```cpp
#include <stdio.h>
#include <iostream>
#include <fstream>
#include <string>
#include <vector>

using namespace std;

/*COSTE:
*
*           k               n = 0
*   T(n):
*           2*T(n/2) + k'       n >= 1
*
*
*   O(n^log2)
*/

//Pre: 0 <= i <= j <= m - 1 && forone k: m = 2^k
int degradado(vector<int> const& lista, int i, int j, bool& solucion) {

    if (i + 1 == j) {

        if (lista[i] >= lista[j]) {
            solucion = false;
        }

        return lista[i] + lista[j];
    }
    else {

        int m = (i + j) / 2;

        bool sol1 = true;
        bool sol2 = true;

        int izq = degradado(lista, i, m, sol1);
        int der = degradado(lista, m + 1, j, sol2);

        if (sol1 && sol2) {

            if (izq >= der) {
                solucion = false;
            }
        }
        else {
            solucion = false;
        }

        return der + izq;
    }
}
```

```
//Post: ((sum u: i <= u <= ((i + j)/2): lista[u]) < (sum w: ((i + j)/2) < w <= j:
lista[w])) && degradado(lista, i, ((i + j)/2 - 1), solucion) && degradado(lista, ((i +
j)/2), j, solucion)

bool resuelveCaso() {

    // Lectura de los datos

    int n, m;
    cin >> n >> m;

    if (!cin) return false;

    vector<vector<int>> lista(n, vector<int>(m));

    int num;

    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {

            cin >> num;
            lista[i][j] = num;
        }
    }

    bool sol = true;

    // Calculo del resultado: una funcion aparte

    int i = 0;
    if (m > 1) {

        while (i < n && sol) {

            degradado(lista[i], 0, m - 1, sol);
            i++;
        }
    }

    // Escritura del resultado
    if (sol) {
        cout << "SI" << endl;
    }
    else {
        cout << "NO" << endl;
    }

    return true;
}


int main() {

    // Para la entrada por fichero.
```

```
#ifndef DOMJUDGE
    std::ifstream in("casos.txt");
    auto cinbuf = std::cin.rdbuf(in.rdbuf());
#endif

    // Resolvemos
    while (resuelveCaso()) {}

#ifndef DOMJUDGE // para dejar todo como estaba al principio
    std::cin.rdbuf(cinbuf);
    system("PAUSE");
#endif

    return 0;
}
```