**Date:**

**Practical 1:** Write a program to make a simple calculator.

**Code :**

import java.util.Scanner;

class Calc

{

public static void main(String []args)

{

Scanner s = new Scanner(System.in);

int a,b,sum,choice;

System.out.println("------Calculator------");

System.out.println("Enter 1st Number : ");

a = s.nextInt();

System.out.println("Enter 2nd Number : ");

b = s.nextInt();

System.out.println("Enter Operation.....\n 1 : Addition \n 2 : Subtraction \n 3 : Multipilcation \n 4 : Division \n 5 : Modulo \n");

choice = s.nextInt();

switch(choice)

{

case 1 : sum = a + b;

System.out.println("Addition of "+a+" and "+b+" is : "+sum);

break;

case 2 : sum = a - b;

System.out.println("Subtraction of "+b+" from "+a+" is : "+sum);

break;

case 3 : sum = a \* b;

System.out.println("Multiplication of "+a+" with "+b+" is : "+sum);

break;

case 4 : sum = a / b;

System.out.println("Division of "+b+" from "+a+" is : "+sum);

break;

case 5 : sum = a % b;

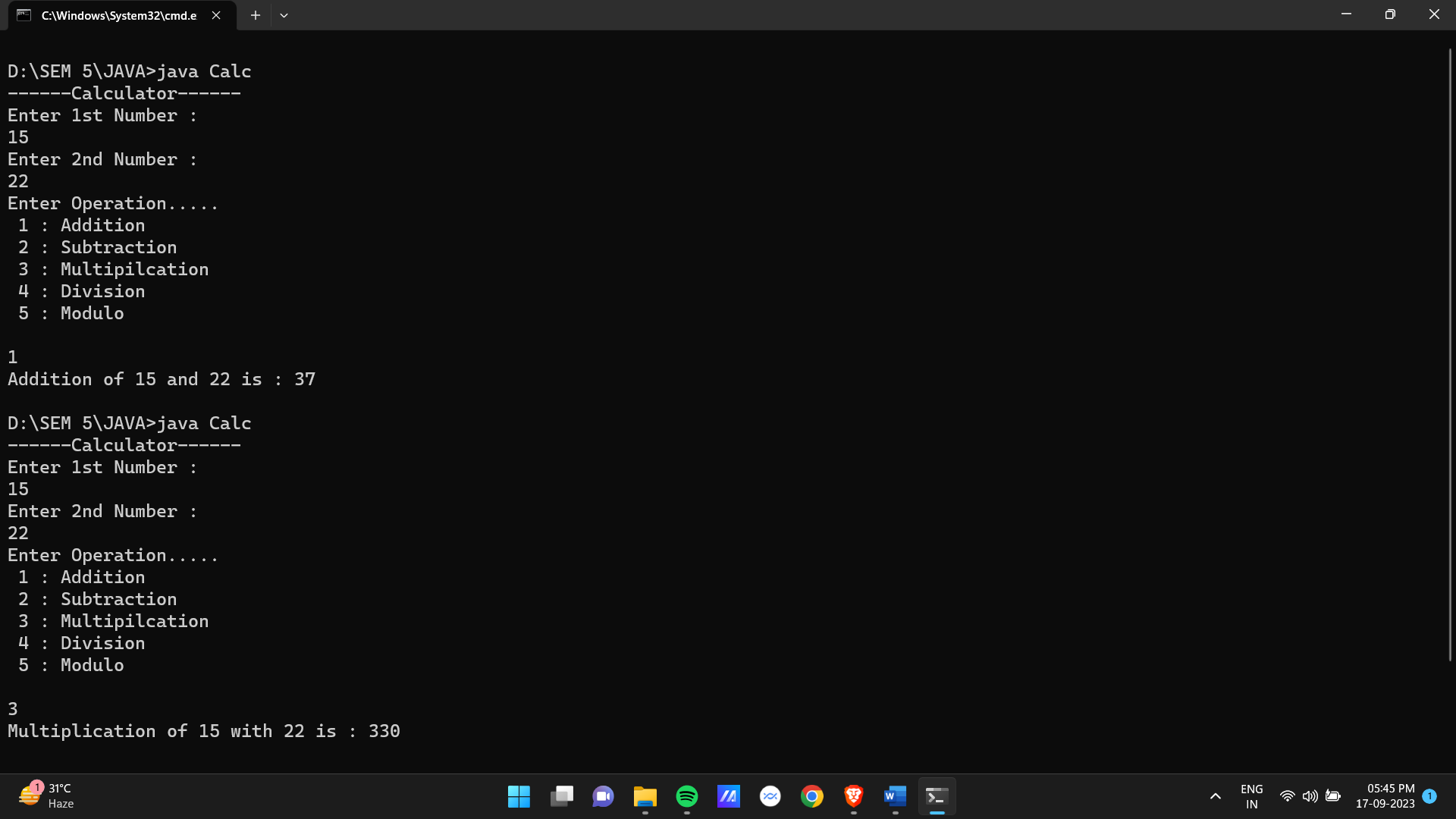
System.out.println("Modulo of "+a+" and "+b+" is : "+sum);

}

}

}

**Output :**



**Date:**

**Practical 2:** Write a program to check a number is palindrome or not.

**Code :**

import java.util.\*;

class Palindrome

{

public static void main(String[] args)

{

Scanner s = new Scanner(System.in);

int num,temp,rem;

int sum=0;

System.out.println("Enter a Number to check weather it is palindrome or not");

num = s.nextInt();

temp = num;

while(num > 0)

{

rem = num%10;

sum = (sum\*10)+rem;

num = num/10;

}

if(temp == sum)

{

System.out.println("Entered number is a Palindrome Number.");

}

else

{

System.out.println("Entered number is not a Palindrome Number.");

}

}

}

**Output :**

A screenshot of a computer

Description automatically generated

**Date:**

**Practical 3:** Write a program to check a number is prime or not between given range.

**Code :**

import java.util.\*;

class PrimeNumber

{

static void prime\_N(int num1,int num2)

{

int x, y, flg;

System.out.println("All the Prime numbers within " +num1+ " and " +num2+ " are:");

for (x = num1; x <= num2; x++)

{

if (x == 1 || x == 0)

continue;

flg = 1;

for (y = 2; y <= x / 2; ++y)

{

if (x % y == 0) {

flg = 0;

break;

}

}

if (flg == 1)

{

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

}

}

}

public static void main(String[] args)

{

Scanner s = new Scanner(System.in);

int num1;

System.out.println("Enter Range 1:");

num1 = s.nextInt();

int num2;

System.out.println("Enter Range 2:");

num2 = s.nextInt();

prime\_N(num1,num2);

}

}

**Output :**

A screenshot of a computer

Description automatically generated

**Date:**

**Practical 4:** Write a program to implement matrix multiplication.

**Code :**

import java.util.\*;

class MatrixMultiplication{

public static void main(String[] args){

Scanner sc = new Scanner(System.in);

int rows,columns;

System.out.println("\n-----Matrix Multipliation----");

System.out.print("\nEnter no of rows : ");

rows=sc.nextInt();

System.out.print("Enter no of columns : ");

columns=sc.nextInt();

int[][] matrix1=new int[rows][columns];

int[][] matrix2=new int[rows][columns];

int[][] mul=new int[rows][columns];

System.out.print("\nEnter value for Matrix1 ("+rows+","+columns+")\n\n");

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

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

System.out.print("Enter value for Matrix1 ("+i+","+j+") : ");

matrix1[i][j]=sc.nextInt();

}

}

System.out.print("\nEnter value for Matrix2 ("+rows+","+columns+")\n\n");

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

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

System.out.print("Enter value for Matrix2 ("+i+","+j+") : ");

matrix2[i][j]=sc.nextInt();

}

}

System.out.println("\n-----Matrix 1-----\n");

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

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

System.out.print(matrix1[i][j]+" ");

}

System.out.println();

}

System.out.println("\n-----Matrix 2-----\n");

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

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

System.out.print(matrix2[i][j]+" ");

}

System.out.println();

}

System.out.println("\n-----Multiplication of Matrix 1 and Matrix 2-----\n");

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

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

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

mul[i][j]+=matrix1[i][k]\*matrix2[k][j];

}

}

}

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

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

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

}

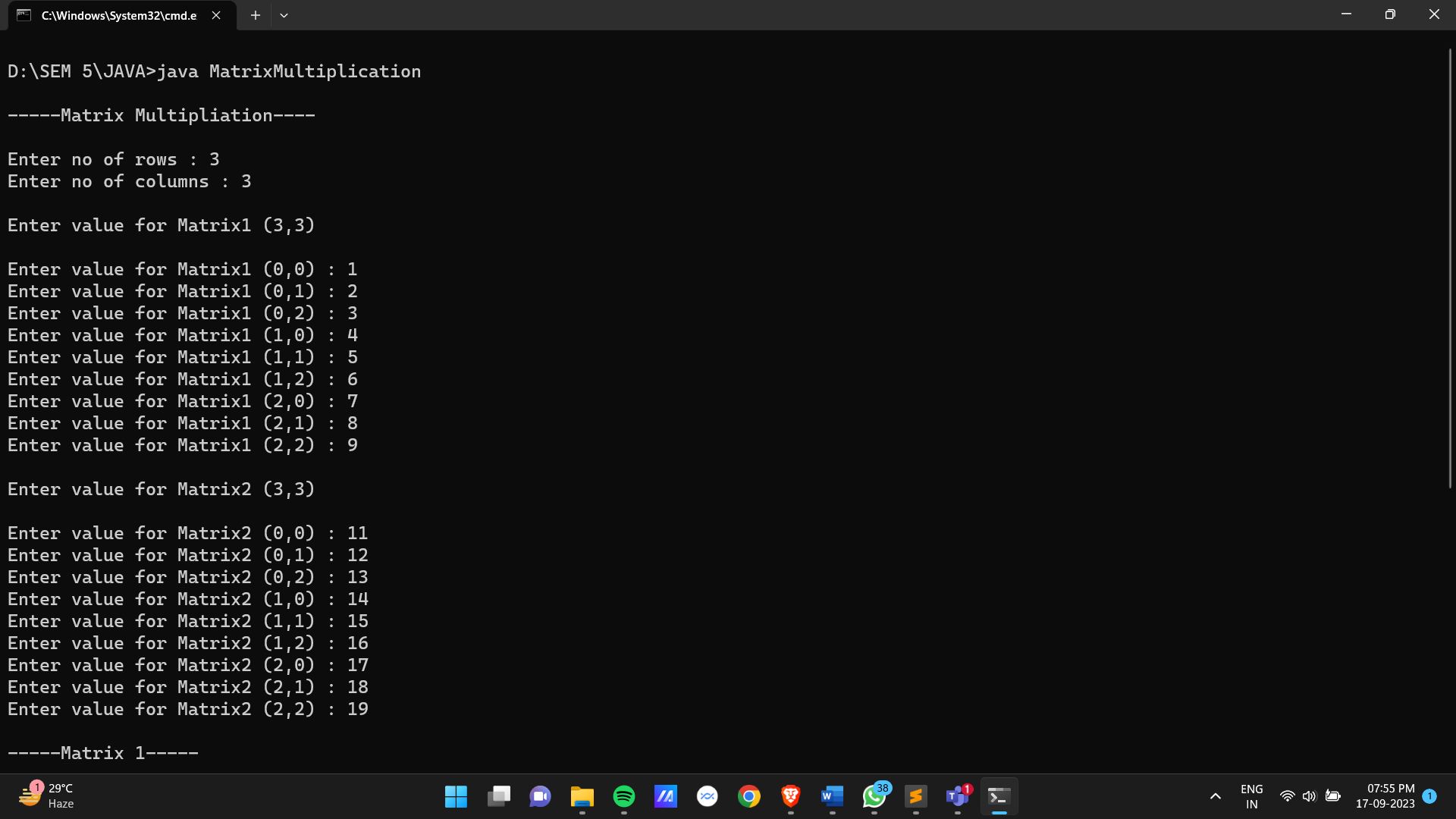
System.out.println();

