ASSIGNMENT-2

P.SIRISHA

192110113

**1.**

public class Account {

private double balance;

private double interestRate;

public Account(double initialBalance) {

this.balance = Math.max(initialBalance, 0);

this.interestRate = 0.02;

}

public void deposit(double amount) {

if (amount > 0) {

balance += amount;

System.out.println("$" + amount + " deposited successfully.");

} else {

System.out.println("Invalid deposit amount.");

}

}

public void withdraw(double amount) {

if (amount > 0) {

if (amount <= balance) {

balance -= amount;

System.out.println("$" + amount + " withdrawn successfully.");

} else {

System.out.println("Insufficient funds. $5 penalty charged.");

balance -= 5; // Charge a $5 penalty

}

} else {

System.out.println("Invalid withdrawal amount.");

}

}

public double getBalance() {

return balance;

}

public void computeInterest() {

double interest = balance \* interestRate;

balance += interest;

System.out.println("Interest of $" + interest + " applied. Current balance: $" + balance);

}

public static void main(String[] args) {

Account myAccount = new Account(1000);

myAccount.deposit(500);

myAccount.withdraw(300);

myAccount.computeInterest();

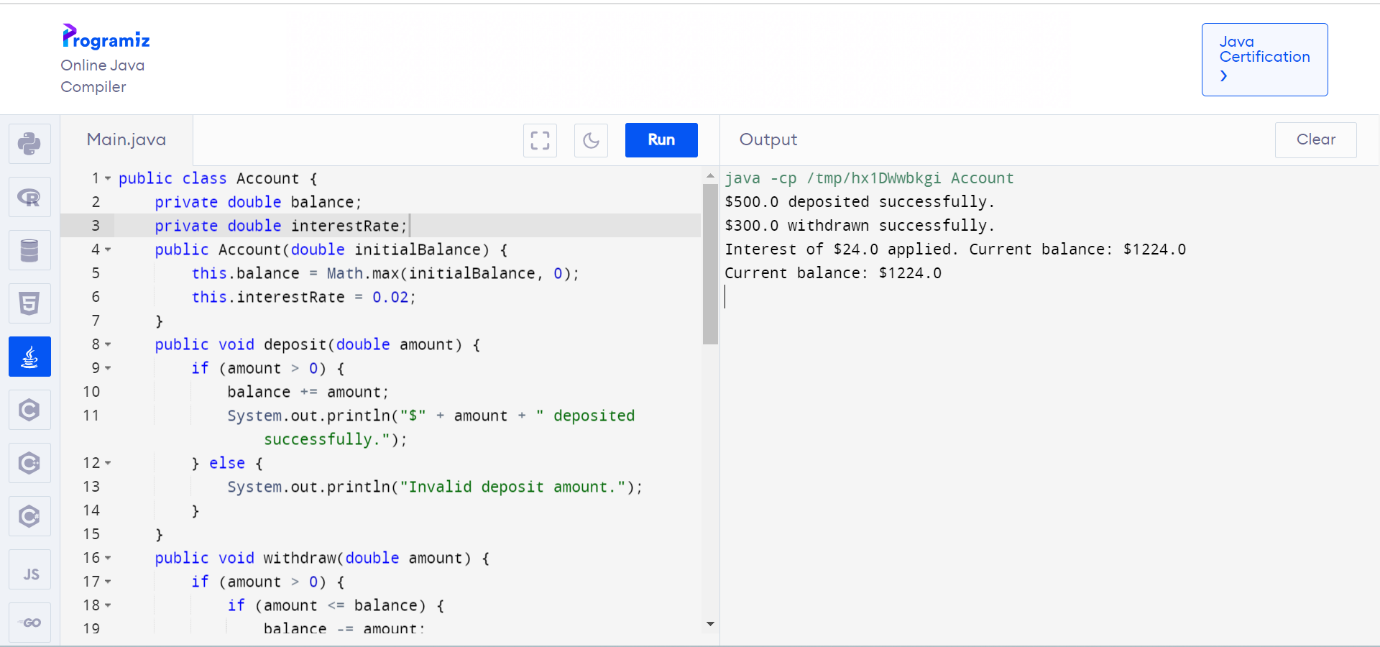
double currentBalance = myAccount.getBalance();

System.out.println("Current balance: $" + currentBalance);

}

}

**OUTPUT:**

****

**2.**

public class Triangle {

private double side1;

private double side2;

private double side3;

public Triangle(double side1, double side2, double side3) {

this.side1 = side1;

this.side2 = side2;

this.side3 = side3;

}

public boolean isRight() {

return (Math.pow(side1, 2) + Math.pow(side2, 2) == Math.pow(side3, 2)

|| Math.pow(side1, 2) + Math.pow(side3, 2) == Math.pow(side2, 2)

|| Math.pow(side2, 2) + Math.pow(side3, 2) == Math.pow(side1, 2));

