**7. Packages: Student, Sports, and Report**

For this program, you need to create a specific folder structure.

1. Create a main project folder.
2. Inside it, create three sub-folders: studentpackage, sportspackage, and reportpackage.
3. Save each file in its corresponding folder.

**File 1: studentpackage/Student.java**

package studentpackage;

public class Student {

private String name;

private int rollNumber;

public Student(String name, int rollNumber) {

this.name = name;

this.rollNumber = rollNumber;

}

public String getName() {

return name;

}

public int getRollNumber() {

return rollNumber;

}

}

**File 2: sportspackage/Sports.java**

package sportspackage;

public interface Sports {

String getSportName();

int getSportScore();

}

**File 3: reportpackage/Report.java**

package reportpackage;

import studentpackage.Student;

import sportspackage.Sports;

public class Report implements Sports {

private Student student;

private String sportName;

private int sportScore;

public Report(Student student, String sportName, int sportScore) {

this.student = student;

this.sportName = sportName;

this.sportScore = sportScore;

}

@Override

public String getSportName() {

return sportName;

}

@Override

public int getSportScore() {

return sportScore;

}

public void generateReport() {

System.out.println("----- Student Report -----");

System.out.println("Name : " + student.getName());

System.out.println("Roll No. : " + student.getRollNumber());

System.out.println("Sport : " + getSportName());

System.out.println("Score : " + getSportScore());

}

public static void main(String[] args) {

Student student = new Student("Alice", 101);

Report report = new Report(student, "Basketball", 85);

report.generateReport();

}

}

**8.A) Integer Division with Exception Handling**

**File Name: IntegerDivision.java**

import java.util.Scanner;

public class IntegerDivision {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

try {

System.out.print("Enter first number (Num1): ");

String input1 = scanner.nextLine();

System.out.print("Enter second number (Num2): ");

String input2 = scanner.nextLine();

int num1 = Integer.parseInt(input1);

int num2 = Integer.parseInt(input2);

int result = num1 / num2;

System.out.println("Result of " + num1 + " / " + num2 + " = " + result);

} catch (NumberFormatException e) {

System.out.println("Error: Please enter valid integers only.");

} catch (ArithmeticException e) {

System.out.println("Error: Cannot divide by zero.");

} finally {

scanner.close();

}

}

}

**8.B) User-Defined Exception**

**File Name: User.java**

import java.util.\*;

class MyExp extends Exception {

public MyExp(String str) {

super(str);

}

}

public class User {

public static void main(String args[]) {

Scanner sc = new Scanner(System.in);

String n1, n2;

try {

System.out.print("Enter first number: ");

n1 = sc.nextLine();

int a = Integer.parseInt(n1);

System.out.print("Enter second number: ");

n2 = sc.nextLine();

int b = Integer.parseInt(n2);

if (b <= 0) {

throw new MyExp("Arithmetic Exception: Division by zero or negative number is not allowed.");

}

System.out.println("Division: " + (a / b));

} catch (NumberFormatException e) {

System.out.println("Error: Please enter only integers.");

} catch (MyExp e) {

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

} finally {

sc.close();

}

}

}

**9.A) Multithreading by Extending Thread**

**File Name: MultiThreadExample.java**

class Thread1 extends Thread {

public void run() {

for (int i = 1; i <= 6; i++) {

try {

Thread.sleep(1000);

} catch (InterruptedException e) {

System.out.println(e);

}

System.out.println("Good morning");

}

}

}

class Thread2 extends Thread {

public void run() {

for (int i = 1; i <= 6; i++) {

try {

Thread.sleep(2000);

} catch (InterruptedException e) {

System.out.println(e);

}

System.out.println("Hello");

}

}

}

class Thread3 extends Thread {

public void run() {

for (int i = 1; i <= 6; i++) {

try {

Thread.sleep(3000);

} catch (InterruptedException e) {

System.out.println(e);

}

System.out.println("Welcome");