}

}

}

**Output :**

 A screenshot of a computer

Description automatically generated

**Date:**

**Practical 5:** Write a program to implement sum of digits of a number.

**Code :**

import java.util.Scanner;

class SumofDigits{

public static void main(String[] args){

Scanner s = new Scanner(System.in);

int num,temp;

int sum=0;

System.out.print("Enter the Number : ");

num = s.nextInt();

temp = num;

while(temp>0)

{

sum+=temp%10;

temp/=10;

}

System.out.println("Sum of digits is : " +sum);

}

}

**Output :**

A screenshot of a computer

Description automatically generated

**Date:**

**Practical 6:** Write a program to implement a number is Armstrong or not.

**Code :**

import java.util.Scanner;

class Armstrong{

public static void main(String[] args){

int num,temp,rem,check=0,digit=0;

Scanner sc = new Scanner(System.in);

System.out.println("------Armstrong------");

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

num=sc.nextInt();

temp=num;

while(temp>0){

temp/=10;

digit++;

}

temp=num;

while(temp>0){

rem=temp%10;

check+=Math.pow(rem,digit);

temp/=10;

}

if(num==check){

System.out.println(num +" is an armstrong number");

}else{

System.out.println(num +" is not an armstrong number");

}

}

}

**Output :**

A screenshot of a computer

Description automatically generated

**Date:**

**Practical 7:** Write a program to implement dynamic stack.

**Code :**

import java.util.Scanner;

class StackClass{

int tos;

int len;

int stack[];

StackClass(){

tos = -1;

}

StackClass(int d\_len){

tos = -1;

len = d\_len;

stack = new int[len];

}

void push(int ele){

if(tos==len-1){

int[] temp = new int[len\*2];

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

temp[i]=stack[i];

}

stack=temp;

stack[++tos]=ele;

System.out.println("Pushed Element " +stack[tos] +" in stack");

len=len\*2;

}else{

stack[++tos]=ele;

System.out.println("Pushed SElement " +stack[tos] +" in stack");

}

}

void pop(){

if(tos==-1){

System.out.println("\nStack is empty!!!");

}else{

tos--;

System.out.println("Popped element " +stack[tos+1] +" in stack");

}

}

void display(){

if(tos==-1){

System.out.println("\nStack is empty.");

}else{

System.out.println("\n------Stack-----\n");

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

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

}

}

}

}

class Stack{

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int choice,ele,len;

System.out.println("-----Dyanamic Stack-----\n");

System.out.print("Enter initial size of stack : ");

len=sc.nextInt();

StackClass stk = new StackClass(len);

do{

System.out.print("\n1 : PUSH\n2 : POP\n3 : DISPLAY\n");

choice = sc.nextInt();

if(choice == 1){

System.out.print("\nEnter element : ");

ele=sc.nextInt();

stk.push(ele);

}else if (choice == 2) {

stk.pop();

}else if (choice == 3) {

stk.display();

}else if (choice == 4) {

System.out.println("Thanks for visit");

}else{

System.out.println("\nInvalid input");

}

}while(choice!=4);

}

}

**Output :**

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

**Date:**

**Practical 8:** Write a program to demonstrate constructor overloading and method overloading.

**Code :**

class Rectangle {

int height;

int width;

Rectangle() {

height = 10;

width = 10;

}

Rectangle(int height, int width) {

this.height = height;

this.width = width;

}

Rectangle(Rectangle obj) {

height = obj.height;

width = obj.width;

}

void area() {

System.out.println("The Height of Rectangle : " + height);

System.out.println("The width of Rectangle : " + width);

System.out.println("\nThe Area of Rectangle : " + (height \* width));

}

}

class Area {

int a;

int b;

void sum(int a, int b) {

System.out.println("\nArea of Rectangle L = " + a + " and B = " + b + " is : " + (a \* b));

}

void sum(int a, double b) {

System.out.println("Area of Rectangle L = " + a + " and B = " + b + " is : " + (a \* b));

}

void sum(double a, double b) {

System.out.println("Area of Rectangle L = " + a + " and B = " + b + " is : " + (a \* b));

}

}

class ConstructorOverloading {

public static void main(String[] args) {

System.out.println("\n-----Overloading Constructor-----");

System.out.println("\nObject1 with Default Constructor.....\n");

Rectangle obj1 = new Rectangle();

obj1.area();

System.out.println("\n->Object2 with Parameterized Constructor\n");

Rectangle obj2 = new Rectangle(10, 20);

obj2.area();

System.out.println("\n->Object3 with object2 in formal argument\n");

Rectangle obj3 = new Rectangle(obj2);

obj3.area();

System.out.println("\n-----Method Overloading-----");

Area obj4 = new Area();

obj4.sum(10, 20);

obj4.sum(10, 20.98);

obj4.sum(10.20, 9.80);

}

}

**Output :**

A screenshot of a computer

Description automatically generated

**Date:**

**Practical 9:** Write a program to set up an array of 10 variables each containing an arbitrary string of form month date year for example 30/10/19 and output as 30th October 1999.

**Code :**

class MonthNumber {

static String nameofMonth(String month) {

if(month.equals("01")) {

return "January";

}

else if(month.equals("02")) {

return "February";

}

else if(month.equals("03")) {

return "March";

}

else if(month.equals("04")) {

return "April";

}

else if(month.equals("05")) {

return "May";

}

else if(month.equals("06")) {

return "June";

}

else if(month.equals("07")) {

return "July";

}

else if(month.equals("08")) {

return "August";

}

else if(month.equals("09")) {

return "September";

}

else if(month.equals("10")) {

return "October";

}

else if(month.equals("11")) {

return "November";

}

else if(month.equals("12")) {

return "December";

}

else {

return "Illegal";

}

}

public static void main(String[] args) {

String array[] = {"30-10-2019", "27-07-2022", "14-01-2004", "09-08-2021", "24-08-2023", "17-01-2019", "01-07-2022", "10-12-2000", "25-11-2021", "26-09-2020"};

System.out.println("\n----Date with Month's Name-----");

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

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

System.out.println(array[i]);

}

System.out.println("\nModified Date : \n");

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

String month = array[i].substring(3, 5);

String res = nameofMonth(month);

if(!res.equals("Illegal")) {

System.out.println(array[i].substring(0, 2) + " " + res + " " + array[i].substring(6, 10));

} else {

System.out.println("\n\nInvalid Input");

break;

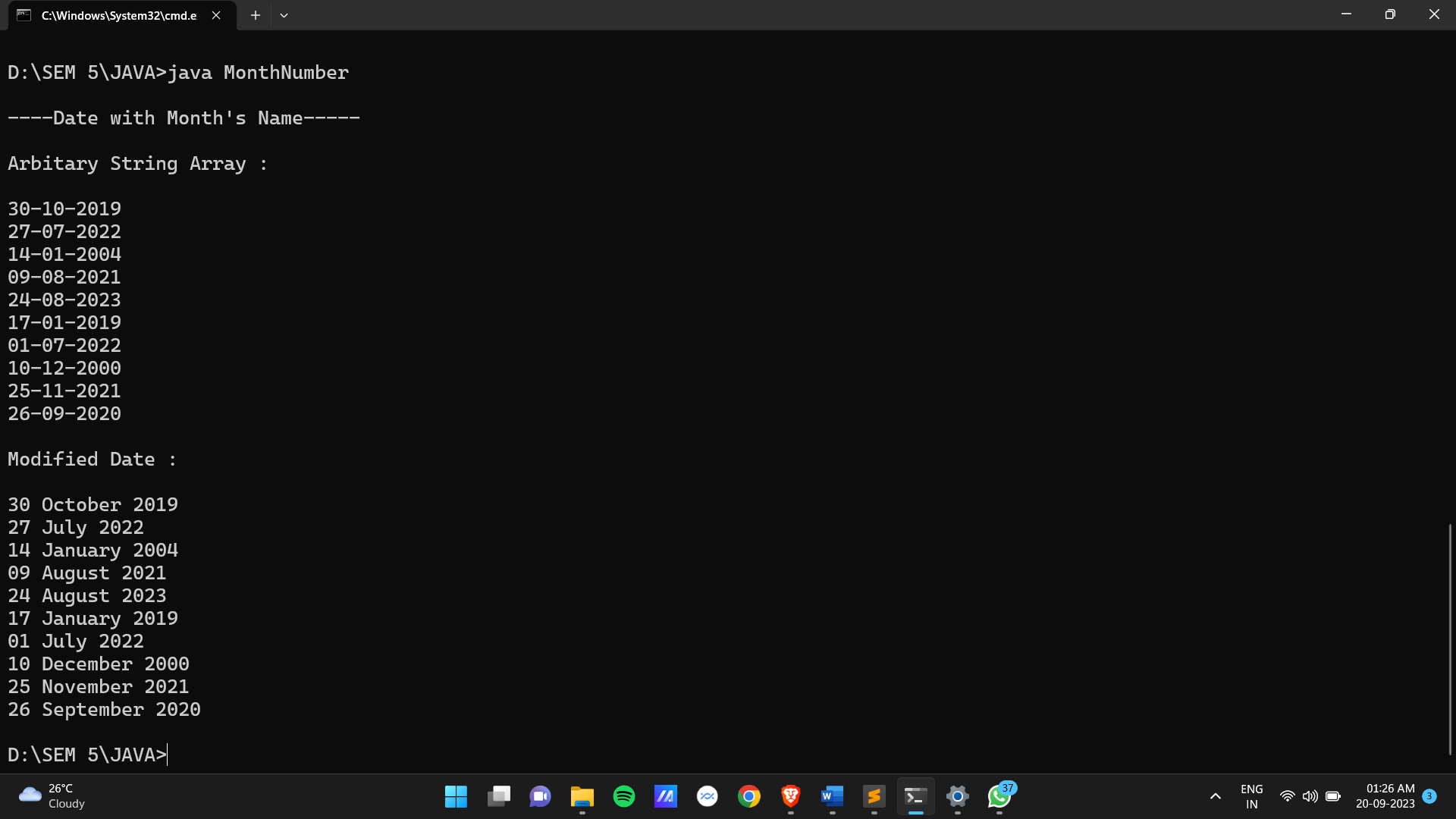
}

}

}

}

**Output :**



**Date:**

**Practical 10:** Write a program to define a mcm length to represent a length measured in meters and millimeters each stored as integers. Include method to add and subtract object to multiply and dived an object by an integer value to calculate area resulting from the product of two objects and two compare objects. Include constructors that accept

1. Three arguments meters, cm, mm
2. One integer argument with length set to zero. Create the class by creating some objects and testing the class operations.

