#include <iostream>

#include <vector>

using namespace std;

void input(vector <vector<double>> &mas, int n, int m)

{

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

{

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

{

cout << "mas[" << i << "," << j << "]= ";

cin >> mas[i][j];

}

}

}

void output(vector <vector<double>>& mas, int n, int m)

{

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

{

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

{

cout << mas[i][j]<<" ";

}

cout << "\n";

}

}

void north\_west(vector <vector<double>>& mas1, vector <vector<double>>& mas, int n, int m)//заполнение северо-западным углом

{

int s = 0;

int r = 0;

double a;

double b;

int i = 0;

while (s<n-1 || r<m-1)

{

a = mas[s][m - 1];

b = mas[n - 1][r];

cout << a << ' ' << b<<'\n';

if (a < b)

{

mas1[s][r] = a;

mas[s][m - 1] = mas[s][m - 1] - a;

mas[n - 1][r] = mas[n - 1][r] - a;

s++;

}

else

{

mas1[s][r] = b;

mas[s][m - 1] = mas[s][m - 1] - b;

mas[n - 1][r] = mas[n - 1][r] - b;

r++;

}

i++;

}

}

int find\_opor(vector <vector<double>>& mas1, int n, int m, bool flag)

{

int max1=0;

int str=0;

int count=0;

int max2 = 0;

int row = 0;

for (int i = 0; i < n-1; i++)//проход по строкам

{

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

{

if (mas1[i][j] > 0)

{

//cout << mas1[i][j] << ' ';

count++;

}

}

if (max1 < count)

{

max1 = count;

str = i;

}

count = 0;

}

//cout << max1 <<' '<<str << '\n';

for (int j = 0; j < m-1; j++)//проход по столбцам

{

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

{

if (mas1[i][j] > 0)

{

//cout << mas1[i][j] << ' ';

count++;

}

}

cout << '\n';

if (count > max2)

{

max2 = count;

row = j;

}

count = 0;

}

//cout << max2 << ' ' << row << '\n';

if (max1 > max2)

{

flag = true;

}

else

{

flag = false;

}

return (max1 > max2 ? str : row);

}

void zero(vector <vector<double>>& mas, int n, int m) //обнуление матрицы

{

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

{

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

{

mas[i][j]=0;

}

}

}

void check\_output(vector <double>& v, int n)

{

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

{

cout << v[i] << ' ';

}

}

void sostav\_u\_v(vector <vector<double>>& mas, vector <vector<double>>& mas1, vector <double>& v, vector <double>& u, int n, int m, bool flag, int s)

{

if (flag == true)

{

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

{

if (mas1[s][j] != 0)

{

v[j] = mas[s][j];

cout << v[j]<<':'<<j<<' ';

}

}

cout << '\n';

int k = 0;

while (k <= 3)

{

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

{

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

{

if (i != s)

{

if (mas1[i][j] != 0)

{

if (v[j] != 0)

{

cout << v[j] << '-';

u[i] = mas[i][j] - v[j];

cout << u[i] << ':' << i << ' ';

}

}

}

}

}

cout << '\n';

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

{

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

{

if (mas1[i][j] != 0)

{

if (u[i] != 0)

{

v[j] = mas[i][j] - u[i];

}

}

}

}

k++;

}

}

}

void raschet(vector <vector<double>>& mas1, vector <vector<double>>& mas, vector <double>& v, vector <double>& u, int n, int m, int &str, int &row)

{

double min=0;

double temp = 0;

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

{

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

{

if (mas1[i][j] == 0)

{

temp = mas[i][j] - (v[j] + u[i]);

cout << mas[i][j] << "-(" << v[j] << "+" << u[i] << ") = " << temp << '\n';

if (temp < min)

{

min = temp;

str = i;

row = j;

}

}

}

}

cout << "min"<<min << '\n';

// cout << "str: " << str << " row: " << row << '\n';

} //можно вернуть минимальное, если 0, то все закончено

void moving(vector <vector<double>>& mas1, int n, int m, vector <double> &mov, int str, int row)

{

mov.push\_back(mas1[str][row]);

int k = 0;

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

{

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

{

if (mas1[i][j] != 0)

{

if (k < 2)

{

mov.push\_back(mas1[i][j]);

k++;

}

else

{

mov.pop\_back();

mov.push\_back(mas1[i][j]);

}

}

}

k = 0;

}

}

void moving1(vector <vector<double>>& mas1, int n, int m, vector <double>& mov, int str, int row, bool &flag)

{

int st;

for (int i = 0; i < n-1; i++) //находим границы

{

if (mas1[i][row] != 0)

{

st = i;

}

}

if (st < str)

{

mov.push\_back(mas1[str][row]);

mov.push\_back(mas1[st][row]);

flag = false;

}

else

{

mov.push\_back(mas1[st][row]);

mov.push\_back(mas1[str][row]);

flag = true;

}

int a;

int b;

if (st > str)

{

b = st;

a = str;

}

else

{

a = st;

b = str;

}

int k = 0;

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

{

for (int i = a; i <= b; i++)

{

if (mas1[i][j] != 0)

{

//cout << mas1[i][j] << ' ';

mov.push\_back(mas1[i][j]);

k++;

}

}

//cout << '\n';

//cout << k << '\n';

if (k ==1)

{

mov.pop\_back();

}

k = 0;

}

//cout << st << '\n';

//cout << str << '\n';

}

void sub(vector <vector<double>>& mas1, int n, int m, vector <double>& mov, int str, int row, double min)

{

int st;

for (int i = 0; i < n - 1; i++) //находим границы

{

if (mas1[i][row] != 0)

{

st = i;

}

}

int a;

int b;

bool flag = true;

if (st > str)