}

}

}

public class MultiThreadExample {

public static void main(String[] args) {

Thread1 t1 = new Thread1();

Thread2 t2 = new Thread2();

Thread3 t3 = new Thread3();

t1.start();

t2.start();

t3.start();

}

}

**9.B) Thread Synchronization**

**File Name: SyncExample.java**

class Printer {

synchronized void print(String message) {

for (int i = 1; i <= 3; i++) {

System.out.println(message + " - " + i);

try {

Thread.sleep(500);

} catch (InterruptedException e) {

System.out.println(e);

}

}

}

}

class PrintJob extends Thread {

Printer printer;

String message;

PrintJob(Printer printer, String message) {

this.printer = printer;

this.message = message;

}

public void run() {

printer.print(message);

}

}

public class SyncExample {

public static void main(String[] args) {

Printer sharedPrinter = new Printer();

PrintJob t1 = new PrintJob(sharedPrinter, "Thread 1 printing");

PrintJob t2 = new PrintJob(sharedPrinter, "Thread 2 printing");

t1.start();

t2.start();

}

}

**10.A) Swing Registration Page**

**File Name: Registration.java**

import javax.swing.\*;

import java.awt.\*;

public class Registration {

public static void main(String[] args) {

JFrame frame = new JFrame("Registration Form");

frame.setSize(400, 300);

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.setLayout(new BorderLayout());

JPanel panel = new JPanel();

panel.setLayout(new GridLayout(6, 2, 10, 10));

JLabel nameLabel = new JLabel("Enter your name:");

JTextField nameField = new JTextField(20);

JLabel ageLabel = new JLabel("Enter your age:");

JTextField ageField = new JTextField(20);

JLabel emailLabel = new JLabel("Enter your email:");

JTextField emailField = new JTextField(20);

JLabel passwordLabel = new JLabel("Enter your password:");

JPasswordField passwordField = new JPasswordField(20);

JLabel confirmPasswordLabel = new JLabel("Confirm password:");

JPasswordField confirmPasswordField = new JPasswordField(20);

JButton submitButton = new JButton("Register");

panel.add(nameLabel);

panel.add(nameField);

panel.add(ageLabel);

panel.add(ageField);

panel.add(emailLabel);

panel.add(emailField);

panel.add(passwordLabel);

panel.add(passwordField);

panel.add(confirmPasswordLabel);

panel.add(confirmPasswordField);

panel.add(new JLabel());

panel.add(submitButton);

frame.add(panel, BorderLayout.CENTER);

frame.setVisible(true);

}

}

**10.B) Mouse Events with Adapter Classes**

**File Name: MouseEventDemo.java**

import java.awt.\*;

import java.awt.event.\*;

public class MouseEventDemo extends Frame {

Label label;

MouseEventDemo() {

label = new Label();

label.setBounds(20, 50, 100, 20);

add(label);

addMouseListener(new MouseAdapter() {

public void mouseClicked(MouseEvent e) {

label.setText("Mouse Clicked");

}

public void mouseEntered(MouseEvent e) {

label.setText("Mouse Entered");

}

public void mouseExited(MouseEvent e) {

label.setText("Mouse Exited");

}

public void mouseReleased(MouseEvent e) {

label.setText("Mouse Released");

}

});

setSize(300, 300);

setLayout(null);

setVisible(true);

}

public static void main(String[] args) {

new MouseEventDemo();

}

}

**12.A) Registration Validation**

**File Name: Registration3.java**

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class Registration3 extends JFrame implements ActionListener {

JLabel l1, l2, l3, l4;

JTextField tf1, tf2;

JButton btn1;

JPasswordField pf1, pf2;

Registration3() {

setVisible(true);

setSize(700, 700);

setLayout(new BorderLayout());

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setTitle("Registration Form");

JPanel panel = new JPanel();

panel.setLayout(new GridLayout(30, 30));

l1 = new JLabel("Name");

tf1 = new JTextField();

panel.add(l1);

panel.add(tf1);

l2 = new JLabel("Email");

tf2 = new JTextField();

panel.add(l2);

panel.add(tf2);

btn1 = new JButton("Submit");

btn1.addActionListener(this);

panel.add(btn1);

add(panel);