**Code :**

class Length {

int meter;

int cm;

int mm;

Length(int milimeter) {

meter = 0;

cm = 0;

mm = milimeter;

}

Length(int meter, int cm, int mm) {

this.meter = meter;

this.cm = cm;

this.mm = mm;

}

int toMilimeter() {

return ((this.meter \* 1000) + (this.cm \* 10) + (this.mm));

}

Length add(Length obj) {

int newMilimeter = this.toMilimeter() + obj.toMilimeter();

return new Length(0, 0, newMilimeter);

}

Length sub(Length obj) {

int newMilimeter = this.toMilimeter() - obj.toMilimeter();

return new Length(0, 0, newMilimeter);

}

Length mul(int factor) {

int newMilimeter = this.toMilimeter() \* factor;

return new Length(0, 0, newMilimeter);

}

Length div(int factor) {

int newMilimeter = this.toMilimeter() / factor;

return new Length(0, 0, newMilimeter);

}

int area(Length obj) {

int area = this.toMilimeter() \* obj.toMilimeter();

return area;

}

int compare(Length obj) {

int thisMilimeter = this.toMilimeter();

int objMilimeter = obj.toMilimeter();

if(thisMilimeter > objMilimeter) {

return 1;

} else if(thisMilimeter < objMilimeter) {

return -1;

} else {

return 0;

}

}

}

class LengthClass {

public static void main(String[] args) {

System.out.println("\n-----Length Class With Functionality-----");

System.out.println("\nLength Object 1 with 10 milimeter");

Length obj1 = new Length(10);

System.out.println("Length of obj1 is : " + obj1.toMilimeter());

System.out.println("\nLength obj2 with 10 meter, 20 cm, 30 milimeter");

Length obj2 = new Length(10, 20, 30);

System.out.println("Length of obj2 is : " + obj2.toMilimeter());

System.out.println("\nCompare Object 1 & Object 2");

int check = obj1.compare(obj2);

if(check == 0) {

System.out.println("The obj1 length is equal to obj2");

} else if(check == 1) {

System.out.println("The object1 length is greater than obj2");

} else if(check == -1) {

System.out.println("The obj1 length is less than obj2");

}

System.out.println("\nAddition by 10 meter, 20 cm, 10 mm on obj1");

Length res = obj1.add(new Length(10, 20, 20));

System.out.println("Length of object after Addition is : " + res.toMilimeter());

System.out.println("\nSubtraction operation by 5 mm on obj1");

res = obj1.sub(new Length(0, 0, 5));

System.out.println("Length of object after subtraction is : " + res.toMilimeter());

System.out.println("\nMultiplication operation by 3 on obj1");

res = obj1.mul(3);

System.out.println("Length of object after multiplication is : " + res.toMilimeter());

System.out.println("\nDivision operation by 5 on obj1");

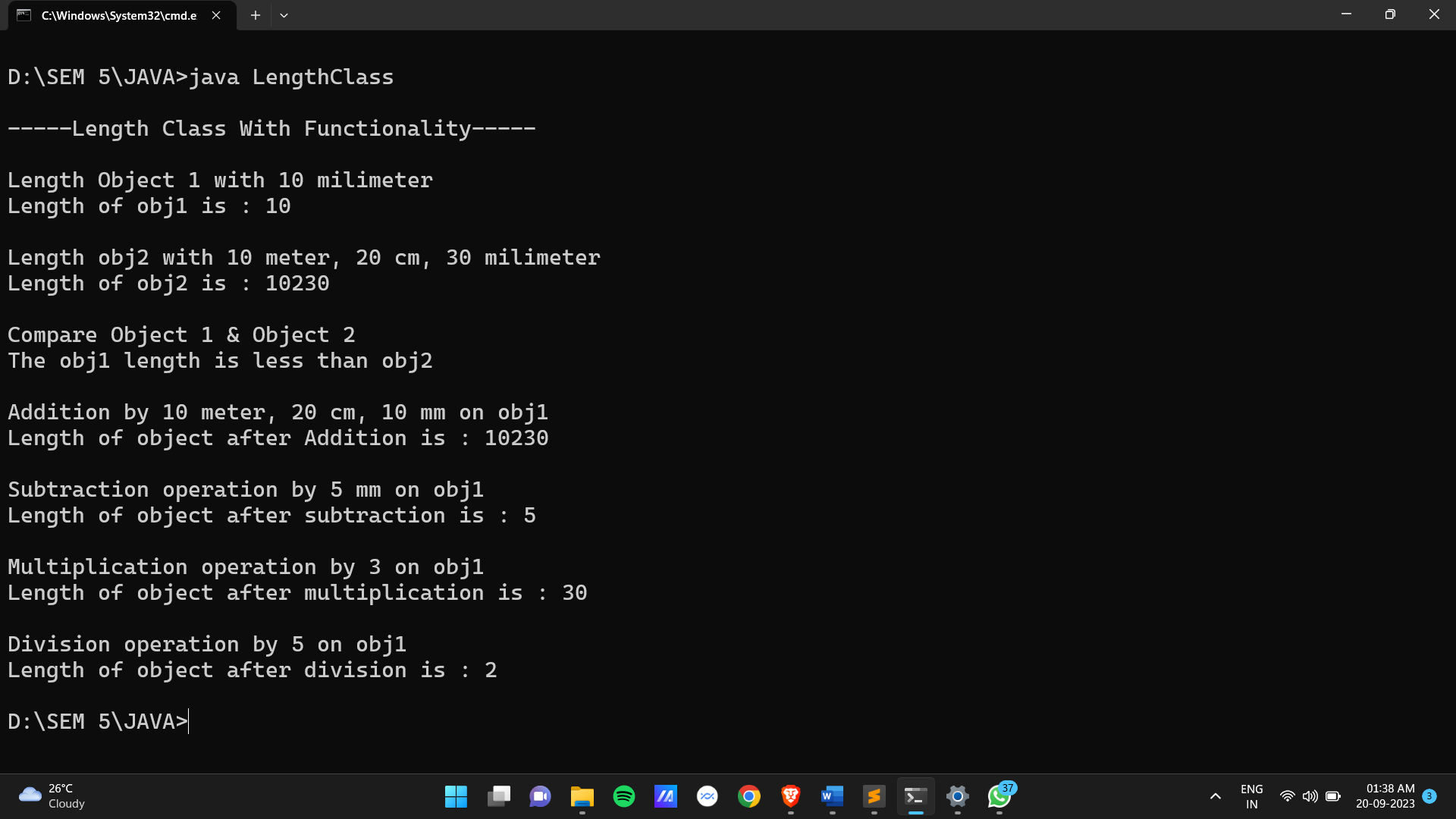
res = obj1.div(5);

System.out.println("Length of object after division is : " + res.toMilimeter());

}

}

**Output :**



**Date:**

**Practical 11:** Write a program to implement factorial of a number using recursion.

**Code :**

import java.util.Scanner;

class Factorial {

static int rec(int n) {

if(n == 1) {

return n;

} else {

return n \* rec(n - 1);

}

}

public static void main(String[] args) {

System.out.println("-----Factorial-----");

Scanner sc = new Scanner(System.in);

System.out.print("Enter Value : ");

int val = sc.nextInt();

int ans = rec(val);

System.out.println("Factorial of " + val + " is : " + ans);

}

}

**Output :**

A screenshot of a computer

Description automatically generated

**Date:**

**Practical 12:** Write a program to implement G.C.D of numbers using recursion.

**Code :**

import java.util.Scanner;

class Gcd {

static int gcd(int a, int b) {

if (b == 0)

return a;

return gcd(b, a % b);

}

public static void main(String[] args) {

System.out.println("----- GCD -----");

Scanner sc = new Scanner(System.in);

System.out.print("Enter Value 1 : ");

int num1 = sc.nextInt();

System.out.print("Enter Value 2 : ");

int num2 = sc.nextInt();

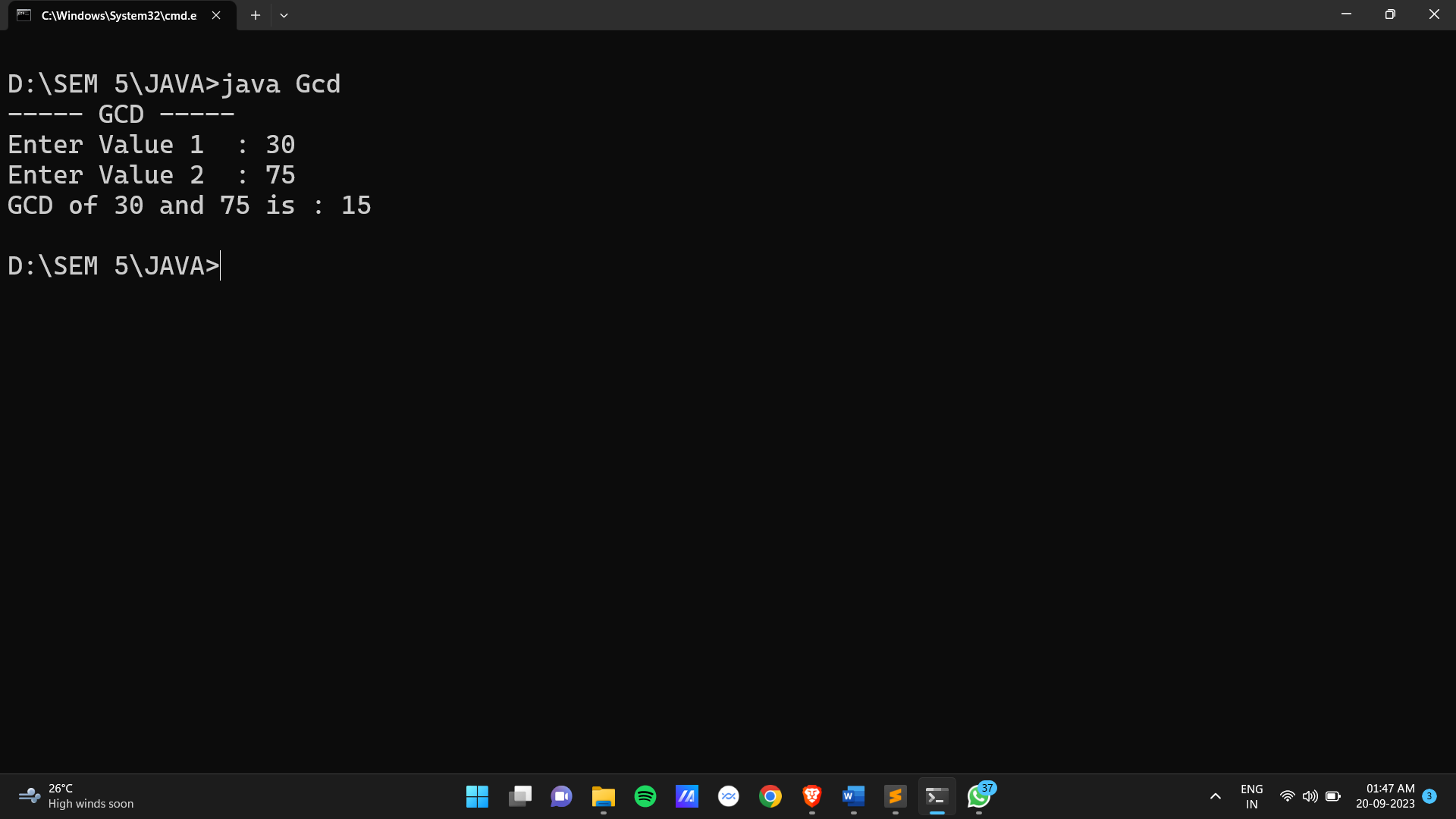
int gcd = gcd(num1, num2);

