#include <iostream>

#include <vector>

// Define a structure to represent a sparse matrix element

struct SparseMatrixElement {

int row;

int column;

int value;

};

// Define a class for sparse matrix

class SparseMatrix {

private:

int numRows;

int numColumns;

std::vector<SparseMatrixElement> elements;

public:

SparseMatrix(int rows, int columns) : numRows(rows), numColumns(columns) {}

// Function to add a non-zero element to the sparse matrix

void addElement(int row, int column, int value) {

if (row < 0 || row >= numRows || column < 0 || column >= numColumns) {

std::cout << "Invalid element position." << std::endl;

return;

}

SparseMatrixElement element = {row, column, value};

elements.push\_back(element);

}

// Function to print the sparse matrix

void printMatrix() {

std::cout << "Sparse Matrix:" << std::endl;

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

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

bool found = false;

for (const SparseMatrixElement& element : elements) {

if (element.row == i && element.column == j) {

std::cout << element.value << " ";

found = true;

break;

}

}

if (!found) {

std::cout << "0 ";

}

}

std::cout << std::endl;

}

}

};

int main() {

// Create a sparse matrix with 4 rows and 4 columns

SparseMatrix sparseMatrix(4, 4);

// Add non-zero elements to the matrix

sparseMatrix.addElement(0, 1, 5);

sparseMatrix.addElement(1, 2, 8);

sparseMatrix.addElement(2, 3, 3);

sparseMatrix.addElement(3, 1, 2);

// Print the sparse matrix

sparseMatrix.printMatrix();