#include <iostream>

#include <vector>

using namespace std;

// Define a structure for each non-zero element in the sparse matrix

struct SparseElement {

int row;

int col;

int value;

};

// Define a class for the Sparse Matrix

class SparseMatrix {

private:

int rows;

int cols;

vector<SparseElement> elements;

public:

// Constructor to initialize the sparse matrix with dimensions

SparseMatrix(int r, int c) : rows(r), cols(c) {}

// Function to insert a non-zero element into the sparse matrix

void insertElement(int row, int col, int value) {

// Validate row and column indices

if (row >= 0 && row < rows && col >= 0 && col < cols) {

SparseElement element = {row, col, value};

elements.push\_back(element);

} else {

cout << "Invalid indices for insertion." << endl;

}

}

// Function to display the sparse matrix

void displayMatrix() {

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

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

bool found = false;

for (const auto& element : elements) {

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

cout << element.value << " ";

found = true;

break;

}

}

if (!found) {

cout << "0 ";

}

}

cout << endl;

}

}

};

int main() {

// Create a sparse matrix with 3 rows and 4 columns

SparseMatrix sparse(3, 4);

// Insert non-zero elements into the sparse matrix

sparse.insertElement(0, 1, 5);

sparse.insertElement(1, 2, 8);

sparse.insertElement(2, 3, 3);

// Display the sparse matrix

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

sparse.displayMatrix();

return 0;