System.out.println("GCD of " + num1 + " and " + num2 + " is : " + gcd);

}

}

**Output :**



**Date:**

**Practical 13:** Write a program to check a matrix is a magic square matrix or not.

**Code :**

import java.util.\*;

class MagicMatrix {

public static void main(String[] args) {

int row, col, matrix[][], sum\_rows[], sum\_cols[];

int ind = 0, sum\_diagonal1 = 0, sum\_diagonal2 = 0;

Scanner sc = new Scanner(System.in);

System.out.println("\n.....Magic Matrix Implementation.....\n");

System.out.print("Enter no. rows of matrix : ");

row = sc.nextInt();

System.out.print("Enter no. columns of matrix : ");

col = sc.nextInt();

if(row == col) {

matrix = new int[row][col];

sum\_rows = new int[row];

sum\_cols = new int[col];

System.out.println("\nEnter data for matrix : ");

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

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

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

matrix[i][j] = sc.nextInt();

}

}

System.out.println("\n\nGiven Matrix is : ");

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

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

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

}

System.out.println();

}

int flag = 0;

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

int sum1 = 0, sum2 = 0;

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

sum1 += matrix[i][j];

sum2 += matrix[j][i];

if(i == j) {

sum\_diagonal1 += matrix[i][j];

} import java.util.Scanner;

class Matrix {

int row,column,rawTotal,columnTotal,rowCount,columnCount,diagonalTotal,antiDiagonalTotal,total;

int matrix[][];

boolean falgTotal,falgRow,flagColumn,flagDiagonal,flagAntiDiagonal;

Scanner sc = new Scanner(System.in);

Matrix(){

row=column=rawTotal=columnTotal=rowCount=columnCount=diagonalTotal=antiDiagonalTotal=total=0;

falgTotal=falgRow=flagColumn=flagDiagonal=flagAntiDiagonal=false;

}

void setMatrix() {

System.out.print("Enter num of rows : ");

row = sc.nextInt();

System.out.print("Enter num of columns : ");

column = sc.nextInt();

matrix = new int[row][column];

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

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

System.out.print("Enter value for (" + (i + 1) + "," + (j + 1) + ") : ");

matrix[i][j] = sc.nextInt();

}

}

}

void getMatrix() {

System.out.println("\n-----Matrix-----");

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

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

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

}

System.out.println();

}

}

void checkMatrix(){

if(row != column){

System.out.println("\nThis matrix is not a Magic Matrix!!!");

}else{

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

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

if(!falgTotal){

total += matrix[i][j];

}

rawTotal += matrix[i][j];

falgRow = true;

}

if(!falgTotal){

falgTotal = true;

}

if(falgRow){

if(total == rawTotal){

rowCount++;

}

}

rawTotal = 0;

falgRow = false;

}

if(row != rowCount){

System.out.println("\nThis matrix is not a Magic Matrix!!!");

}else{

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

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

columnTotal += matrix[j][i];

flagColumn = true;

}

if(flagColumn){

if(total == columnTotal){

columnCount++;

}

}

columnTotal = 0;

flagColumn = false;

}

if(rowCount != columnCount){

System.out.println("\nThis matrix is not a Magic Matrix");

}else{

int k = 0;

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

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

if(i == j){

diagonalTotal+=matrix[j][i];

}

if(k==i && j == (column-i-1)){

antiDiagonalTotal+=matrix[i][j];

k++;

}

}

if(total == diagonalTotal && total == antiDiagonalTotal){

flagDiagonal = true;

flagAntiDiagonal = true;

}

}

if(flagDiagonal && flagAntiDiagonal){

System.out.println("\nThis matrix is a Magic Matrix");

}else{

System.out.println("\nThis matrix is not a Magic Matrix!!!");

}

}

}

}

}

}

class MagicMatrix{

public static void main(String args[]){

Matrix m = new Matrix();

m.setMatrix();

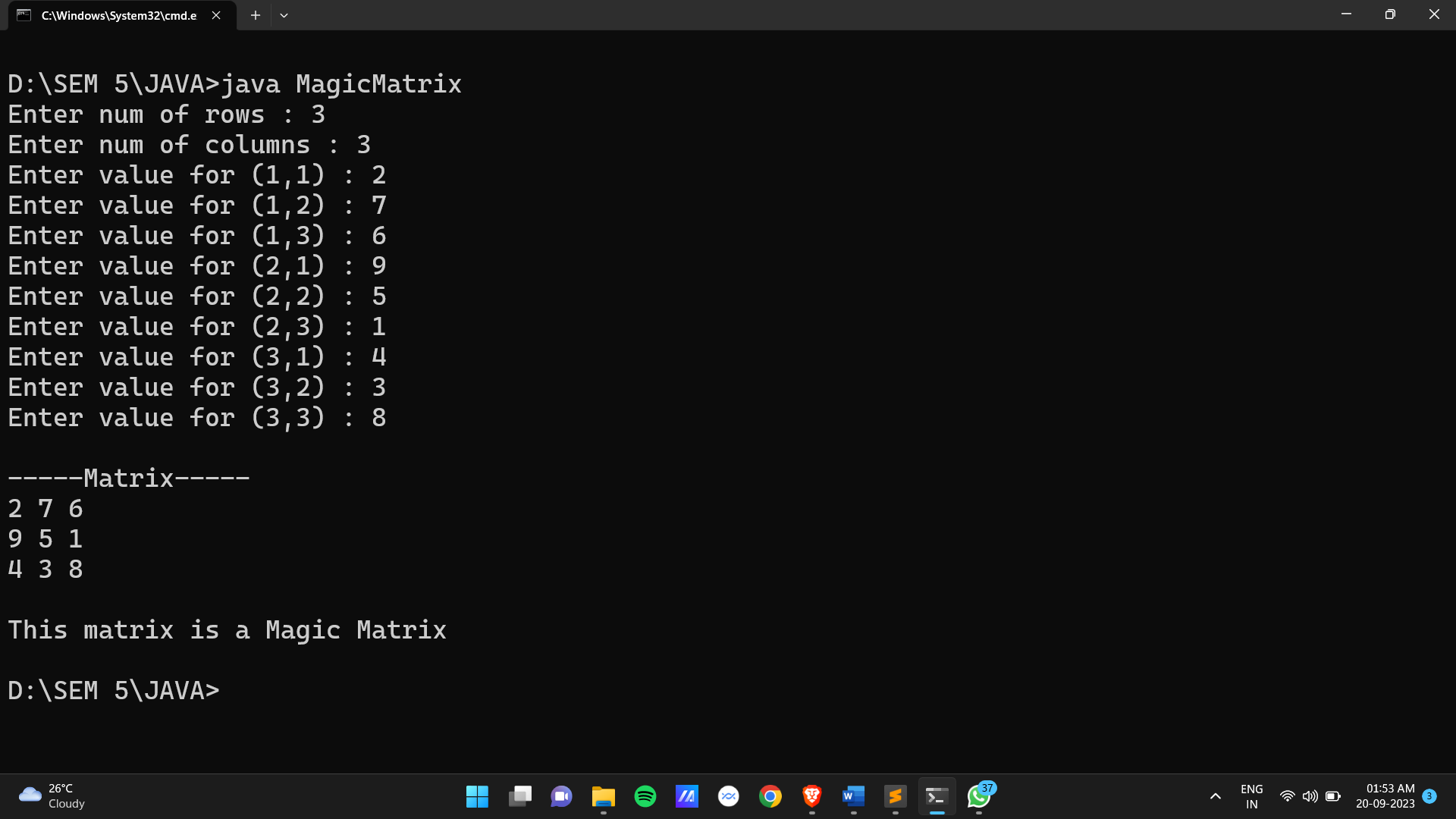
m.getMatrix();

m.checkMatrix();

}

}

**Output :**



**Date:**

**Practical 14:** Write a program which shows inheritance.

**Code :**

class Square1 {

int a;

Square1() {

a = 0;

}

Square1(int a) {

this.a = a;

}

void showSquare1() {

System.out.println("Lenght of a from class Square1 is : " + a);

}

}

class Square2 extends Square1{

int b;

Square2() {

b = 0;

}

Square2(int b) {

this.b = b;

}

void showSquare2() {

System.out.println("Length of b from class Square2 is : " + b);

}

}

class Inheritance {

public static void main(String[] args) {

System.out.println("\n----- Inheritance -----");

System.out.println("\nLength of Square1 set to 10");

Square1 objA = new Square1(10);

System.out.println("Length of Square2 set to 20");

Square2 objB = new Square2(20);

System.out.println("\nBy object of subclass B showSquare1() from class A");

objB.showSquare1();

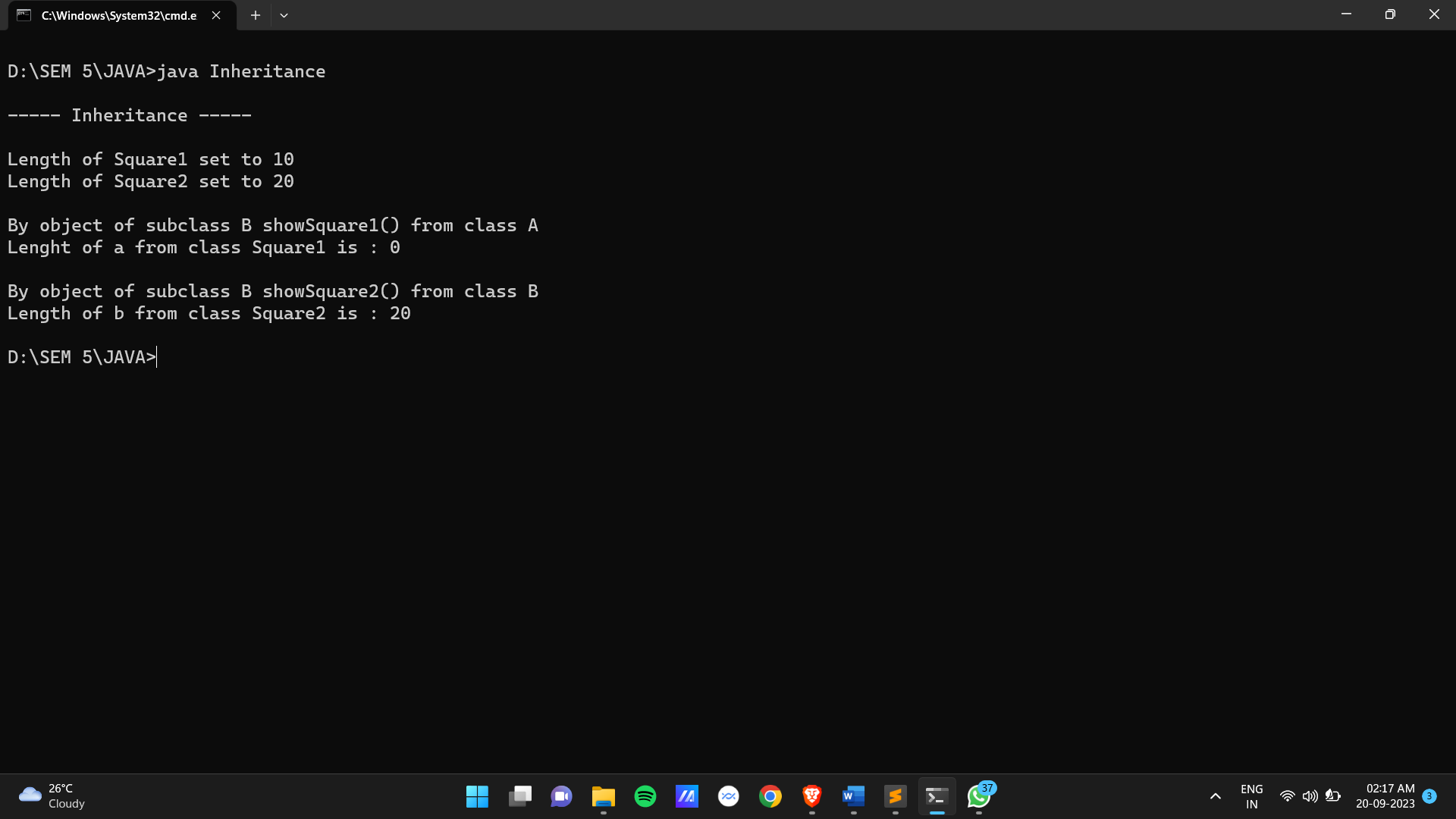
System.out.println("\nBy object of subclass B showSquare2() from class B");

objB.showSquare2();

