**Matrix Multiplication:**

Write a Java program to perform matrix multiplication.

package matrix;

public class MatrixMultiplication {

public static void main(String[] args) {

int[][] num1 = { { 3, 4, 5 }, { 8, 6, 7 } };

int[][] num2 = { { 3, 4 }, { 8, 6 }, { 7, 5 } };

int result[][] = new int[num1.length][num2[0].length];

for (int i = 0; i < num1.length; i++) {

for (int j = 0; j < num2[0].length; j++) {

for (int k = 0; k < num1[0].length; k++) {

result[i][j] += num1[i][k] \* num2[k][j];

}

}

}

// Print the result

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

for (int j = 0; j < result[0].length; j++) {

System.out.print(result[i][j] + " ");

}

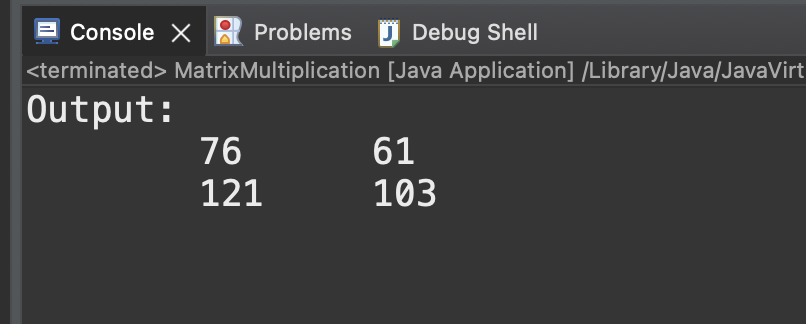
System.out.println();

}

}

}

**OUTPUT**:



**Swing GUI:**

Create a Java Swing GUI application that allows users to input their name and displays a personalized greeting message using JLabel and JTextField.

package Swing;

// imports

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JTextField;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

public class JavaGui {

public static void main(String[] args) {

JFrame frame = new JFrame("Greeting Application");

JLabel label = new JLabel(" Name:");

// create objects

final JTextField textField = new JTextField();

final JLabel resultLabel = new JLabel();

// position set Display

label.setBounds(10, 20, 150, 30);

textField.setBounds(170, 20, 100, 30);

resultLabel.setBounds(10, 70, 300, 30);

textField.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

String name = textField.getText();

resultLabel.setText("Hello, " + name + "!");

}

});

// add to frames

frame.add(label);

frame.add(textField);

frame.add(resultLabel);

frame.setLayout(null);

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

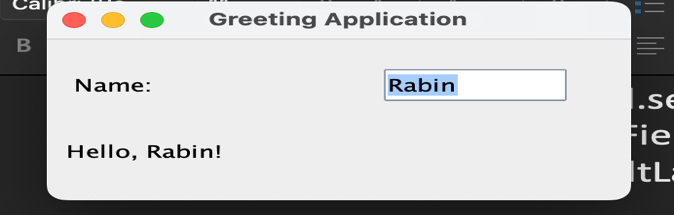
frame.setSize(300, 150);

frame.setVisible(true);

}

}

**Output**:



**Java Applet:**

Develop a simple Java applet program

Interface:

Define an interface named Shape with methods calculateArea() and calculatePerimeter(). Implement this interface in two classes: Circle and Rectangle. Write a program to demonstrate the use of the interface.

// interface class

package interfaces;

public interface Shape {

double calculateArea();

double calculatePerimeter();

}

package interfaces;

public class Circle implements Shape{

private double radius;

public Circle(double radi) {

this.radius= radi;

}

public double calculateArea() {

return Math.PI \* radius \* radius;

}

public double calculatePerimeter() {

return 2 \* Math.PI \* radius;

}

}

package interfaces;

class Rectangle implements Shape {

private double length;

private double width;

public Rectangle(double length, double width) {

this.length = length;

this.width = width;

}

public double calculateArea() {

return length \* width;

}

public double calculatePerimeter() {

return 2 \* (length + width);

}

}

package interfaces;

public class Main {

public static void main(String[] args) {

Circle circle = new Circle(5.0);

Rectangle rectangle = new Rectangle(7.0, 6.0);

System.out.println("\nCircle Area: "+ circle.calculateArea());

System.out.println("Circle Perimeter: " + circle.calculatePerimeter());

