import java.util.Scanner;

class MatrixOperation implements Runnable {

    private int[][] matrixA;

    private int[][] matrixB;

    private int[][] result;

    private int row, col;

    private String operation;

    // Constructor

    public MatrixOperation(int[][] matrixA, int[][] matrixB, int[][] result, int row, int col, String operation) {

        this.matrixA = matrixA;

        this.matrixB = matrixB;

        this.result = result;

        this.row = row;

        this.col = col;

        this.operation = operation;

    }

    @Override

    public void run() {

        if (operation.equals("add")) {

            // Perform matrix addition

            result[row][col] = matrixA[row][col] + matrixB[row][col];

        } else if (operation.equals("multiply")) {

            // Perform matrix multiplication

            int sum = 0;

            for (int k = 0; k < matrixB.length; k++) {

                sum += matrixA[row][k] \* matrixB[k][col];

            }

            result[row][col] = sum;

        }

    }

}

public class MultithreadingMatrixOperations {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        // Input dimensions of matrices

        System.out.print("Enter number of rows for Matrix A: ");

        int rowsA = scanner.nextInt();

        System.out.print("Enter number of columns for Matrix A / rows for Matrix B: ");

        int colsA = scanner.nextInt();

        System.out.print("Enter number of columns for Matrix B: ");

        int colsB = scanner.nextInt();

        // Initialize matrices

        int[][] matrixA = new int[rowsA][colsA];

        int[][] matrixB = new int[colsA][colsB];

        int[][] result = new int[rowsA][colsB]; // Result matrix for multiplication

        // Input elements of Matrix A

        System.out.println("Enter elements of Matrix A:");

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

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

                matrixA[i][j] = scanner.nextInt();

            }

        }

        // Input elements of Matrix B

        System.out.println("Enter elements of Matrix B:");

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

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

                matrixB[i][j] = scanner.nextInt();

            }

        }

        // Choose operation

        System.out.print("Choose operation (add/multiply): ");

        String operation = scanner.next();

        // Validate operation and dimensions

        if (operation.equals("add") && (rowsA != colsA || colsA != colsB)) {

            System.out.println("Addition requires both matrices to have the same dimensions.");

            return;

        } else if (operation.equals("multiply") && colsA != matrixB.length) {

            System.out.println("Multiplication requires columns of Matrix A to match rows of Matrix B.");

            return;

        }

        // Create threads for each element in the result matrix

        Thread[][] threads = new Thread[rowsA][colsB];

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

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

                threads[i][j] = new Thread(new MatrixOperation(matrixA, matrixB, result, i, j, operation));

                threads[i][j].start();

            }

        }

        // Wait for all threads to finish

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

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

                try {

                    threads[i][j].join();

                } catch (InterruptedException e) {

                    e.printStackTrace();

                }

            }

        }

        // Display the result matrix

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

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

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

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

            }

            System.out.println();

        }

        scanner.close();

    }

}