STEMgames

STEP

TECHNOLOGY ARENA

Problem 1: Largest triangle

1 second / 512 MiB / 100 points

You are given N points in $2D$ coordinate system, all of which have integer coordinates.

Among all of the triangles whose vertices are among the given points, you must find the largest one by area.

Area of a triangle whose vertices are points $P_1(x_1, y_1)$, $P_2(x_2, y_2)$, $P_3(x_3, y_3)$ is calculated using the formula:

$$2 \cdot Area(P_1P_2P_3) = |x_1(y_2 - y_3) + x_2(y_3 - y_1) + x_3(y_1 - y_2)|.$$

Notice that the area multiplied by 2 is an integer, so you will be asked to **print the largest area multiplied by 2** so that the output is an integer.

Below is a short code sample how the formula would look in C++.

```
struct Point{
    long long x;
    long long y;
};

long long DoubleArea(Point A, Point B, Point C){
    return abs(A.x * (B.y - C.y) + B.x * (C.y - A.y) + C.x * (A.y - B.y));
}
```

Be careful to use **long long** to avoid overflow.

INPUT DATA:

In the first line, there is an integer $3 \leqslant N \leqslant 100000$, the number of points.

In each of the next N lines there are two integers $0 \leqslant x_i, y_i \leqslant 200000$, each pair representing the coordinates of the i-th point.

OUTPUT DATA:

In the first and only line, print one integer: the largest area out of all the triangles formed by the given N points **multiplied by 2**.

SCORING:

First test data cluster is worth 10 points and $3 \leqslant N \leqslant 250$.
Second test data cluster is worth 10 points and $3 \leqslant N \leqslant 600$.
Third test data cluster is worth 30 points and the points are generated randomly (uniformly).
Fourth test data cluster is worth 30 points and $3 \leqslant N \leqslant 6000$.
Fifth test data cluster is worth 20 points and $3 \leqslant N \leqslant 12000$.

TEST SAMPLES:

Input 1:

```
3
0 0
200000 200000
200000 0
```

Output 1:

```
40000000000
```

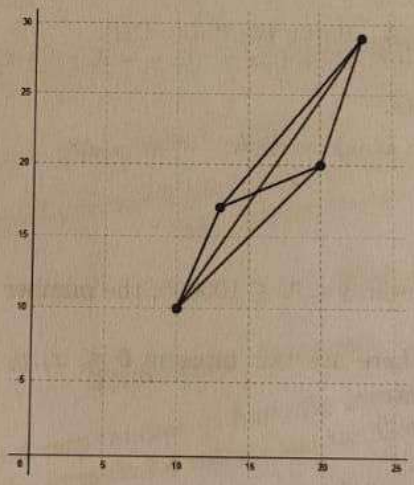
Explanation 1: There is only one triangle of area 20000000000. Its double area is 40000000000 and that is the answer.

Input 2:

```
4
10 10
20 20
13 17
23 29
```

Output 2:

```
60
```



Explanation 2: The triangle formed by points $(10, 10)$, $(20, 20)$ and $(23, 29)$ has the largest area, which is 30.0, so the answer is 60.

STEMgames

TECHNOLOGY ARENA

Problem 2: Undistortion

2 seconds / 512 MiB / 100 points

You are given a point $P_u(x_u, y_u)$ in $2D$ coordinate system and two positive real numbers k_1 and k_2. The numbers k_1 and k_2 are called the **distortion parameters**. With those values, we construct a point $P_d(x_d, y_d)$ with the following formula:

$$r^2 := x_u^2 + y_u^2$$
$$s := 1.0 + k_1 r^2 + k_2 r^4$$
$$x_d := s \cdot x_u, \quad y_d := s \cdot y_u.$$

This is called the distortion formula and point P_d is called the **distorted point** of P_u. Similarly, P_u is called the **undistorted point** of P_d.