}

public void actionPerformed(ActionEvent e) {

String name = tf1.getText();

String email = tf2.getText();

if (name.isEmpty() || email.isEmpty()) {

JOptionPane.showMessageDialog(this, "Error: All fields must be filled.");

} else {

JOptionPane.showMessageDialog(this, "Success: All fields are filled!");

}

}

public static void main(String[] args) {

new Registration3();

}

}

**12.B) Color Sliders**

**File Name: ColorSlider.java**

import java.awt.\*;

import javax.swing.\*;

import javax.swing.event.\*;

public class ColorSlider extends JFrame implements ChangeListener {

private JSlider redSlider, greenSlider, blueSlider;

private Container contentPane;

public ColorSlider() {

super("Color Slider");

contentPane = getContentPane();

contentPane.setLayout(new FlowLayout());

redSlider = new JSlider(JSlider.HORIZONTAL, 0, 255, 0);

greenSlider = new JSlider(JSlider.HORIZONTAL, 0, 255, 0);

blueSlider = new JSlider(JSlider.HORIZONTAL, 0, 255, 0);

redSlider.addChangeListener(this);

greenSlider.addChangeListener(this);

blueSlider.addChangeListener(this);

contentPane.add(new JLabel("Red"));

contentPane.add(redSlider);

contentPane.add(new JLabel("Green"));

contentPane.add(greenSlider);

contentPane.add(new JLabel("Blue"));

contentPane.add(blueSlider);

setSize(300, 200);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setVisible(true);

}

public void stateChanged(ChangeEvent e) {

int r = redSlider.getValue();

int g = greenSlider.getValue();

int b = blueSlider.getValue();

Color color = new Color(r, g, b);

contentPane.setBackground(color);

}

public static void main(String[] args) {

new ColorSlider();

}

}

**13.A) Custom ArrayList**

**File Name: MyArrayListDemo.java**

class MyArrayList<T> {

private Object[] data;

private int size;

private static final int INITIAL\_CAPACITY = 5;

public MyArrayList() {

data = new Object[INITIAL\_CAPACITY];

size = 0;

}

public void add(T element) {

if (size == data.length) {

resize();

}

data[size++] = element;

}

public T get(int index) {

checkIndex(index);

return (T) data[index];

}

public void remove(int index) {

checkIndex(index);

for (int i = index; i < size - 1; i++) {

data[i] = data[i + 1];

}

data[size - 1] = null;

size--;

}

public int size() {

return size;

}

private void resize() {

int newCapacity = data.length \* 2;

Object[] newData = new Object[newCapacity];

for (int i = 0; i < data.length; i++) {

newData[i] = data[i];

}

data = newData;

}

private void checkIndex(int index) {

if (index < 0 || index >= size) {

throw new IndexOutOfBoundsException("Index " + index + " out of bounds");

}

}

}

public class MyArrayListDemo {

public static void main(String[] args) {

MyArrayList<Integer> list = new MyArrayList<>();

System.out.println("Adding elements:");

list.add(10);

list.add(20);

list.add(30);

list.add(40);

list.add(50);

list.add(60);

for (int i = 0; i < list.size(); i++) {

System.out.println("Element at index " + i + ": " + list.get(i));

}

System.out.println("\nRemoving element at index 2 (value: 30):");

list.remove(2);

System.out.println("\nElements after removal:");

for (int i = 0; i < list.size(); i++) {

System.out.println("Element at index " + i + ": " + list.get(i));

}

System.out.println("\nCurrent size of list: " + list.size());

}

}

**13.B) Employee HashMap**

**File Name: EmployeeDemo.java**

import java.util.HashMap;

import java.util.Scanner;

class Employee {

private int id;

private String name;

private String department;

private double salary;

public Employee(int id, String name, String department, double salary) {

this.id = id;

this.name = name;

this.department = department;

this.salary = salary;

