Java Day 12

Task - 4

Advanced JAVA

Q1. Ramesh is developing a student management system for a university. In this system, you have a Student class to represent student information. You are asked to help Ramesh to handle exception which can be occurred into program according to following Scenarios:

class Student with attributes roll no, name, age and course. Initialize values through parameterized constructors.

If the age of the student is not between 15 and 21 then generate a user-defined exception "AgeNotWithinRangeException".

If a name contains numbers or special symbols, raise exception "NameNotValidException". Define the two exception classes.

Input -

package task.four.one;

class AgeNotWithinRangeException extends Exception {

public AgeNotWithinRangeException(String message) {

super(message);

}

}

class NameNotValidException extends Exception {

public NameNotValidException(String message) {

super(message);

}

}

class Student {

private int rollNo;

private String name;

private int age;

private String course;

public Student(int rollNo, String name, int age, String course)

throws AgeNotWithinRangeException, NameNotValidException {

this.rollNo = rollNo;

if (!name.matches("[a-zA-Z\\s]+")) {

throw new NameNotValidException("Name must not contain numbers or special characters.");

}

this.name = name;

if (age < 15 || age > 21) {

throw new AgeNotWithinRangeException("Age must be between 15 and 21.");

}

this.age = age;

this.course = course;

}

public void display() {

System.*out*.println("Roll No: " + rollNo);

System.*out*.println("Name: " + name);

System.*out*.println("Age: " + age);

System.*out*.println("Course: " + course);

}

}

public class StudentManagement {

public static void main(String[] args) {

try {

Student student = new Student(101, "Suna", 21, "Science");

student.display();

} catch (AgeNotWithinRangeException | NameNotValidException e) {

System.*out*.println("Exception: " + e.getMessage());

}

try {

Student student2 = new Student(102, "Ramya", 22, "Management");

student2.display();

} catch (AgeNotWithinRangeException | NameNotValidException e) {

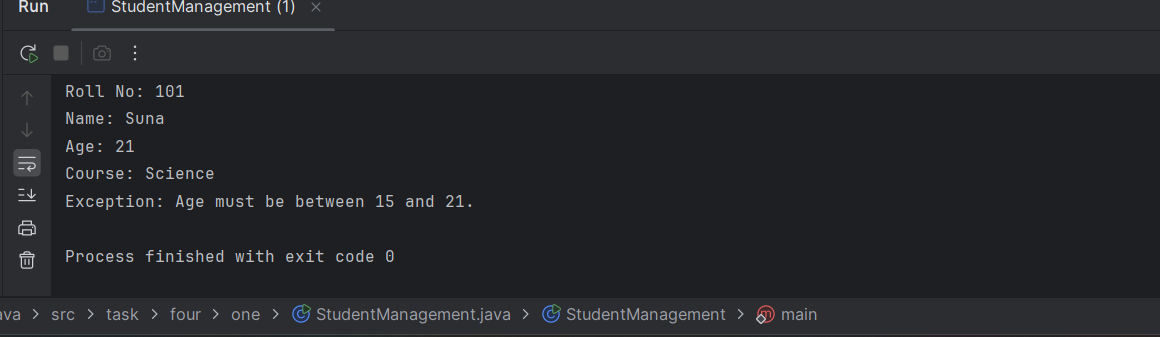
System.*out*.println("Exception: " + e.getMessage());

}

}

}

Output -



Q2. Create a class Voter(voterId, name, age) with parameterized constructor. The parameterized constructor should throw a checked/Unchecked exception if age is less than 18. The message of exception is "invalid age for voter "

Input–

package task.four.two;

class InvalidVoterAgeException extends RuntimeException {

public InvalidVoterAgeException(String message) {

super(message);

}

}

class Voter {

private int voterId;

private String name;

private int age;

public Voter(int voterId, String name, int age) {

if (age < 18) {

throw new InvalidVoterAgeException("invalid age for voter");

}

this.voterId = voterId;

this.name = name;

this.age = age;

