");
182                     return;
183                 default:
183                 default:
184                     System.out.println(x: "Invalid choice! Please try again.\n");
185             }
186         }
187     }
188 }
```

OUTPUT:


===== MAIN MENU =====
1. Add Student
2. Display Students
3. Exit
Enter your choice: 2
----- Student Records -----
Roll No: 101
Name: Sophia Soni
Email: sophia@gmail.com
Course: BCA
Marks: 88.5
Grade: B