Program 1. /\*  Develop a JAVA program to add TWO matrices of suitable order N (The value of N should be read from command line arguments). \*/

public class MatrixAddition {

    public static void main(String[] args) {

        // Check if the number of command line arguments is correct

        if (args.length != 1) {

            System.out.println("Usage: java MatrixAddition <order N>");

            return;

        }

        // Parse the order N from command line arguments

        int N = Integer.parseInt(args[0]);

        // Check if N is a positive integer

        if (N <= 0) {

            System.out.println("Please enter a positive integer for N.");

            return;

        }

        // Create two matrices of order N

        int[][] matrixA = generateMatrix(N);

        int[][] matrixB = generateMatrix(N);

        // Print the matrices

        System.out.println("Matrix A:");

        printMatrix(matrixA);

        System.out.println("Matrix B:");

        printMatrix(matrixB);

        // Add the matrices

        int[][] resultMatrix = addMatrices(matrixA, matrixB);

        // Print the result

        System.out.println("Sum of Matrix A and Matrix B:");

        printMatrix(resultMatrix);

    }

    // Function to generate a random matrix of order N

    private static int[][] generateMatrix(int N) {

        int[][] matrix = new int[N][N];

        for (int i = 0; i < N; i++) {

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

                matrix[i][j] = i\*3+j; // Random integer between 0 and 9

            }

        }

        return matrix;

    }

    // Function to add two matrices

    private static int[][] addMatrices(int[][] matrixA, int[][] matrixB) {

        int N = matrixA.length;

        int[][] resultMatrix = new int[N][N];

        for (int i = 0; i < N; i++) {

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

                resultMatrix[i][j] = matrixA[i][j] + matrixB[i][j];

            }

        }

        return resultMatrix;

    }

    // Function to print a matrix

    private static void printMatrix(int[][] matrix) {

        for (int[] row : matrix) {

            for (int value : row) {

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

            }

            System.out.println();

        }

        System.out.println();

    }

}

2. Develop a stack class to hold a maximum of 10 integers with suitable methods. Develop a JAVA main method to illustrate Stack operations.

class Stack {

int top;

int stack[]=new int[10];

Stack() {

top=-1;

}

void push(int item) {

if (top==9) {

System.out.println("Stack is full. Cannot push " + item);

}

else{

stack[++top] = item;

}

}

int pop()

{

if (top<0)

{ System.out.println("Stack underflow. Cannot pop.");

return -1; // Assuming -1 is not a valid element in the stack

}

else

{ return stack[top--];

}

}

void display()

{ if (top < 0) {

System.out.println("Stack is empty.");

} else {

System.out.print("Stack elements: ");

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

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

}

System.out.println();

}

}

}

public class TestStack

{

public static void main(String[] args)

{

Stack mystack1 = new Stack();

Stack mystack2 = new Stack();

for(int i=0;i<10;i++) mystack1.push(i);

for(int i=0;i<10;i++) mystack2.push(i);

mystack1.display();

mystack2.display();

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

for(int i=0;i<10;i++) System.out.print(mystack1.pop()+" ");

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

for(int i=0;i<10;i++) System.out.print(mystack2.pop()+" ");

}

}