}

public void display() {

System.*out*.println("Voter ID: " + voterId);

System.*out*.println("Name: " + name);

System.*out*.println("Age: " + age);

}

}

public class VoterMain {

public static void main(String[] args) {

try {

Voter v1 = new Voter(101, "Gil", 34);

v1.display();

Voter v2 = new Voter(102, "Jay", 22);

v2.display();

} catch (InvalidVoterAgeException e) {

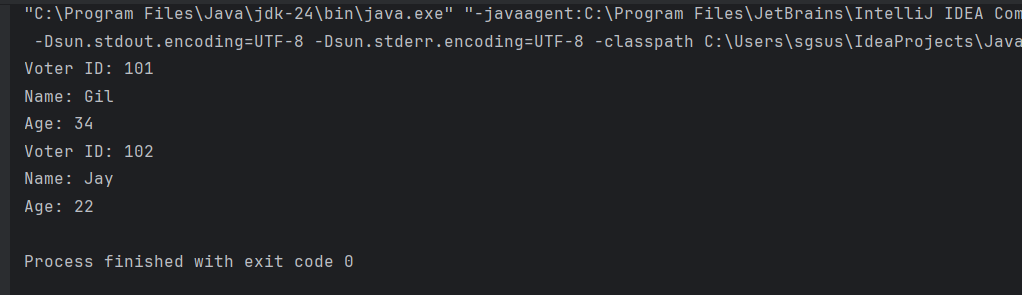
System.*out*.println("Exception: " + e.getMessage());

}

}

}

Output -



Q3. Store name of weekdays in an array (starting from "Sunday" at 0 index). Ask day position from user and print day name. Handle array index out of bound exception and give proper message if user enters day index outside range (0-6).

Input -

package task.four.three;

import java.util.Scanner;

public class WeekdayFinder {

public static void main(String[] args) {

String[] weekdays = {"Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"};

Scanner scanner = new Scanner(System.*in*);

System.*out*.print("Enter the day index (0-6): ");

try {

int index = scanner.nextInt();

System.*out*.println("Day: " + weekdays[index]);

} catch (ArrayIndexOutOfBoundsException e) {

System.*out*.println("Invalid index");

} catch (Exception e) {

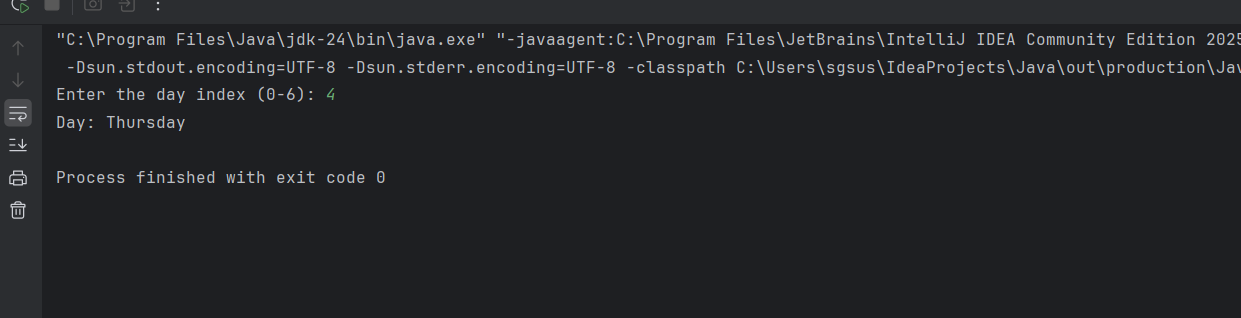
System.*out*.println("Kindly Enter a valid integer");

}

scanner.close();

}

}

Output-   
  


Q4. Create a HashMap where keys are student names (strings) and values are their corresponding grades (integers). Create methods to add a new student, remove a student, and Display up a student's grade by name.