}

public void setName(String name) {

this.name = name;

}

public void setDepartment(String department) {

this.department = department;

}

public void setSalary(double salary) {

this.salary = salary;

}

public int getId() {

return id;

}

public String getName() {

return name;

}

public String getDepartment() {

return department;

}

public double getSalary() {

return salary;

}

public void displayEmployee() {

System.out.println("ID: " + id);

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

System.out.println("Department: " + department);

System.out.println("Salary: $" + salary);

}

}

public class EmployeeDemo {

public static void main(String[] args) {

HashMap<Integer, Employee> employeeMap = new HashMap<>();

Scanner scanner = new Scanner(System.in);

System.out.print("How many employees you want to add? ");

int n = scanner.nextInt();

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

System.out.println("\nEnter details for Employee " + (i + 1) + ":");

System.out.print("ID: ");

int id = scanner.nextInt();

scanner.nextLine();

System.out.print("Name: ");

String name = scanner.nextLine();

System.out.print("Department: ");

String department = scanner.nextLine();

System.out.print("Salary: ");

double salary = scanner.nextDouble();

Employee emp = new Employee(id, name, department, salary);

employeeMap.put(id, emp);

}

System.out.print("\nEnter Employee ID to search: ");

int searchId = scanner.nextInt();

if (employeeMap.containsKey(searchId)) {

System.out.println("\nEmployee Found:");

employeeMap.get(searchId).displayEmployee();

} else {

System.out.println("Employee with ID " + searchId + " not found.");

}

scanner.close();

}

}

**14.A) File Info and Content Display**

**File Name: FileInfoDisplay.java**

import java.io.File;

import java.io.FileInputStream;

import java.io.IOException;

import java.util.Scanner;

public class FileInfoDisplay {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the file name with path: ");

String fileName = scanner.nextLine();

File file = new File(fileName);

if (file.exists()) {

System.out.println("File exists: Yes");

System.out.println("Readable: " + file.canRead());

System.out.println("Writable: " + file.canWrite());

System.out.println("File type: " + (file.isFile() ? "Regular file" : "Directory"));

System.out.println("File length (bytes): " + file.length());

System.out.println("\n--- File Content ---");

try (FileInputStream fis = new FileInputStream(file)) {

int ch;

while ((ch = fis.read()) != -1) {

System.out.print((char) ch);

}

} catch (IOException e) {

System.out.println("Error reading file: " + e.getMessage());

}

} else {

System.out.println("File does not exist.");

}

scanner.close();

}

}

**14.B) Copy File using Character Streams**

**File Name: FileCopyCharacterStream.java**

import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;

public class FileCopyCharacterStream {

public static void main(String[] args) {

String sourceFile = "source.txt";

String destinationFile = "destination.txt";

FileReader reader = null;

FileWriter writer = null;

try {

reader = new FileReader(sourceFile);

writer = new FileWriter(destinationFile);

int ch;

while ((ch = reader.read()) != -1) {

writer.write(ch);

}

System.out.println("File copied successfully.");

} catch (IOException e) {

System.out.println("An error occurred: " + e.getMessage());

} finally {

try {

if (reader != null)

reader.close();

if (writer != null)

writer.close();

} catch (IOException e) {

System.out.println("Error closing files: " + e.getMessage());

}

}

}

}

**14.C) Sum Integers with StringTokenizer**

**File Name: SumOfIntegers.java**

import java.util.Scanner;

import java.util.StringTokenizer;

public class SumOfIntegers {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.println("Enter a line of integers separated by spaces:");

String inputLine = scanner.nextLine();

StringTokenizer tokenizer = new StringTokenizer(inputLine);

int sum = 0;

System.out.println("\nIntegers found:");

while (tokenizer.hasMoreTokens()) {

String token = tokenizer.nextToken();

try {

int number = Integer.parseInt(token);

System.out.println(number);

sum += number;