}

}

**Output :**



**Date:**

**Practical 15:** Write a program which shows use of final and abstract keyword.

**Code :**

abstract class One {

final int charge = 10;

public abstract void printBankName();

final public void reserve() {

System.out.println("Reserve Bank has 1000 crore in reserve");

}

}

final class BankA extends One {

public void printBankName() {

System.out.println("\nThe Bank name is : Bank of Baroda");

System.out.println("Charge for transaction is : " + charge);

}

}

class BankB extends One {

public void printBankName() {

System.out.println("\nThe Bank name is : Panjab National Bank");

System.out.println("Charge for transaction is : " + charge);

}

}

class Keyword {

public static void main(String[] args) {

System.out.println("\n----- Use Of Keywords -----");

System.out.println("\nCharge set to 10 rupees which cannot be immutable by any class");

System.out.println("Class BankA is set as final, it can not be inherit");

System.out.println("Class BankA has reserve method as final, it can not be override");

System.out.println("\nThe abstract method in class One is implement by class BankA & BankB");

System.out.println("The abstract method in class One make class One abstract");

BankA obj1 = new BankA();

obj1.printBankName();

obj1.reserve();

BankB obj2 = new BankB();

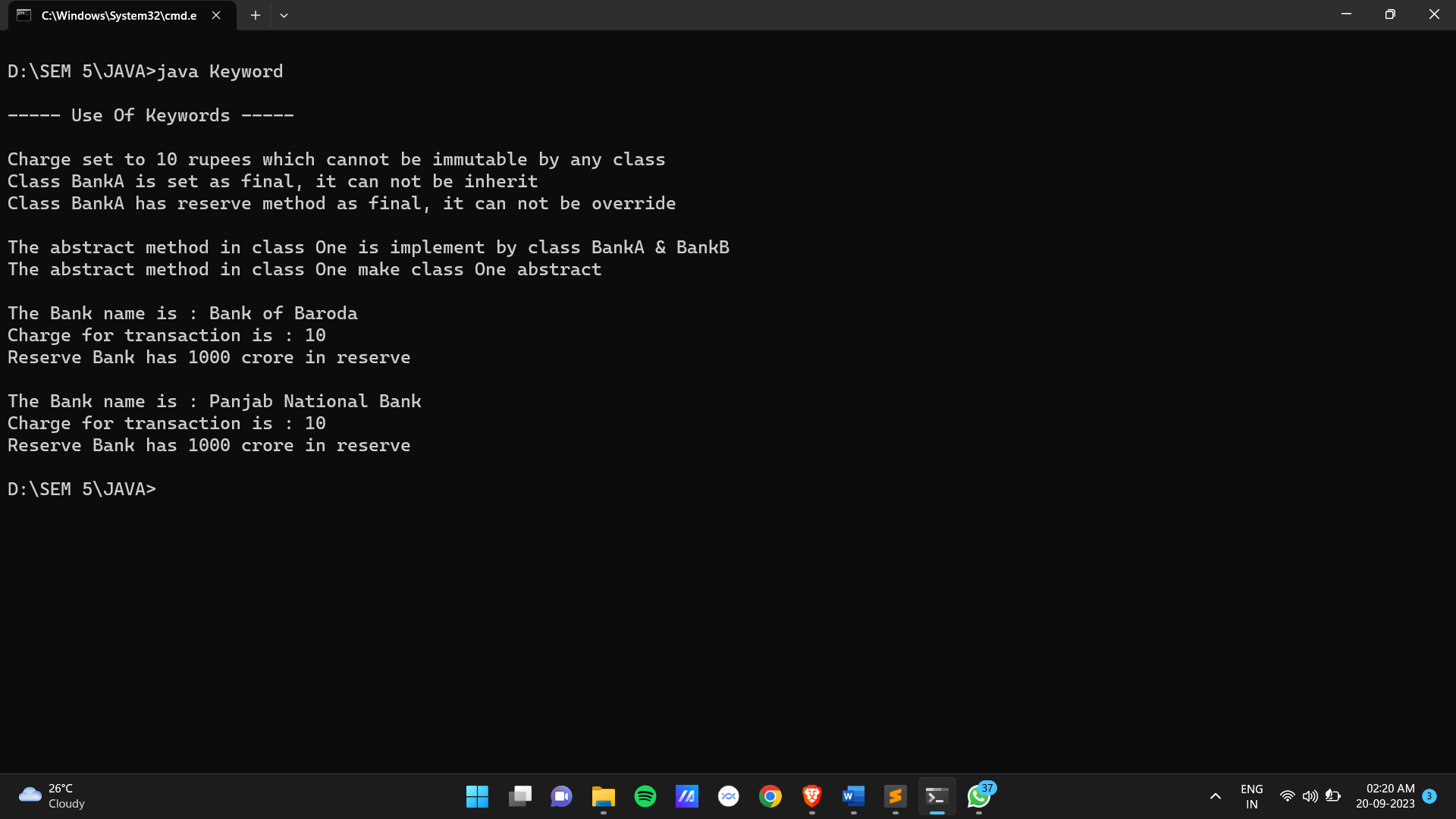
obj2.printBankName();

obj1.reserve();

}

}

**Output :**



**Date:**

**Practical 16:** Write a program which implement addition and subtraction for complex number.

**Code :**

import java.util.\*;

class ComplexNumbers {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("\n----- Complex Numbers -----");

System.out.println("\n1 : Addition\n2 : Subtraction\n\nEnter your choice : ");

int choice = sc.nextInt();

if(choice >= 1 && choice <= 3) {

System.out.println("\nEnter the number 1 : ");

int num1\_real = sc.nextInt();

System.out.println("\nEnter the number 1 complex part : ");

int num1\_imagi = sc.nextInt();

System.out.println("\nEnter the number 2 : ");

int num2\_real = sc.nextInt();

System.out.println("\nEnter the number 2 complex part : ");

int num2\_imagi = sc.nextInt();

System.out.println("\nComplex Number 1 : " + num1\_real + " , " + num1\_imagi + "i");

System.out.println("Complex Number 2 : " + num2\_real + " , " + num2\_imagi + "i");

if(choice == 1) {

int sum\_real = num1\_real + num2\_real;

int sum\_imagi = num1\_imagi + num2\_imagi;

System.out.println("\nAddition : " + sum\_real + " + " + sum\_imagi + "i");

} else {

int sub\_real = num1\_real - num2\_real;

int sub\_imagi = num1\_imagi - num2\_imagi;

System.out.println("\nSubtraction : " + sub\_real + " - " + sub\_imagi + "i");

}

} else {

System.out.println("\n\nInvalid Input");

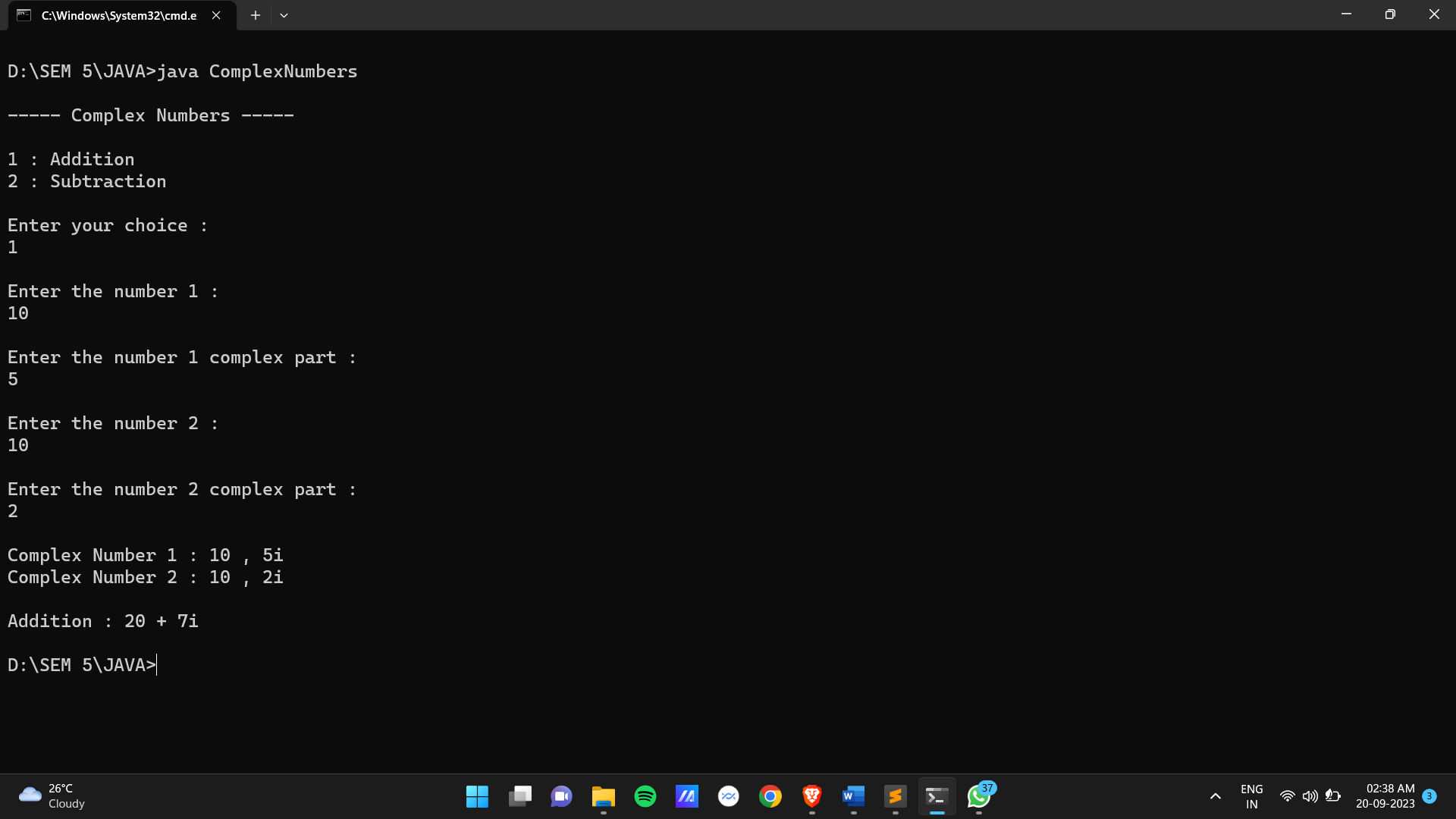
}

}

}

**Output :**

A screenshot of a computer

Description automatically generated 

**Date:**

**Practical 17:** Write a program to implement singly linked list.

**Code :**

import java.util.\*;

class Node {

int data;

