#include <stdio.h>

#include <string.h>

#define inf 1000000001

#define swap(a, b) {a^=b;b^=a;a^=b;}

int n, m;

int c[510][510];

long long ans = 0;

void init()

{

freopen("juice.in", "r", stdin);

freopen("juice.out", "w", stdout);

int i, j;

scanf("%d%d", &m, &n);

for (i = 1; i <= n; i++)

for (j = 1; j <= m; j++)

scanf("%d", c[i]+j);

}

int s;

int h[300010], pl[510][510];

void up(int p)

{

int par, pi, pj, i, j;

i = (h[p]+m-1)/m;

j = 1+(h[p]-1)%m;

while (p > 1)

{

par = p>>1;

pi = (h[par]+m-1)/m;

pj = 1+(h[par]-1)%m;

if (c[i][j] >= c[pi][pj])

return;

swap(h[p], h[par]);

swap(pl[i][j], pl[pi][pj]);

p = par;

}

}

void down(int p)

{

int q, qi, qj, l, li, lj, r, ri, rj, i, j;

i = (h[p]+m-1)/m;

j = 1+(h[p]-1)%m;

while (p < s)

{

q = p;

qi = i;

qj = j;

l = p<<1;

li = (h[l]+m-1)/m;

lj = 1+(h[l]-1)%m;

r = l|1;

ri = (h[r]+m-1)/m;

rj = 1+(h[r]-1)%m;

if (l <= s && c[li][lj] < c[qi][qj])

{

q = l;

qi = li;

qj = lj;

}

if (r <= s && c[ri][rj] < c[qi][qj])

{

q = r;

qi = ri;

qj = rj;

}

if (p == q)

return;

swap(h[p], h[q]);

swap(pl[qi][qj], pl[i][j]);

p = q;

}

}

void init\_heap()

{

int i, j;

for (i = 2; i < n; i++)

for (j = 2; j < m; j++)

{

h[++s] = (i-1)\*m+j;

pl[i][j] = s;

up(s);

}

}

bool t[510][510];

bool del[510][510];

int vi[300010], vj[300010], v;

int si[300010], sj[300010];

int di[4] = {-1, 0, 0, 1};

int dj[4] = {0, -1, 1, 0};

void flood(int p, int q)

{

int sz = 0, i, j, k;

v = 0;

t[p][q] = 1;

si[sz] = p;

sj[sz++] = q;

while (sz)

{

p = si[--sz];

q = sj[sz];

vi[v] = p;

vj[v++] = q;

for (k = 0; k < 4; k++)

{

i = p+di[k];

j = q+dj[k];

if (!t[i][j] && !del[i][j] && c[i][j] >= c[p][q])

{

t[i][j] = 1;

si[sz] = i;

sj[sz++] = j;

}

}

}

}

void solve()

{

bool lim;

int si, sj, l, p, q, i, j, k;

for (i = 0; i <= n+1; i++)

del[i][0] = del[i][m+1] = 1;

for (i = 0; i <= m+1; i++)

del[0][i] = del[n+1][i] = 1;

while (s)

{

p = h[1];

si = (p+m-1)/m;

sj = 1+(p-1)%m;

for (i = 1; i <= n; i++)

for (j = 1; j <= m; j++)

t[i][j] = 0;

flood(si, sj);

l = inf;

lim = 0;

for (p = 0; p < v; p++)

{

i = vi[p];

j = vj[p];

for (k = 0; k < 4 && t[i+di[k]][j+dj[k]]; k++);

if (k < 4 && l > c[i][j])

l = c[i][j];

}

if (l == c[si][sj])

{

for (k = 0; s && k < v; k++)

{

i = vi[k];

j = vj[k];

if (i != 1 && i != n && j != 1 && j != m && c[i][j] == l)

{

del[i][j] = 1;

q = h[pl[i][j]] = h[s--];

si = (q+m-1)/m;

sj = 1+(q-1)%m;

pl[si][sj] = pl[i][j];

down(pl[si][sj]);

}

}

continue;

}

for (p = 0; p < v && !lim; p++)

{

i = vi[p];

j = vj[p];

for (k = 0; k < 4; k++)

{

si = i+di[k];

sj = j+dj[k];

if (del[si][sj] && c[si][sj] >= c[i][j])

lim = 1;

}

}

if (lim)

for (k = 0; k < v; k++)

{

i = vi[k];

j = vj[k];

if (c[i][j] < l)

{

ans += l-c[i][j];

c[i][j] = l;

p = pl[i][j];

q = h[s];

h[p] = h[s--];

pl[(q+m-1)/m][1+(q-1)%m] = p;

del[i][j] = 1;

down(p);

}

}

else

for (k = 0; k < v; k++)

{

i = vi[k];

j = vj[k];

if (c[i][j] < l)

{

ans += l-c[i][j];

c[i][j] = l;

down(pl[i][j]);

}

}

}

}

int main()

{

init();

init\_heap();

solve();

printf("%lld\n", ans);

return 0;

}