5 Diagonal Matrix.

6c. Multiplication of two matrices

#include <iostream>

using namespace std;

int main() {

int row1, col1, row2, col2;

int arr1[100][100], arr2[100][100];

int s1[100][3], s2[100][3], trans[100][3], mult[100][3];

cout << "Enter rows and columns for matrix 1: ";

cin >> row1 >> col1;

cout << "Enter rows and columns for matrix 2: ";

cin >> row2 >> col2;

if (col1 != row2) {

cout << "Multiplication not possible\n";

return 0;

}

cout << "Enter elements of matrix 1:\n";

for (int i = 0; i < row1; i++)

for (int j = 0; j < col1; j++)

cin >> arr1[i][j];

cout << "Enter elements of matrix 2:\n";

for (int i = 0; i < row2; i++)

for (int j = 0; j < col2; j++)

cin >> arr2[i][j];

// Convert matrix 1 to sparse

int k = 1;

for (int i = 0; i < row1; i++)

for (int j = 0; j < col1; j++)

if (arr1[i][j] != 0) {

s1[k][0] = i;

s1[k][1] = j;

s1[k][2] = arr1[i][j];

k++;

}

s1[0][0] = row1;

s1[0][1] = col1;

s1[0][2] = k - 1;

// Convert matrix 2 to sparse

int z = 1;

for (int i = 0; i < row2; i++)

for (int j = 0; j < col2; j++)

if (arr2[i][j] != 0) {

s2[z][0] = i;

s2[z][1] = j;

s2[z][2] = arr2[i][j];

z++;

}

s2[0][0] = row2;

s2[0][1] = col2;

s2[0][2] = z - 1;

// Transpose s2

int x = 1;

for (int col = 0; col < s2[0][1]; col++)

for (int i = 1; i <= s2[0][2]; i++)

if (s2[i][1] == col) {

trans[x][0] = s2[i][1];

trans[x][1] = s2[i][0];

trans[x][2] = s2[i][2];

x++;

}

trans[0][0] = s2[0][1];

trans[0][1] = s2[0][0];

trans[0][2] = s2[0][2];

// Multiply s1 and trans

int m = 1;

for (int i = 1; i <= s1[0][2]; i++) {

for (int j = 1; j <= trans[0][2]; j++) {

if (s1[i][1] == trans[j][1]) {

int row = s1[i][0];

int col = trans[j][0];

int value = s1[i][2] \* trans[j][2];

// Check if already present

int found = 0;

for (int p = 1; p < m; p++) {

if (mult[p][0] == row && mult[p][1] == col) {

mult[p][2] += value;

found = 1;

break;

}

}

if (!found) {

mult[m][0] = row;

mult[m][1] = col;

mult[m][2] = value;

m++;

}

}

}

}

mult[0][0] = row1;

mult[0][1] = col2;

mult[0][2] = m - 1;

// Output result

cout << "Resultant Sparse Matrix (Multiplication):\n";

cout << "Row Col Value\n";

for (int i = 0; i < m; i++)

cout << mult[i][0] << " " << mult[i][1] << " " << mult[i][2] << "\n";

return 0;

}

OUTPUT :