Node next;

Node() {

data = 0;

next = null;

}

Node(int d\_data, Node d\_obj) {

this.data = d\_data;

this.next = d\_obj;

}

void append(Node obj) {

Node temp = this;

while(temp.next != null) {

temp = temp.next;

}

temp.next = obj;

}

void display() {

System.out.println("\nData of linked list is : ");

Node temp = this;

while(temp.next != null) {

System.out.print(temp.data + " ");

temp = temp.next;

}

System.out.print(temp.data + " ");

}

Node insert(int type, int data, Node obj) {

Node temp = this;

boolean flag = true;

if(type == 1) {

if(temp.data != data) {

while(temp.next.data != data) {

temp = temp.next;

if(temp.next == null) {

flag = false;

break;

}

}

if(flag == false) {

System.out.println("\nThe " + data + " is not available in linked list");

} else {

obj.next = temp.next;

temp.next = obj;

}

} else {

obj.next = temp;

return obj;

}

} else if(type == 2) {

while(temp.data != data) {

temp = temp.next;

if(temp == null) {

flag = false;

break;

}

}

if(flag == false) {

System.out.println("\nThe " + data + " is not available in linked list");

} else {

obj.next = temp.next;

temp.next = obj;

}

}

return this;

}

Node delete(int data) {

Node temp = this;

boolean flag = true;

if(temp.data != data) {

while(temp.next.data != data) {

temp = temp.next;

if(temp.next == null) {

flag = false;

break;

}

}

if(flag == false) {

System.out.println("\nThe " + data + " is not available in linked list");

} else {

temp.next = temp.next.next;

}

return this;

} else {

return temp.next;

}

}

void search(int data) {

int count = 0;

Node temp = this;

boolean flag = true;

while(temp.data != data) {

temp = temp.next;

count++;

if(temp == null) {

flag = false;

break;

}

}

if(flag == false) {

System.out.println("\nThe " + data + " is not available in linked list");

} else {

System.out.println("\nThe " + data + " is founded in linked list at : " + (count + 1));

}

}

}

class SinglyLinkedList {

public static void main(String[] args) {

int n, choice, data;

boolean repe = true;

Scanner sc = new Scanner(System.in);

System.out.println("\n----- Singly LinkedList -----");

System.out.println("Enter Number of Nodes to Insert : ");

n = sc.nextInt();

Node obj[] = new Node[n];

int num = 1;

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

System.out.println("\nEnter element "+ num +" : ");

data = sc.nextInt();

num++;

obj[i] = new Node(data, null);

if(i > 0) {

obj[0].append(obj[i]);

}

}

obj[0].display();

while(repe) {

System.out.print("\n\n1 : Insert \n2 : Delete \n3 : Search \n4 : Display");

choice = sc.nextInt();

if(choice == 1) {

int ele, data\_choice, data\_pos;

System.out.print("\nEnter the element to insert : ");

ele = sc.nextInt();

Node new\_node = new Node(ele, null);

System.out.print("\n1 : Before Element\n2 : After Element");

data\_choice = sc.nextInt();

if(data\_choice == 1) {

System.out.print("\nEnter the Element : ");

data\_pos = sc.nextInt();

Node first\_node = obj[0].insert(1, data\_pos, new\_node);

obj[0] = first\_node;

obj[0].display();

} else if(data\_choice == 2) {

System.out.print("\nEnter the Element : ");

data\_pos = sc.nextInt();

Node first\_node = obj[0].insert(2, data\_pos, new\_node);

obj[0] = first\_node;

obj[0].display();

} else {

System.out.println("\nInvalid Input");

}

} else if(choice == 2) {

int ele;

System.out.print("\nEnter the element to Delete : ");

ele = sc.nextInt();

Node first\_node = obj[0].delete(ele);

obj[0] = first\_node;

obj[0].display();

} else if(choice == 3) {

int ele;

System.out.print("\nEnter the element to Search : ");

ele = sc.nextInt();

obj[0].search(ele);

} else if(choice == 4) {

obj[0].display();

repe = false;

} else {

System.out.println("\nInvalid Input");

repe = false;

}

}

}

}

**Output :**

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

**Date:**

**Practical 18:** Write a program to implement Circular singly linked list.

**Code :**

import java.util.\*;

class Node{

int data;

Node next;

void append(Node obj){

Node temp = this;

if(this.next==null){

temp.next=obj;

obj.next=temp;

}else{

while(temp.next.data != this.data){

temp=temp.next;

}

temp.next=obj;

obj.next=this;

}

}

void display(){

Node temp=this;

if(temp.next==null){

System.out.print(temp.data+" ");

}else{

while(temp.next.data != this.data){

System.out.print(temp.data+" ");

temp=temp.next;

}

System.out.print(temp.data+" ");

}

}

boolean search(int data){

Node temp=this;

while(temp.data != data){

temp=temp.next;

if(temp.data!=data && temp.next.data==this.data){

return false;

}

}

return true;

}

Node insert(Node obj,int data,int pos){

Node temp=this;

if(pos==1){

if(temp.next==null){

obj.next=temp;

temp.next=obj;

System.out.println("Node inserted successfully");

return obj;

}else{

while(temp.next.data!=data){

temp=temp.next;

}

obj.next=temp.next;

temp.next=obj;

System.out.println("Node inserted successfully");

}

}else if(pos==2){

if(this.next==null){

append(obj);

}else{

while(temp.data != data){

temp=temp.next;

}

if(temp.next.data==this.data){

obj.next=this;

temp.next=obj;

}else{

obj.next=temp.next;

temp.next=obj;

}

System.out.println("Node inserted successfully");

return this;

}

}

return this;

}

Node delete(int val){

Node temp = this;

Node head = this;

if(head.next==null && head.data==val){

head=null;

System.out.println("Node deleted successfully");

return head;

}else{

if(head.data == val){

while(head.next.data != this.data){

head=head.next;

}

head.next=temp.next;

System.out.println("Node deleted successfully");

return head.next;

}else{

while(temp.next.data != val){

temp=temp.next;

}

temp.next=temp.next.next;

System.out.println("Node deleted successfully");

return this;

}

}

}

}

class SinglyCircularLinkedList{

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("-----Singly Circular LinkedList-----\n");

System.out.print("Enter size of linked-list : ");

int size = sc.nextInt();

Node[] list = new Node[size];

int choice;

int inc=1;

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

list[i]=new Node();

System.out.print("Enter Element "+ inc +" : ");

inc++;

if(i>0){

list[i].data=sc.nextInt();

list[0].append(list[i]);

}else{

list[0].data=sc.nextInt();

list[0].next=null;

}

}

do{

System.out.println("1 : Append\n2 : Insert\n3 : Delete\n4 : Search\n5 : Display");

System.out.print("your choice : ");

choice=sc.nextInt();

Node temp=new Node();

int pos,data;

boolean ans;

switch (choice) {

case 1:

System.out.print("\nEnter Value to append : ");

temp.data=sc.nextInt();

list[0].append(temp);

System.out.println("Node appended successfully");

break;

case 2:

System.out.print("\nEnter node data : ");

data=sc.nextInt();

ans = list[0].search(data);

if(ans){

System.out.print("\nEnter element to insert : ");

temp.data=sc.nextInt();

do{

System.out.println("\n1 : Before\n2 : After");

pos = sc.nextInt();

}while(pos>2);

if(pos==1){

list[0]=list[0].insert(temp,data,pos);

}else if(pos==2){

list[0]=list[0].insert(temp,data,pos);

}

}else{

System.out.print("\nElement not found\n");

}

break;

case 3:

System.out.print("\nEnter element to delete: ");

data=sc.nextInt();

ans = list[0].search(data);

if(ans){

list[0]=list[0].delete(data);

}else{

System.out.print("\nNode not found\n");

}

break;

case 4:

System.out.print("\nEnter element for search : ");

data = sc.nextInt();

boolean res = list[0].search(data);

if(res){

System.out.print("\nNode found in linked list\n");

}else{

System.out.print("\nNode not found in linked list\n");

}

break;

case 5:

list[0].display();

break;

default:

System.out.println("Invalid input\n");

}

}while(choice!=6);

}

}

**Output :**

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

**Date:**

**Practical 19:** Write a program to implement Doubly linked list.

**Code :**

import java.util.\*;

class Node {

int data;

Node next;

Node pre;

Node() {

data = 0;

next = null;

pre = null;

}

Node(int d\_data, Node d\_obj, Node d\_pre\_obj) {

this.data = d\_data;

this.next = d\_obj;

this.pre = d\_pre\_obj;

}

void append(Node obj) {

Node temp = this;

while(temp.next != null) {

temp = temp.next;

}

temp.next = obj;

obj.pre = temp;

}

void display() {

System.out.println("\nLinked list : ");

Node temp = this;

while(temp.next != null) {

System.out.print(temp.data + " ");

temp = temp.next;

}

System.out.print(temp.data + " ");

System.out.println("\nLinked list by pre : ");

while(temp.pre != null) {

System.out.print(temp.data + " ");

temp = temp.pre;

}

System.out.print(temp.data + " ");

}

Node insert(int type, int data, Node obj) {

Node temp = this;

boolean flag = true;

if(type == 1) {

if(temp.data != data) {

while(temp.next.data != data) {

temp = temp.next;

if(temp.next == null) {

flag = false;

break;

}

}

if(flag == false) {

System.out.println("\nThe " + data + " not Found!");

} else {

temp.next.pre = obj;

obj.next = temp.next;

temp.next = obj;

obj.pre = temp;

}

} else {

obj.next = temp;

temp.pre = obj;

return obj;

}

} else if(type == 2) {

while(temp.data != data) {

if(temp.next == null) {

flag = false;

break;

}

temp = temp.next;

}

if(flag == false) {

System.out.println("\nThe " + data + " not Found!");

} else {

obj.next = temp.next;

obj.pre = temp;

temp.next = obj;

}

}

return this;

}

Node delete(int data) {

Node temp = this;

boolean flag = true, flag2 = true;

if(temp.data != data) {

while(temp.next.data != data) {

temp = temp.next;

if(temp.next == null) {

flag = false;

break;

}

if(temp.next.next == null) {

flag2 = false;

break;

}

}

if(flag == false) {

System.out.println("\nThe " + data + " not Found!");

} else if(flag2 == false) {

temp.next = null;

} else {

temp.next.next.pre = temp;

temp.next = temp.next.next;

}

return this;

} else {

temp.next.pre = null;

return temp.next;

}

}

void search(int data) {

int count = 0;

Node temp = this;

boolean flag = true;

while(temp.data != data) {

temp = temp.next;

count++;

if(temp == null) {

flag = false;

break;

}

}

if(flag == false) {

System.out.println("\nThe " + data + " not Found!");

