№1

#include <iostream>

using namespace std;

int main()

{

int n, m;

cin >> n >> m;

int\*\* dynarr = new int\* [n];

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

{

dynarr[i] = new int[m];

}

int number = 1;

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

{

if (i % 2 == 0)

{

for (int k = 0; k < m; ++k)

{

dynarr[i][k] = number;

++number;

}

}

else

{

for(int k=m-1;k>=0;--k)

{

dynarr[i][k] = number;

++number;

}

}

}

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

{

for (int k = 0; k < m; ++k)

{

cout << dynarr[i][k] << " ";

}

cout << endl;

}

}

№2

#include <iostream>

#include <ctime>

using namespace std;

int number\_length(int n)

{

int digit\_count = 0;

while (n >= 1)

{

n = n / 10;

++digit\_count;

}

return digit\_count;

}

void printSpace(int n)

{

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

{

cout << " ";

}

}

int main()

{

srand(time(NULL));

int n, m;

cin >> n >> m;

int\*\* dynarr = new int\* [n];

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

{

dynarr[i] = new int[m];

}

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

{

for (int k = 0; k < m; ++k)

{

dynarr[i][k] = rand() % 100000;

}

}

int\* maxlength = new int[m];

maxlength[0] = 0;

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

{

for (int k = 0; k < n; ++k)

{

int value = number\_length(dynarr[k][i]);

if (maxlength[i] < value)

{

maxlength[i] = value;

}

}

}

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

{

for (int k = 0; k < m; ++k)

{

cout << dynarr[i][k];

printSpace(maxlength[i] - number\_length(dynarr[i][k]));

cout << " ";

}

cout << endl;

}

}

№6

#include <iostream>

#include <ctime>

using namespace std;

int number\_length(int n)

{

int digit\_count = 0;

while (n >= 1)

{

n = n / 10;

++digit\_count;

}

return digit\_count;

}

void printSpace(int n)

{

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

{

cout << " ";

}

}

int main()

{

srand(time(NULL));

int n, m;

cin >> n >> m;

int\*\* dynarr = new int\* [n];

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

{

dynarr[i] = new int[m];

}

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

{

for (int k = 0; k < m; ++k)

{

dynarr[i][k] = rand() % 100000;

}

}

int\* maxlength = new int[m];

maxlength[0] = 0;

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

{

for (int k = 0; k < n; ++k)

{

int value = number\_length(dynarr[k][i]);

if (maxlength[i] < value)

{

maxlength[i] = value;

}

}

}

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

{

for (int k = 0; k < m; ++k)

{

printSpace(maxlength[i] - number\_length(dynarr[i][k]));

cout << dynarr[i][k];

cout << " ";

}

cout << endl;

}

}

№8

#include <iostream>

using namespace std;

int main()

{

int n, m;

cin >> n >> m;

if (m == 1 && n == 1)

{

cout << "Error" << endl;

return 0;

}

int\*\* dynarr = new int\* [n];

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

{

dynarr[i] = new int[m];

}

int num = 1, i = 0, k = 0, k1 = 1, nm = n \* m, i1, m1=m-1;

bool diagonalIncr = true, diagonalDecr = false, fullarr=false;

for (int diagonal = 1;!fullarr;)

{

if (diagonalIncr)

{

for (int p = 0;; ++p)

{

dynarr[i][k] = num;

if (p + 1 == diagonal)

{

if ((n<=m && diagonal==n) || (n>m&& diagonal==m) )

{

if (n == m)

{

diagonalDecr = true;

diagonalIncr = false;

--diagonal;

++num;

i = 1;

i1 = 2;

k = m - 1;

break;

}

else if (n < m)

{

diagonalIncr = false;

++num;

i = 0;

k = k + diagonal;

break;

}

else

{

diagonalIncr = false;

++num;

i = 1;

k = m - 1;

break;

}

}

else

{

i = 0;

if (k != m1)

{

k = k1;

if (k1 != m1)

{

++k1;

}

}

}

++num;

++diagonal;

break;

}

else

{

++i;

--k;

++num;

}

}

}

else if (!diagonalDecr)

{

if (n < m)

{

int mdiagonal = m - diagonal, a;

while(1)

{

a = dynarr[i][k - 1] + diagonal;

for (int p = 0; ; ++p)

{

dynarr[i][k] = a;

if (p + 1 == mdiagonal)

{

k = k - mdiagonal;

++i;

break;

}

else

{

a = a + diagonal;

++k;

}

}

if (i == n)

{

i = 1;

i1 = 2;

k = m - 1;

num = a + 1;

diagonalDecr = true;

--diagonal;

break;

}

}

}

else

{

int ndiagonal=n-diagonal, temp = i + 1, temp1=temp, a;

for (; k >= 0; --k)

{

a = dynarr[i - 1][k] + diagonal;

for (int p = 0; ; ++p)

{

dynarr[i][k] = a;

if (p + 1 == ndiagonal)

{

if (k == 0)

{

k = m1;

i = temp1;

i1 = i + 1;

diagonalDecr = true;

--diagonal;

num = a + 1;

break;

}

else if (k == m1)

{

temp1 = i + 1;

}

i = temp;

++temp;

break;

}

else

{

a = a + diagonal;

++i;

}

}

if (diagonalDecr)

{

break;

}

}

}

}

else

{

for (int p = 0; ; ++p)

{

dynarr[i][k] = num;

if (p + 1 == diagonal)

{

if (num == nm)

{

fullarr = true;

break;

}

else

{

k = m - 1;

i = i1;

++i1;

--diagonal;

++num;

break;

}

}

else

{

++i;

--k;

++num;

}

}

}

}

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

{

for (int k = 0; k < m; ++k)

{

cout << dynarr[i][k] << '\t';

}

cout << endl;

}

}