} catch (NumberFormatException e) {

System.out.println("Skipping non-integer token: " + token);

}

}

System.out.println("\nSum of all integers: " + sum);

scanner.close();

}

}

**15.A) Vector and Wrapper Classes**

**File Name: VectorDemo.java**

import java.util.\*;

public class VectorDemo {

public static void main(String[] args) {

int n;

Vector<Integer> v = new Vector<>();

Scanner s = new Scanner(System.in);

for (int j = 0; j < 10; j++) {

System.out.print("Enter a number --> ");

n = s.nextInt();

v.add(n);

}

System.out.println("Vector contents: " + v);

System.out.print("Enter the first occurrence of a number to remove --> ");

n = s.nextInt();

boolean removed = v.remove(Integer.valueOf(n));

System.out.println("First occurrence of element " + n + " is removed: " + removed);

System.out.println("Vector contents after remove operation: " + v);

System.out.print("Enter an index of an element to be removed --> ");

n = s.nextInt();

if (n >= 0 && n < v.size()) {

System.out.println("Removed element at index " + n + ": " + v.remove(n));

} else {

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

}

System.out.println("Final Vector contents: " + v);

}

}

**File Name: WrapperDemo.java**

public class WrapperDemo {

public static void main(String args[]) {

Integer a = new Integer(10);

int b = a.intValue();

int c = a;

System.out.println("int value of Integer obj b is " + b);

System.out.println("int value of Integer obj c is " + c);

int a1 = 10;

Integer b1 = Integer.valueOf(a1);

Integer c1 = a;

if (b1 instanceof Integer)

System.out.println("TRUE b1 is an instance of Integer and value is " + b1);

if (c1 instanceof Integer)

System.out.println("TRUE c1 is an instance of Integer and value is " + c1);

}

}

**15.B) Generate Random Numbers**

**File Name: RandomBetweenTwoNumbers.java**

import java.util.Scanner;

import java.util.Random;

public class RandomBetweenTwoNumbers {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

Random random = new Random();

System.out.print("Enter the lower bound (x1 > 0): ");

int x1 = scanner.nextInt();

System.out.print("Enter the upper bound (x2): ");

int x2 = scanner.nextInt();

if (x1 <= 0 || x1 >= x2) {

System.out.println("Invalid input! Make sure x1 > 0 and x1 < x2.");

return;

}

System.out.print("Enter how many random numbers to generate: ");

int count = scanner.nextInt();

System.out.println("\nRandom numbers between " + x1 + " and " + x2 + ":");

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

int rand = random.nextInt(x2 - x1 + 1) + x1;

System.out.println(rand);

}

}

}

**15.C) Client-Server Application**

**File Name: CircleServer.java**

import java.io.\*;

import java.net.\*;

public class CircleServer {

public static void main(String[] args) throws IOException {

ServerSocket serverSocket = new ServerSocket(5000);

System.out.println("Server is running and waiting for client...");

Socket socket = serverSocket.accept();

System.out.println("Client connected.");

DataInputStream in = new DataInputStream(socket.getInputStream());

DataOutputStream out = new DataOutputStream(socket.getOutputStream());

double radius = in.readDouble();

System.out.println("Received radius from client: " + radius);

double area = Math.PI \* radius \* radius;

out.writeDouble(area);

System.out.println("Sent area to client: " + area);

socket.close();

serverSocket.close();

}

}

**File Name: CircleClient.java**

import java.io.\*;

import java.net.\*;

import java.util.Scanner;

public class CircleClient {

public static void main(String[] args) throws IOException {

Socket socket = new Socket("localhost", 5000);

DataInputStream in = new DataInputStream(socket.getInputStream());

DataOutputStream out = new DataOutputStream(socket.getOutputStream());

Scanner scanner = new Scanner(System.in);

System.out.print("Enter radius of circle: ");

double radius = scanner.nextDouble();

out.writeDouble(radius);

double area = in.readDouble();

System.out.println("Area of circle received from server: " + area);

socket.close();

}

}