} else {

System.out.println("\nThe " + data + "found at : " + (count + 1));

}

}

}

class DoublyLinkedList {

public static void main(String[] args) {

int n, choice, data;

boolean repe = true;

Scanner sc = new Scanner(System.in);

System.out.println("\n-----Doubly Linked List-----");

System.out.println("Enter size of List : ");

n = sc.nextInt();

Node obj[] = new Node[n];

int inc=1;

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

System.out.println("\nEnter Element "+ inc +" : ");

data = sc.nextInt();

inc++;

obj[i] = new Node(data, null, null);

if(i > 0) {

obj[0].append(obj[i]);

}

}

obj[0].display();

while(repe) {

System.out.print("\n\n1 : Insert\n2 : Delete \n3 : Search\n4 : Display\n");

choice = sc.nextInt();

if(choice == 1) {

int ele, data\_choice, data\_pos;

System.out.print("\nEnter Element to insert : ");

ele = sc.nextInt();

Node new\_node = new Node(ele, null, null);

System.out.print("\n1 : Before data\n2 : After data\n");

data\_choice = sc.nextInt();

if(data\_choice == 1) {

System.out.print("\nEnter Element : ");

data\_pos = sc.nextInt();

Node first\_node = obj[0].insert(1, data\_pos, new\_node);

obj[0] = first\_node;

obj[0].display();

} else if(data\_choice == 2) {

System.out.print("\nEnter Element : ");

data\_pos = sc.nextInt();

Node first\_node = obj[0].insert(2, data\_pos, new\_node);

obj[0] = first\_node;

obj[0].display();

} else {

System.out.println("\nInvalid Input");

}

} else if(choice == 2) {

int ele;

System.out.print("\nEnter the element to delete : ");

ele = sc.nextInt();

Node first\_node = obj[0].delete(ele);

obj[0] = first\_node;

obj[0].display();

} else if(choice == 3) {

int ele;

System.out.print("\nEnter Element : ");

ele = sc.nextInt();

obj[0].search(ele);

} else if(choice == 4) {

obj[0].display();

} else if(choice == 5) {

repe = false;

} else {

System.out.println("\nInvalid Input");

repe = false;

}

}

}

}

**Output :**

A screenshot of a computer

Description automatically generated A screenshot of a computer

Description automatically generated

**Date:**

**Practical 20:** Write a program to implement Circular Doubly linked list.

**Code :**

import java.util.\*;

class Node {

int data;

Node next;

Node pre;

Node() {

data = 0;

next = null;

pre = null;

}

Node(int d\_data, Node d\_obj, Node d\_pre\_obj) {

this.data = d\_data;

this.next = d\_obj;

this.pre = d\_pre\_obj;

}

void append(Node obj) {

Node temp = this;

if(temp.next != null) {

while(temp.next != this) {

temp = temp.next;

}

}

temp.next = obj;

obj.pre = temp;

obj.next = this;

this.pre = obj;

}

void display() {

System.out.println("\nLinked list : ");

Node temp = this;

while(temp.next != this) {

System.out.print(temp.data + " ");

temp = temp.next;

}

System.out.print(temp.data + " ");

Node end = temp;

System.out.println("\nLinked list by pre : ");

while(temp.pre != end) {

System.out.print(temp.data + " ");

temp = temp.pre;

}

System.out.print(temp.data + " ");

}

Node insert(int type, int data, Node obj) {

Node temp = this;

boolean flag = true;

if(type == 1) {

if(temp.data != data) {

while(temp.next.data != data) {

temp = temp.next;

if(temp == this) {

flag = false;

break;

}

}

if(flag == false) {

System.out.println("\n" + data + " not Found! ");

} else {

obj.next = temp.next;

obj.pre = temp;

temp.next = obj;

temp.next.pre = obj;

}

} else {

while(temp.next != this) {

temp = temp.next;

}

temp.next = obj;

obj.pre = temp;

obj.next = this;

this.pre = obj;

return obj;

}

} else if(type == 2) {

while(temp.data != data) {

temp = temp.next;

if(temp == this) {

flag = false;

break;

}

}

if(flag == false) {

System.out.println("\n" + data + " not Found! ");

} else {

temp.next.pre = obj;

obj.next = temp.next;

obj.pre = temp;

temp.next = obj;

}

}

return this;

}

Node delete(int data) {

Node temp = this;

boolean flag = true;

if(temp.data != data) {

while(temp.next.data != data) {

temp = temp.next;

if(temp.next == this) {

flag = false;

break;

}

}

if(flag == false) {

System.out.println("\n" + data + " not Found! ");

} else {

temp.next.next.pre = temp;

temp.next = temp.next.next;

}

return this;

} else {

while(temp.next != this) {

temp = temp.next;

}

temp.next = this.next;

this.next.pre = temp;

return this.next;

}

}

void search(int data) {

int count = 0;

Node temp = this;

boolean flag = true;

while(temp.data != data) {

temp = temp.next;

count++;

if(temp == this) {

flag = false;

break;

}

}

if(flag == false) {

System.out.println("\n" + data + " not Found! ");

} else {

System.out.println("\n " + data + " is found at : " + (count + 1));

}

}

}

class DoublyCircularLinkedList {

public static void main(String[] args) {

int n, choice, data;

boolean repe = true;

Scanner sc = new Scanner(System.in);

System.out.println("\n----- Doubly Circular Linked List -----");

System.out.println("Enter size of List : ");

n = sc.nextInt();

Node obj[] = new Node[n];

int inc=1;

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

System.out.println("\nEnter Ellement " + inc + " : ");

data = sc.nextInt();

inc++;

obj[i] = new Node(data, null, null);

if(i > 0) {

obj[0].append(obj[i]);

}

}

obj[0].display();

while(repe) {

System.out.print("\n\n1 : Insert\n2 : Delete\n3 : Search\n4 : Display");

choice = sc.nextInt();

if(choice == 1) {

int ele, data\_choice, data\_pos;

System.out.print("\nEnter element to insert : ");

ele = sc.nextInt();

Node new\_node = new Node(ele, null, null);

System.out.print("\n1 : Before data\n2 : After data\n");

data\_choice = sc.nextInt();

if(data\_choice == 1) {

System.out.print("\nEnter Element : ");

data\_pos = sc.nextInt();

Node first\_node = obj[0].insert(1, data\_pos, new\_node);

obj[0] = first\_node;

obj[0].display();

} else if(data\_choice == 2) {

System.out.print("\nEnter Element : ");

data\_pos = sc.nextInt();

Node first\_node = obj[0].insert(2, data\_pos, new\_node);

obj[0] = first\_node;

obj[0].display();

} else {

System.out.println("\nInvalid Input");

}

} else if(choice == 2) {

int ele;

System.out.print("\nEnter element to delete : ");

ele = sc.nextInt();

Node first\_node = obj[0].delete(ele);

obj[0] = first\_node;

obj[0].display();

} else if(choice == 3) {

int ele;

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

ele = sc.nextInt();

obj[0].search(ele);

} else if(choice == 4) {

obj[0].display();

} else {

System.out.println("\nInvalid Input");

repe = false;

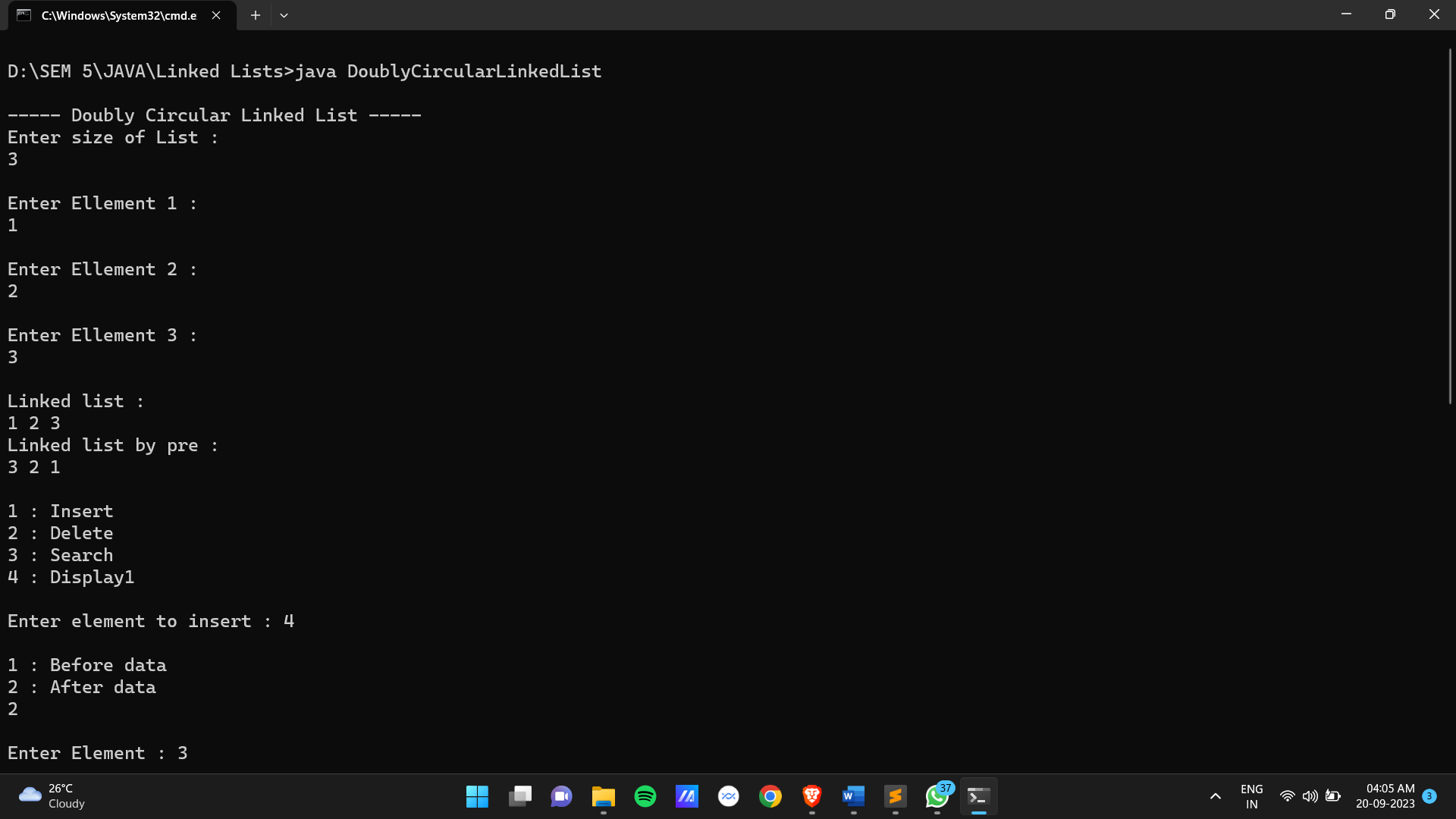
}

}

}

}

**Output :**

 A screenshot of a computer

Description automatically generated

**Date:**

**Practical 21:** Write a program to implement access specification using package.

**Code :**

**Date:**

**Practical 22:** Write a program to implement user (Custom) exception subclass.

**Code :**

import java.util.\*;

class InvalidAgeException extends Exception {

String msg;

InvalidAgeException(String d\_msg) {

msg = d\_msg;

}

public String toString() {

return msg;