Input -

package task.four.four;

import java.util.HashMap;

import java.util.Scanner;

public class StudentGrades {

private HashMap<String, Integer> studentMap = new HashMap<>();

public void addStudent(String name, int grade) {

studentMap.put(name, grade);

System.*out*.println("Student added: " + name + " -> " + grade);

}

public void removeStudent(String name) {

if (studentMap.containsKey(name)) {

studentMap.remove(name);

System.*out*.println("Removed student: " + name);

} else {

System.*out*.println("Student not found.");

}

}

public void displayGrade(String name) {

if (studentMap.containsKey(name)) {

System.*out*.println(name + "'s grade: " + studentMap.get(name));

} else {

System.*out*.println("Student not found.");

}

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.*in*);

StudentGrades sg = new StudentGrades();

while (true) {

System.*out*.println("\n--- Menu ---");

System.*out*.println("1. Add Student");

System.*out*.println("2. Remove Student");

System.*out*.println("3. Display Student Grade");

System.*out*.println("4. Exit");

System.*out*.print("Choose option: ");

int choice = scanner.nextInt();

scanner.nextLine();

switch (choice) {

case 1:

System.*out*.print("Enter student name: ");

String nameToAdd = scanner.nextLine();

System.*out*.print("Enter grade: ");

int grade = scanner.nextInt();

sg.addStudent(nameToAdd, grade);

break;

case 2:

System.*out*.print("Enter student name to remove: ");

String nameToRemove = scanner.nextLine();

sg.removeStudent(nameToRemove);

break;

case 3:

System.*out*.print("Enter student name to display: ");

String nameToDisplay = scanner.nextLine();

sg.displayGrade(nameToDisplay);

break;

case 4:

System.*out*.println("Exiting...");

scanner.close();

return;

default:

System.*out*.println("Invalid option.");

}

}

}

}

Output -

"C:\Program Files\Java\jdk-24\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2025.1.2\lib\idea\_rt.jar=56560" -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8 -classpath C:\Users\sgsus\IdeaProjects\Java\out\production\Java task.four.four.StudentGrades

--- Menu ---

1. Add Student

2. Remove Student

3. Display Student Grade

4. Exit

Choose option: 1

Enter student name: Sush

Enter grade: 10

Student added: Sush -> 10

--- Menu ---

1. Add Student

2. Remove Student

3. Display Student Grade

4. Exit

Choose option: 1

Enter student name: Ramya

Enter grade: 5

Student added: Ramya -> 5

--- Menu ---

1. Add Student

2. Remove Student

3. Display Student Grade

4. Exit

Choose option: 1

Enter student name: Chinnu

Enter grade: 7

Student added: Chinnu -> 7

--- Menu ---

1. Add Student

2. Remove Student

3. Display Student Grade

4. Exit

Choose option: 2

Enter student name to remove: Chinnu

Removed student: Chinnu

--- Menu ---

1. Add Student

2. Remove Student

3. Display Student Grade

4. Exit

Choose option: 3

Enter student name to display: Chinnu

Student not found.

--- Menu ---

1. Add Student

2. Remove Student

3. Display Student Grade

4. Exit

Choose option:

Q5. Use Collection Classes to store Integers.Create some methods for following functionalities.

a. Include functions for pushing elements onto the stack.

b. popping elements from the stack.

Input -   
  
package task.four.five;

import java.util.Stack;

public class IntegerStack {

private Stack<Integer> stack = new Stack<>();

public void pushElement(int element) {

stack.push(element);

System.*out*.println("Pushed: " + element);

}

public void popElement() {

if (stack.isEmpty()) {

System.*out*.println("Stack is empty. Cannot pop.");

} else {

int popped = stack.pop();

System.*out*.println("Popped: " + popped);

}

}

public void displayStack() {

System.*out*.println("Current Stack: " + stack);

}

public static void main(String[] args) {

IntegerStack intStack = new IntegerStack();

intStack.pushElement(10);

intStack.pushElement(20);

intStack.pushElement(30);

intStack.displayStack();

intStack.popElement();

intStack.displayStack();

intStack.popElement();

intStack.popElement();

intStack.popElement();

}

}

Output-