{

b = st;

a = str;

}

else

{

a = st;

b = str;

flag = false;

}

cout << "a= " << a<<'\n';

cout << "b= " << b<<'\n';

int k = 0;

int min1 = n;

int max1 = 0;

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

{

for (int i = a; i <= b; i++)

{

if (mas1[i][j] != 0)

{

cout << mas1[i][j] << ';';

k++;

}

}

cout << '\n';

if (k == 2)

{

//cout << j<<'\n';

if (j < min1)

{

min1 = j;

}

if (j > max1)

{

max1 = j;

}

}

k = 0;

}

int c;

int d;

if (row < min1)

{

c = row;

d = min1;

}

else

{

c = min1;

d = row;

}

if (row > max1)

{

max1 = row;

}

cout << "c= " << c << '\n';

cout << "d= " << d << '\n';

k = 0;

cout << "matrix for sub:" << '\n';

for (int i = a; i <= b; i++)

{

for (int j = c; j <= max1; j++)

{

if (mas1[i][j] != 0)

{

cout << mas1[i][j] << ' ';

if (k < 2)

{

min = min \* (-1);

mas1[i][j] = mas1[i][j] + min;

k++;

}

else

{

//

mas1[i][j] = mas1[i][j] + min;

mas1[i][j-1] = mas1[i][j-1] - min;

}

}

}

//

cout << '\n';

k = 0;

}

min = min \* (-1);

mas1[str][row] = mas1[str][row] + min;

}

double find\_min(vector <double>& move, bool &flag)

{

double min = 10000;

for (int i = 0; i < move.size(); i++)

{

if (flag == true)

{

if (move[i] < min && i % 2 == 0)

{

//cout<<"min"

min = move[i];

}

}

else

{

if (move[i] < min && i % 2 != 0)

{

//cout<<"min"

min = move[i];

}

}

}

return min;

}

int main()

{

setlocale(LC\_ALL, "rus");

int o{}; //источники

int d{}; //пункты назначения

cout << "Введите кол-во источников: ";

cin >> o;

cout << "Введите кол-во пунктов назначения: ";

cin >> d;

int n = o + 1;//строк

int m = d + 1; //столбцов

vector <vector<double>> mas(n, vector <double>(m)); //массив тарифов

input(mas, n, m);

output(mas, n, m);

vector <vector<double>> mas1(n, vector <double>(m));//для начального заполнения

zero(mas1, n, m);

north\_west(mas1, mas, n, m);

cout << '\n';

output(mas, n, m);

cout << '\n';

output(mas1, n, m);

cout << '\n';

bool flag1 = true;

bool flag = true;

int s;

int st;

int ro;

vector <double> u(n);

vector <double> v(m);

int str = 0;

int row = 0;

vector <double> move;

unsigned int v\_size;

double min;

int z = 0;

s = find\_opor(mas1, n, m, flag);

cout << s << '\n';

//check\_output(v, m);

if (flag == true)

{

u[s] = 0;

}

else

{

v[s] = 0;

}

sostav\_u\_v(mas, mas1, v, u, n, m, flag, s);

check\_output(v, m);

cout << '\n';

check\_output(u, n);

cout << '\n';

raschet(mas1, mas, v, u, n, m, str, row);

cout << str << ' ' << row << '\n';

moving1(mas1, n, m, move, str, row, flag1);

v\_size = move.size();

for (int i = 0; i < v\_size; i++)

{

cout << move[i] << ' ';

}

cout << "\n";

min = find\_min(move, flag1);

cout << min;

cout << "\n";

sub(mas1, n, m, move, str, row, min);

output(mas1, n, m);

cout << "2" << '\n';

cout << '\n';

v\_size = move.size();

for (int i = 0; i < v\_size; i++)

{

move.pop\_back() ;

}

//2

s = find\_opor(mas1, n, m, flag);

cout <<"опорный: " << s << '\n';

if (flag == true)

{

u[s] = 0;

cout << "str" << '\n';

}

else

{

v[s] = 0;

cout << "row" << '\n';

}

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

{

u[i] = 0;

}

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

{

v[j] = 0;

}

check\_output(v, m);

cout << '\n';

check\_output(u, n);

cout << '\n';

sostav\_u\_v(mas, mas1, v, u, n, m, flag, s);

check\_output(v, m);

cout << '\n';

check\_output(u, n);

cout << '\n';

raschet(mas1, mas, v, u, n, m, str, row);

cout <<"str: " << str << " row: " << row << '\n';

moving1(mas1, n, m, move, str, row, flag1);

v\_size = move.size();

for (int i = 0; i < v\_size; i++)

{

cout << move[i] << ' ';

}

cout << "\n";

min = find\_min(move, flag1);

cout <<"min: " << min;

cout << "\n";

sub(mas1, n, m, move, str, row, min);

output(mas1, n, m);

cout << "\n";

s = find\_opor(mas1, n, m, flag);

cout << "опорный: " << s << '\n';

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

{

u[i] = 0;

}

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

{

v[j] = 0;

}

check\_output(v, m);

cout << '\n';

check\_output(u, n);

cout << '\n';

sostav\_u\_v(mas, mas1, v, u, n, m, flag, s);

check\_output(v, m);

cout << '\n';

check\_output(u, n);

cout << '\n';

raschet(mas1, mas, v, u, n, m, str, row);

cout << "str: " << str << " row: " << row << '\n';

moving1(mas1, n, m, move, str, row, flag1);

v\_size = move.size();

for (int i = 0; i < v\_size; i++)

{

cout << move[i] << ' ';

}

cout << "\n";

min = find\_min(move, flag1);

cout << "min: " << min;

cout << "\n";

return 0;

}