}

public boolean isScalene() {

return !isIsosceles();

}

public boolean isIsosceles() {

return (side1 == side2 || side1 == side3 || side2 == side3);

}

public boolean isEquilateral() {

return (side1 == side2 && side1 == side3);

}

public static void main(String[] args) {

Triangle triangle = new Triangle(3, 4, 5);

System.out.println("Is right triangle? " + triangle.isRight());

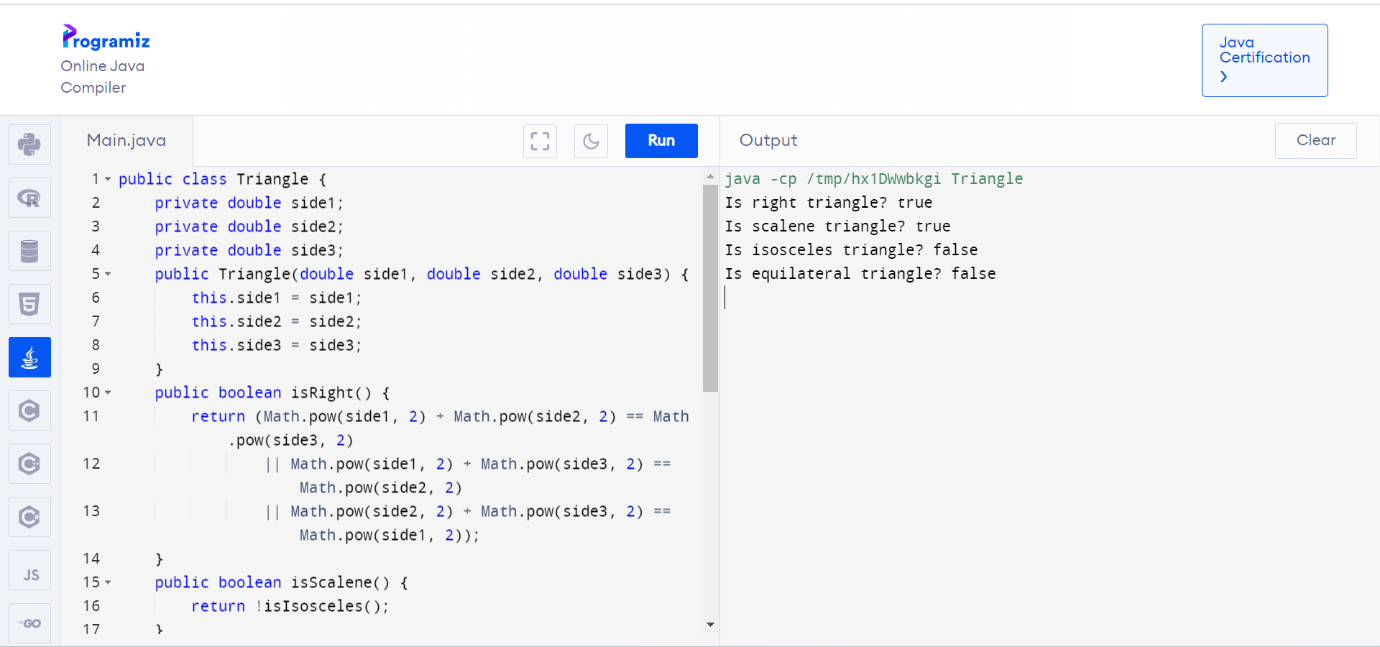
System.out.println("Is scalene triangle? " + triangle.isScalene());

System.out.println("Is isosceles triangle? " + triangle.isIsosceles());

System.out.println("Is equilateral triangle? " + triangle.isEquilateral());

}

}

Output:

**3.**

public class MatrixMultiplication {

public static void main(String[] args) {

int[][] mat1 = {

{1, 2},

{5, 3}

};

int[][] mat2 = {

{2, 3},

{4, 1}

};

int[][] result = multiplyMatrices(mat1, mat2);

System.out.println("Mat Sum = ");

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();

}

}

public static int[][] multiplyMatrices(int[][] mat1, int[][] mat2) {

int rows1 = mat1.length;

int cols1 = mat1[0].length;

int rows2 = mat2.length;

int cols2 = mat2[0].length;

if (cols1 != rows2) {

throw new IllegalArgumentException("Matrix dimensions are not compatible for multiplication");

}

int[][] result = new int[rows1][cols2];

for (int i = 0; i < rows1; i++) {

for (int j = 0; j < cols2; j++) {

int sum = 0;

for (int k = 0; k < cols1; k++) {

sum += mat1[i][k] \* mat2[k][j];

}

result[i][j] = sum;

}

}

return result;

}

}

Output:

