**MATRIX**

Steven rất thích thú với tour du lịch mà mình tham gia. Nhưng ngay cả đến mặt trời cũng có vết đen. Mọi thứ đều rất hấp dẫn, mới lạ, trừ thời gian ngồi trên xe chuyển từ địa điểm này tới địa điểm khác. Một cách vô thức Steve cho vào túi tìm kiếm và ngạc nhiên khi rút ra được một tờ giấy gấp tư ghi đầy số. Sau một lúc suy nghĩ, Steven nhớ ra đó là ma trận n×m của một bài toán quy hoạch tuyến tính đã làm trước khi đi. Có một vài số trên ma trận đã được sửa lại. Steve nảy ra ý nghĩ, tại sao không sửa tiếp một vài số khác để mỗi hàng và mỗi cột đều thành dãy số palindrome, khi đó không cần mở tờ giấy, chỉ nhìn vào một phần tư của nó cũng biết hết các số còn lại. Hãy xác định số lượng số ít nhất cần sửa để mỗi hàng và mỗi cột của ma trận đều là dãy số palindrome.

**Dữ liệu:** Vào từ file văn bản **MATRIX.INP**:

Dòng đầu tiên chứa 2 số nguyên n và m (1 ≤ n, m ≤ 1000).

Mỗi dòng trong n dòng tiếp theo chứa m số nguyên dương xác định một dòng của ma trận. Các số đều không vượt quá 106.

**Kết quả:** Đưa ra file văn bản **MATRIX.OUT** một số nguyên – số lượng ít nhất các số cần sửa.

**VD:**

|  |  |
| --- | --- |
| matrix.inp | matrix.out |
| 2 4  1 2 2 1  1 2 9 9 | 2 |

**C++ Solution:**

#include <bits/stdc++.h>

using namespace std;

const int N = 1111;

int n, m, c[N][N];

int main() {

    cin >> n >> m;

    for (int i = 0; i < n; ++i)

        for (int j = 0; j < m; ++j)

            cin >> c[i][j];

    int ans = 0;

    for (int i = 0; i < (n + 1) / 2; ++i)

       for (int j = 0; j < (m + 1) / 2; ++j)

            if (i \* 2 + 1 != n && j \* 2 + 1 != m) {

                int a[4] = {c[i][j], c[n-1-i][j], c[i][m-1-j], c[n-1-i][m-1-j]};

                int cans = 3;

                    for (int x : a) {

                        int w = 0;

                        for (int y : a)

                        w += (x != y);

                        cans = min(cans, w);

                    }

                ans += cans;

            }

            else if (i \* 2 + 1 != n && j \* 2 + 1 == m)

                ans += (c[i][j] != c[n-1-i][j]);

            else if (i \* 2 + 1 == n && j \* 2 + 1 != m)

                ans += (c[i][j] != c[i][m-1-j]);

    cout << ans;

}