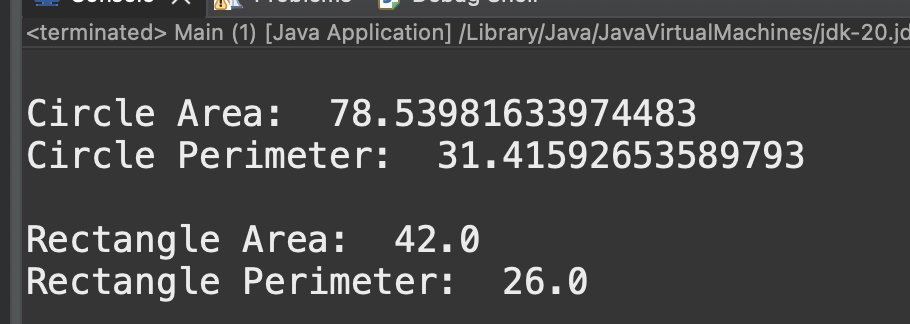
System.out.println("\nRectangle Area: " + rectangle.calculateArea());

System.out.println("Rectangle Perimeter: " + rectangle.calculatePerimeter());

}

}

**OUTPUT:**



**Abstract Class:**

Create an abstract class called Vehicle with abstract methods start() and stop(). Implement this abstract class in two concrete classes: Car and Motorcycle. Demonstrate the use of abstract classes and their methods.

package abstracts;

abstract class Vehicle {

public abstract void start();

public abstract void stop();

}

package abstracts;

public class Car extends Vehicle {

public void start() {

System.out.println("\nRabin Car Enginee start");

}

public void stop() {

System.out.println("Rabin Car Enginee stop");

}

}

package abstracts;

public class Motorcycle extends Vehicle {

public void start() {

System.out.println("\nDuke 200 Enginee start");

}

public void stop() {

System.out.println("Duke 200 Enginee stop");

}

}

package abstracts;

public class Vehicle\_Main {

public static void main(String[] args) {

// create an objects

Motorcycle bike = new Motorcycle();

Car car = new Car();

// call methods

bike.start();

bike.stop();

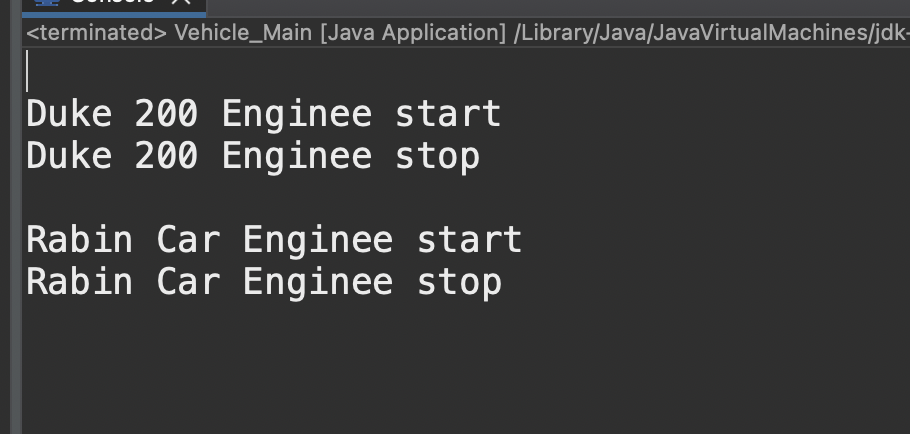
car.start();

car.stop();

}

}

**OUTPUT:**

****

**Super Class / Inheritance:**

Design a class hierarchy representing different types of animals. Include a superclass Animal with subclasses such as Mammal, Bird, and Fish. Implement the concept of inheritance and demonstrate it through a program.

**Constructors**:

Create a class called Book with attributes title, author, and price. Include a default constructor and a parameterized constructor. Write a program that creates instances of the Book class using both constructors.

**Error Handling:**

Write a Java program that reads an integer from the user. Handle the possibility of the user entering a non-integer value and provide appropriate error messages. Ensure the program does not crash due to invalid input.

**GUI-Based Functional Calculator:**

Design a GUI-based functional calculator using Java Swing. Include basic arithmetic operations (addition, subtraction, multiplication, division). Implement error handling for division by zero and invalid input.