Matrix Land

235.74 more points to get your next star!

Rank: **26940** | Points: **1964.26/2200**

You have successfully solved Matrix Land Share Tweet

You are now 235.74 points away from the 6th star for your problem solving badge.

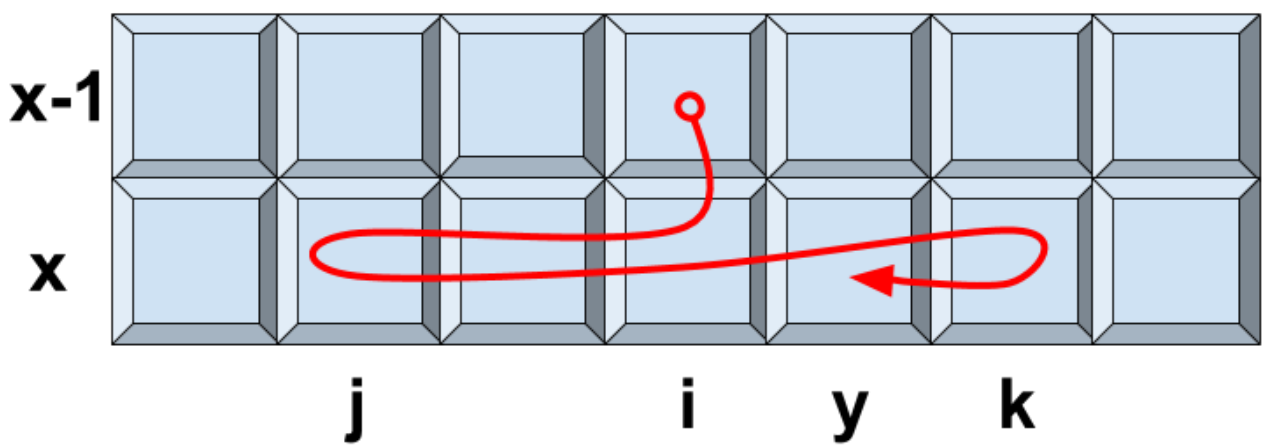
Try the next challenge | **Try a Random Challenge**

Problem Submissions Leaderboard Editorial

Editorial by nikasvanidze

The problem can be solved using DP. Let $dp_{x,y}$ be the maximum sum of numbers with which you can arrive to cell (x, y). Let A be the given matrix.

General way of arriving in (x, y) is (from $(x-1, i)$).



or possibly in a reverse direction i.e go right first and then left.

To calculate best answer some more DPs are being used:

- $msl_{x,y}$ (max sum left) that is maximum sum that you can get by moving only left from (x, y). $msl_{x,y} = max(msl_{x,y-1} + A_{x,y}, 0)$.
- $msr_{x,y}$ (max sum right) that is maximum sum that you can get by moving only right from (x, y). $msl_{x,y} = max(msl_{x,y+1} + A_{x,y}, 0)$.
- $mslit_{x,y}$ (max sum left including top) that is maximum sum that you can arrive from left and also include that yo arrived from top row. $mslit_{x,y} = max(mslit_{x,y-1} + A_{x,y}, dp_{x-1,y} + A_{x,y} + msl_{x,y-1})$.

- $msrit_{x,y}$ (max sum right including top) that is maximum sum that you can arrive from right and also include that you have arrived from top row. $msrit_{x,y} = max(msrit_{x,y+1} + A_{x,y}, dp_{x-1,y} + A_{x,y} + msr_{x,y+1})$.

Finally: $dp_{x,y} = max(mslit_{x,y} + msr_{x,y+1}, msrit_{x,y} + msl_{x,y-1})$

Set by nikasvanidze

Problem Setter's code:

```
#include <bits/stdc++.h>
#define MA(x,y) ((x) > (y) ? (x) : (y))

using namespace std;

const int N = 4000005;

int n, m;

vector <vector <int> > a, dp;
vector <int> msl, msr, d;

void input(){
    scanf("%d %d", &n, &m);

    msl.resize(m+2,0);
    d = msr = msl;

    a.resize(n+2, d);
    dp = a;

    for (int i = 1; i <= n; i++) {
        for (int j = 1; j <= m; j++) {
            scanf("%d", &a[i][j]);
        }
    }
}

void sol(){
    for (int i = 1; i <= n; i++) {
        for (int j = 1; j <= m; j++)
            msl[j] = MA(msl[j-1] + a[i][j],0);
        for (int j = m; 0 < j; j--)
            msr[j] = MA(msr[j+1] + a[i][j],0);

        d[1] = dp[i-1][1] + a[i][1];
        dp[i][1] = d[1] + msr[2];
        for (int j = 2; j <= m; j++) {
            d[j] = MA(d[j-1] + a[i][j], dp[i-1][j] + a[i][j] + msl[j-1]);
            dp[i][j] = d[j] + msr[j + 1];
        }

        d[m] = dp[i-1][m] + a[i][m];
        dp[i][m] = MA(dp[i][m], d[m] + msl[m - 1]);
```

```
        for (int j = m - 1; 0 < j ; j--) {
            d[j] = MA(d[j+1] + a[i][j], dp[i-1][j] + a[i][j] + msr[j+1]);
            dp[i][j] = MA(dp[i][j], d[j] + msl[j - 1]);
        }
    }

    int ans = dp[n][1];
    for (int i = 2; i <= m; i++) {
        ans = MA(ans, dp[n][i]);
    }

    printf("%d\n", ans);
}

int main() {
    input();
    sol();
    return 0;
}
```

Tested by dansagunov

Problem Tester's code:

```
#include <bits/stdc++.h>
#define forn(i,n) for (int i = 0; i < int(n); ++i)
using namespace std;

const int N = int(4e6) + 5;
int dp[2][N];
int a[N], s[N];
int best[N];

int main() {
   int n, m;
   assert(cin >> n >> m);
   assert(1 <= n * m && n * m <= int(4e6));
   memset(dp, 0, sizeof(dp));

   int t = 0;
   forn(_, n) {
      t = !t;

      forn(j, m) {
         assert(scanf("%d", &a[j]) == 1);
         assert(abs(a[j]) <= 250);
      }

      forn(r, 2) {
         s[0] = 0;
         forn(i, m)
            s[i + 1] = s[i] + a[i];
```

```
            best[m] = s[m];
            for (int i = m - 1; i >= 0; --i)
               best[i] = max(s[i], best[i + 1]);

            int mx = -s[0], mxw = -s[0] + dp[!t][0];
            forn(i, m) {
               mxw = max(mxw, mx + dp[!t][i]);

               int val = mxw + best[i + 1];
               if (!r)
                  dp[t][i] = val;
               else
                  dp[t][i] = max(dp[t][i], val);

               mx = max(mx, -s[i + 1]);
            }

            forn(i, 2)
               reverse(dp[i], dp[i] + m);
            reverse(a, a + m);
         }
      }

      cout << *max_element(dp[t], dp[t] + m) << endl;
      return 0;
   }
```

Feedback

Was this editorial helpful?

Yes No