}

}

class Person {

String name;

int age;

Person(String name, int age) {

this.name = name;

this.age = age;

}

public void validAge() throws InvalidAgeException {

InvalidAgeException obj = new InvalidAgeException("");

int flag = 1;

if(!(age > 1 && age < 120)) {

if(!(name.startsWith("RAM"))) {

obj = new InvalidAgeException("The age and name both are invalid");

flag = 0;

} else {

obj = new InvalidAgeException("The age is invalid");

flag = 0;

}

} else {

if(!(name.startsWith("RAM"))) {

obj = new InvalidAgeException("The name is invalid");

flag = 0;

}

}

if(flag == 0) {

throw obj;

} else {

System.out.println("\n----- Entered Details -----");

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

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

System.out.println("\nBoth are valid details");

}

}

}

class CustomException {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("\n----- Custom Exception -----");

System.out.println("\nEnter name for the person : ");

String name = sc.nextLine();

System.out.println("Enter age for the person : ");

int age = sc.nextInt();

try {

Person object = new Person(name, age);

object.validAge();

} catch(InvalidAgeException e) {

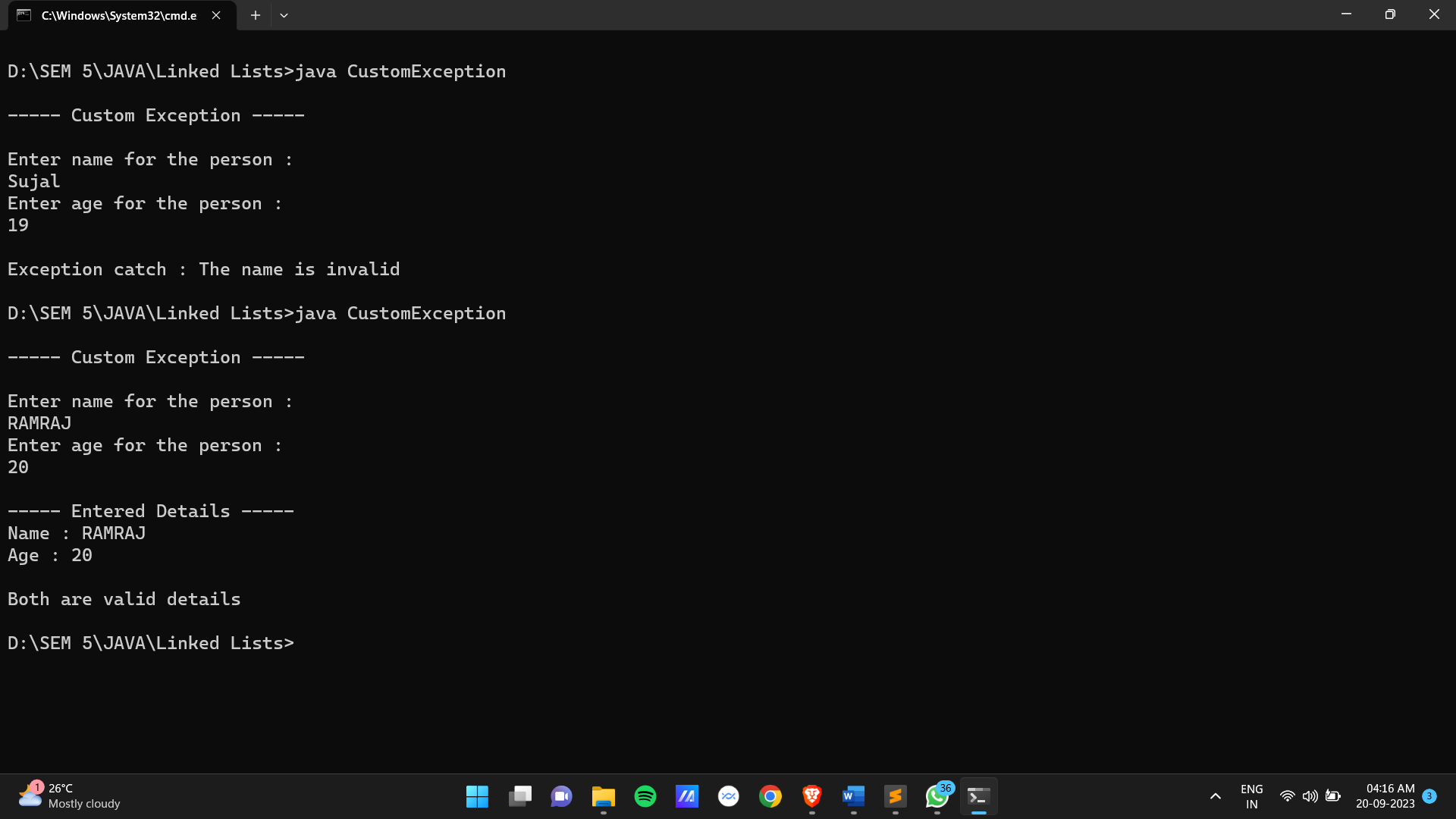
System.out.println("\nException catch : " + e);

}

}

}

**Output :**



**Date:**

**Practical 23:** Write a program to implement threads by implementing Runnable class and by extending Thread class.

**Code :**

**Using Runnable Interface :-**

class NewThread implements Runnable{

Thread t1;

NewThread(){

t1 = new Thread(this,"Child Thred");

t1.start();

}

public void run(){

try{

for(int i=0;i<15;i++){

System.out.println(t1.getName()+" : "+i);

t1.sleep(500);

}

}catch(InterruptedException e){

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

}

}

}

class Threads{

public static void main(String[] args) {

System.out.println("------Thread with Run------");

new NewThread();

Thread t = Thread.currentThread();

try{

for(int i=15;i<16;i++){

System.out.println(t.getName()+" : "+i);

t.sleep(1000);

}

}catch(InterruptedException e){

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

}

}

}

**Using extends Thread :-**

import java.util.\*;

class NewThread extends Thread {

int from, to;

NewThread(int from, int to) {

this.from = from;

this.to = to;

setName("Prime/ArmStrong Thread");

start();

}

public void run() {

try {

for(int i = from; i < to; i++) {

int len = (int)Math.log10(i) + 1;

int sum = 0;

int temp = i;

while(temp > 0) {

int rem = temp % 10;

sum += Math.pow(rem, len);

temp = temp / 10;

}

if(sum == i) {

System.out.println(getName() + " => " + i + " : is Armstrong number");

} else {

System.out.println(getName() + " => " + i + " : is Not Armstrong number");

}

sleep(100);

}

} catch(InterruptedException e) {

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

}

}

}

class MultipleThread2 {

public static void main(String[] args) {

System.out.println(".....Multiple Thread for Prime & Armstrong.....\n\n");

Scanner sc = new Scanner(System.in);

System.out.println("Enter the range for find Prime & Armstrong : ");

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

int from = sc.nextInt();

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

int to = sc.nextInt();

NewThread obj = new NewThread(from, to);

Thread t1 = Thread.currentThread();

t1.setName("Main Thread");

try {

for(int i = from; i < to; i++) {

int flag = 1;

for(int j = 2; j < i / 2; j++) {

if(i % j == 0) {

flag = 0;

}

}

if(flag == 0) {

System.out.println(t1.getName() + " => " + i + " : is Not Prime Number");

} else {

System.out.println(t1.getName() + " => " + i + " : is Prime Number");

}

t1.sleep(100);

}

} catch(InterruptedException e) {

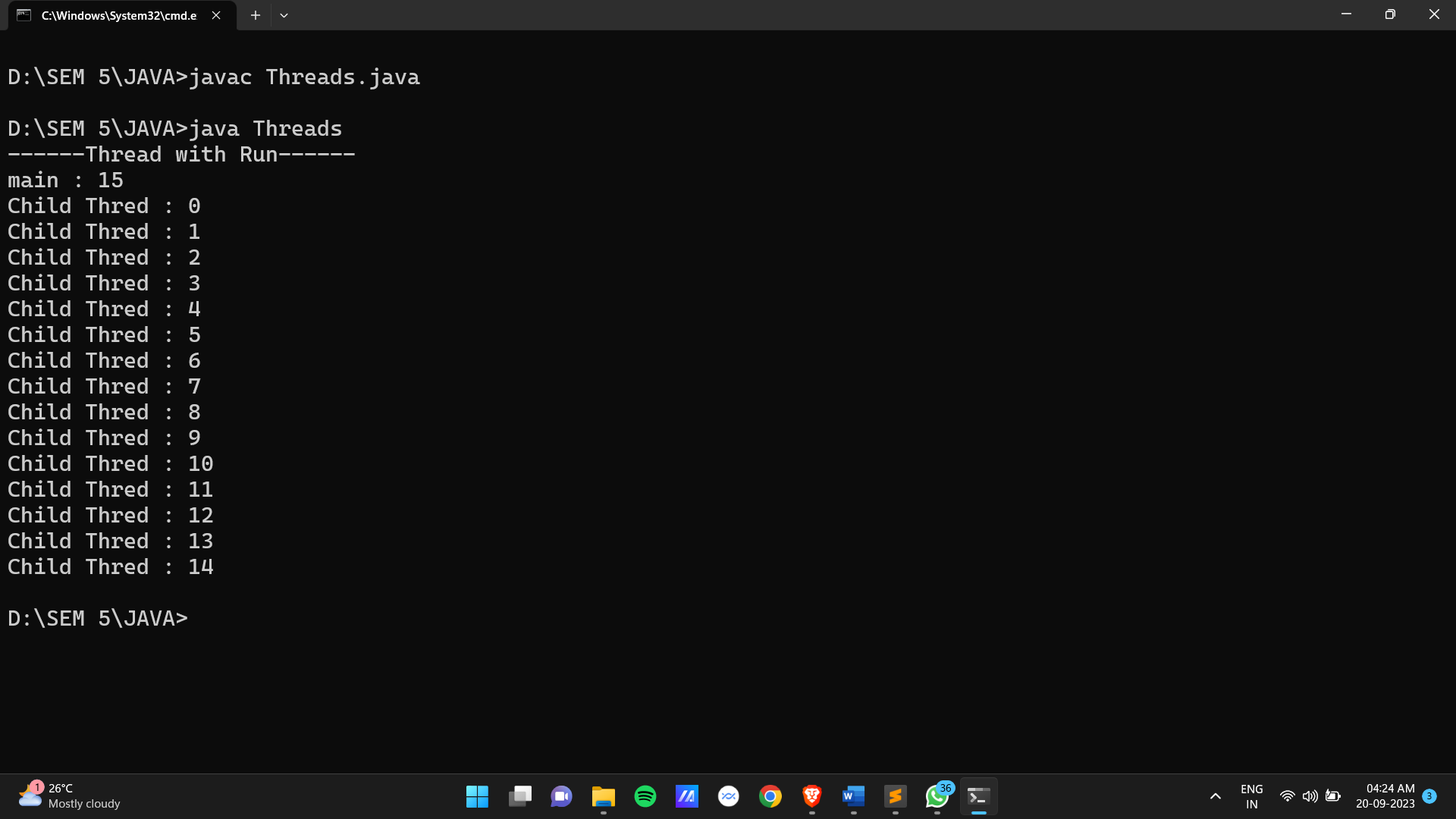
System.out.println("Exception Occured : " + e);

}

}

}

**Output :**

 A screenshot of a computer

Description automatically generated