Ans6) import java.util.Scanner;

public class SparseMatrixOps {

// Each non-zero element stored as triplet

static class Element {

int row, col, val;

Element(int r, int c, int v) {

row = r;

col = c;

val = v;

}

}

static class SparseMatrix {

int rows, cols, nonZero;

Element[] data;

SparseMatrix(int r, int c, int nz) {

rows = r;

cols = c;

nonZero = nz;

data = new Element[nz];

}

// Display triplet representation

void display() {

System.out.println("Row Col Val");

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

System.out.println(data[i].row + " " + data[i].col + " " + data[i].val);

}

}

// (a) Transpose of sparse matrix

SparseMatrix transpose() {

SparseMatrix result = new SparseMatrix(cols, rows, nonZero);

int k = 0;

for (int c = 0; c < cols; c++) {

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

if (data[i].col == c) {

result.data[k++] = new Element(data[i].col, data[i].row, data[i].val);

}

}

}

return result;

}

// (b) Addition of two sparse matrices

SparseMatrix add(SparseMatrix other) {

if (rows != other.rows || cols != other.cols) {

throw new IllegalArgumentException("Matrix dimensions must match for addition.");

}

Element[] temp = new Element[this.nonZero + other.nonZero];

int i = 0, j = 0, k = 0;

while (i < this.nonZero && j < other.nonZero) {

int r1 = this.data[i].row, c1 = this.data[i].col;

int r2 = other.data[j].row, c2 = other.data[j].col;

if (r1 < r2 || (r1 == r2 && c1 < c2)) {

temp[k++] = new Element(r1, c1, this.data[i].val);

i++;

} else if (r2 < r1 || (r1 == r2 && c2 < c1)) {

temp[k++] = new Element(r2, c2, other.data[j].val);

j++;

} else { // same position

int sum = this.data[i].val + other.data[j].val;

if (sum != 0) {

temp[k++] = new Element(r1, c1, sum);

}

i++; j++;

}

}

while (i < this.nonZero) temp[k++] = new Element(data[i].row, data[i].col, data[i++].val);

while (j < other.nonZero) temp[k++] = new Element(other.data[j].row, other.data[j].col, other.data[j++].val);

SparseMatrix result = new SparseMatrix(rows, cols, k);

for (int m = 0; m < k; m++) result.data[m] = temp[m];

return result;

}

// (c) Multiplication of two sparse matrices

SparseMatrix multiply(SparseMatrix other) {

if (this.cols != other.rows) {

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

}

int[][] result = new int[this.rows][other.cols];

// Compute multiplication

for (int i = 0; i < this.nonZero; i++) {

for (int j = 0; j < other.nonZero; j++) {

if (this.data[i].col == other.data[j].row) {

result[this.data[i].row][other.data[j].col] += this.data[i].val \* other.data[j].val;

}

}

}

// Convert result to sparse triplet

int count = 0;

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

for (int j = 0; j < other.cols; j++) {

if (result[i][j] != 0) count++;

}

}

SparseMatrix product = new SparseMatrix(this.rows, other.cols, count);

int idx = 0;

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

for (int j = 0; j < other.cols; j++) {

if (result[i][j] != 0) {

product.data[idx++] = new Element(i, j, result[i][j]);

}

}

}

return product;

}

}

// Driver code

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

// First sparse matrix

System.out.print("Enter rows, cols, non-zero count of Matrix A: ");

int r1 = sc.nextInt(), c1 = sc.nextInt(), nz1 = sc.nextInt();

SparseMatrix A = new SparseMatrix(r1, c1, nz1);

System.out.println("Enter row col value for A:");

for (int i = 0; i < nz1; i++) {

A.data[i] = new Element(sc.nextInt(), sc.nextInt(), sc.nextInt());

}

// Second sparse matrix

System.out.print("Enter rows, cols, non-zero count of Matrix B: ");

int r2 = sc.nextInt(), c2 = sc.nextInt(), nz2 = sc.nextInt();

SparseMatrix B = new SparseMatrix(r2, c2, nz2);

System.out.println("Enter row col value for B:");

for (int i = 0; i < nz2; i++) {

B.data[i] = new Element(sc.nextInt(), sc.nextInt(), sc.nextInt());

}

// Display

System.out.println("\nMatrix A (triplet form):");

A.display();

System.out.println("\nMatrix B (triplet form):");

B.display();

// Transpose

System.out.println("\nTranspose of A:");

A.transpose().display();

// Addition

if (r1 == r2 && c1 == c2) {

System.out.println("\nA + B:");

A.add(B).display();

} else {

System.out.println("\nAddition not possible (dimension mismatch).");

}

// Multiplication

if (c1 == r2) {

System.out.println("\nA x B:");

A.multiply(B).display();

} else {

System.out.println("\nMultiplication not possible (dimension mismatch).");

}

sc.close();

}

}