9)

// File: CIE/Internals.java

package CIE;

public class Internals extends Student {

public int[] internalMarks = new int[6];

public Internals(String usn, String name, int sem, int[] internalMarks) {

this.usn = usn;

this.name = name;

this.sem = sem;

this.internalMarks = internalMarks;

}

}

// File: CIE/Student.java

package CIE;

public class Student {

public String usn;

public String name;

public int sem;

}

// File: SEE/External.java

package SEE;

import CIE.Student;

public class External extends Student {

public int[] seeMarks = new int[6];

public External(String usn, String name, int sem, int[] seeMarks) {

this.usn = usn;

this.name = name;

this.sem = sem;

this.seeMarks = seeMarks;

}

}

// File: Main.java

import CIE.Internals;

import SEE.External;

public class Main {

public static void main(String[] args) {

int N = 5; // Example number of students

Internals[] internalStudents = new Internals[N];

External[] externalStudents = new External[N];

// Example data

for (int i = 0; i < N; i++) {

internalStudents[i] = new Internals("USN" + (i + 1), "Student" + (i + 1), 3, new int[]{80, 85, 75, 90, 88, 92});

externalStudents[i] = new External("USN" + (i + 1), "Student" + (i + 1), 3, new int[]{70, 75, 65, 80, 78, 82});

}

for (int i = 0; i < N; i++) {

System.out.println("Student: " + internalStudents[i].name);

System.out.println("USN: " + internalStudents[i].usn);

System.out.println("Semester: " + internalStudents[i].sem);

int totalMarks = 0;

for (int j = 0; j < 6; j++) {

int finalMarks = internalStudents[i].internalMarks[j] + externalStudents[i].seeMarks[j];

totalMarks += finalMarks;

System.out.println("Course " + (j + 1) + " Final Marks: " + finalMarks);

}

System.out.println("Total Marks: " + totalMarks + "\n");

}

}

}

10)

import java.util.Random;

class GenerateNumber implements Runnable {

public void run() {

Random random = new Random();

while (true) {

int number = random.nextInt(100);

System.out.println("Generated Number: " + number);

if (number % 2 == 0) {

new Thread(new SquareNumber(number)).start();

} else {

new Thread(new CubeNumber(number)).start();

}

try {

Thread.sleep(1000); // Sleep for 1 second

} catch (InterruptedException e) {

System.out.println(e);

}

}

}

}

class SquareNumber implements Runnable {

private int number;

SquareNumber(int number) {

this.number = number;

}

public void run() {

System.out.println("Square of " + number + " is " + (number \* number));

}

}

class CubeNumber implements Runnable {

private int number;

CubeNumber(int number) {

this.number = number;

}

public void run() {

System.out.println("Cube of " + number + " is " + (number \* number \* number));

}

}

public class MultiThreadedApp {

public static void main(String[] args) {

Thread generateThread = new Thread(new GenerateNumber());

generateThread.start();

}}

11)

import java.util.Scanner;

public class ExceptionHandling {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.println("Choose an exception to handle:");

System.out.println("1. ArithmeticException");

System.out.println("2. ArrayIndexOutOfBoundsException");

System.out.println("3. NumberFormatException");

System.out.println("4. StringIndexOutOfBoundsException");

System.out.println("5. NullPointerException");

int choice = scanner.nextInt();

switch(choice) {

case 1:

try {

System.out.println("Enter numerator and denominator:");

int numerator = scanner.nextInt();

int denominator = scanner.nextInt();

int result = numerator / denominator;

System.out.println("Result: " + result);

} catch (ArithmeticException e) {

System.out.println("Error: Division by zero is not allowed.");

}

break;

case 2:

try {

int[] array = {1, 2, 3};

System.out.println("Enter index:");

int index = scanner.nextInt();

System.out.println("Element at index " + index + ": " + array[index]);

} catch (ArrayIndexOutOfBoundsException e) {

System.out.println("Error: Array index is out of bounds.");

}

break;

case 3:

try {

System.out.println("Enter a number:");

String input = scanner.next();

int number = Integer.parseInt(input);

System.out.println("Number: " + number);

} catch (NumberFormatException e) {

System.out.println("Error: Invalid number format.");

}

break;

case 4:

try {

System.out.println("Enter a string:");

String str = scanner.next();

System.out.println("Enter index:");

int index = scanner.nextInt();

char ch = str.charAt(index);

System.out.println("Character at index " + index + ": " + ch);

} catch (StringIndexOutOfBoundsException e) {

System.out.println("Error: String index is out of bounds.");

}

break;

case 5:

try {

String str = null;

System.out.println("Length of the string: " + str.length());

} catch (NullPointerException e) {

System.out.println("Error: Null pointer exception.");

}

break;

default:

System.out.println("Invalid choice.");

}

scanner.close();

}

}

12)

import java.util.Arrays;

class Sort {

// Generic method to arrange (sort) the array

public <T extends Comparable<T>> void Arrange(T[] array) {

Arrays.sort(array);

}

// Generic method to display the array

public <T> void Display(T[] array) {

for (T element : array) {

System.out.print(element + " ");

}

System.out.println();

}

}

public class GenericSortExample {

public static void main(String[] args) {

Sort sorter = new Sort();

// Integer array

Integer[] intArray = {5, 3, 8, 1, 9};

System.out.println("Original Integer Array: ");

sorter.Display(intArray);

sorter.Arrange(intArray);

System.out.println("Sorted Integer Array: ");

sorter.Display(intArray);

// String array

String[] strArray = {"Banana", "Apple", "Cherry", "Date"};

System.out.println("\nOriginal String Array: ");

sorter.Display(strArray);

sorter.Arrange(strArray);

System.out.println("Sorted String Array: ");

sorter.Display(strArray);

// Double array

Double[] doubleArray = {2.5, 3.7, 1.2, 4.8, 0.9};

System.out.println("\nOriginal Double Array: ");

sorter.Display(doubleArray);

sorter.Arrange(doubleArray);

System.out.println("Sorted Double Array: ");

sorter.Display(doubleArray);

}

}