Below is a short code sample how the formula would look in C++.

```
pair<float, float> Distort(float k1, float k2, float xu, float yu){
    float r2 = xu * xu + yu * yu;
    float s = 1.0 + r2 * k1 + r2 * r2 * k2;
    return make_pair(xu * s, yu * s);
}
```

Given several examples of k_1, k_2 and $P_d(x_d, y_d)$, for each of them find the corresponding $P_u(x_u, y_u)$.

In other words, given the distortion parameters and the distorted point, determine the corresponding undistorted point P_u which maps to the given P_d by the above formula.

Round the coordinates of P_u to three decimal places. If your output differs from the official output by at most 0.001, your output will be considered correct.

INPUT DATA:

In the first line, there is an integer $3 \leqslant T \leqslant 40000$, the number of test cases.

In each of the next T lines there are four real numbers k_1, k_2, x_d, y_d in that order, with up to three decimal places. They represent the distortion parameters and the coordinates of the distorted point. It will be $0 \leqslant k_1, k_2 \leqslant 2.0$ and $0 \leqslant x_d, y_d \leqslant 15.0$.

OUTPUT DATA:

Print T lines, one for each input test case. In each line, print two real numbers x_u and y_u, the coordinates of the undistorted point corresponding to that test case, rounded to three decimal places.

SCORING:

First test data cluster is worth 10 points and $T \leqslant 10$.
Second test data cluster is worth 20 points and $T \leqslant 5000$.
Third test data cluster is worth 70 points and there are no additional constraints.

TEST SAMPLES:

Input:

```
2
0.05 1.87 1.016 1.181
0.84 1.48 0.417 0.655
```

Output:

```
0.537 0.624
0.298 0.468
```

Explanation:

For the first line, the point $(0.537, 0.624)$ maps to $(1.016, 1.181)$ with $k_1 = 0.05$ and $k_2 = 1.87$, so that is the answer (everything up to 3 decimals).

For the second line, the point $(0.298, 0.468)$ maps to $(0.417, 0.655)$ with $k_1 = 0.84$ and $k_2 = 1.48$, so that is the answer (everything up to 3 decimals).

STEMgames

STME

TECHNOLOGY ARENA

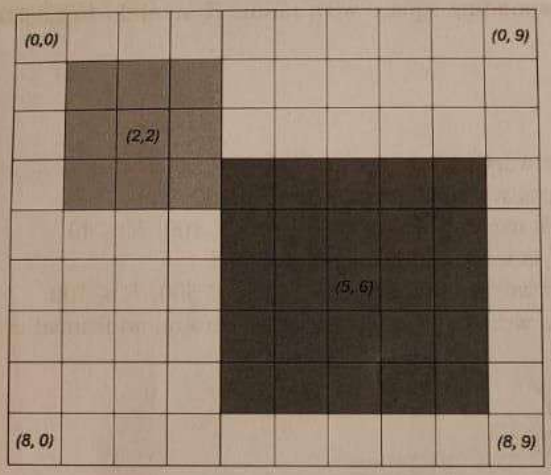
Problem 3: Gaussian blur

1 second / 1024 MiB / 100 points

You are given an $M \times N$ matrix with M rows and N columns whose entries are integers. The rows and columns of the matrix are indexed from 0. That means the top row has index 0, the bottom row has index $M-1$. The leftmost column has index 0 and the rightmost column has index $N-1$.

Denote by (i, j) the cell of the matrix which is located in the i-th row and j-th column. A square whose top left cell is $(i-R, j-R)$ and its bottom right cell is $(i+R, j+R)$ is called a **square with center** (i, j) **and radius** R.

Below you can see a matrix with $M = 9$, $N = 10$. In light gray we have a **square with center** $(2, 2)$ **and radius** 1. In dark gray we have a **square with center** $(5, 6)$ **and radius** 2.



Call the given matrix A. You are also given a positive integer C. For a **square with center** (i_0, j_0) **and radius** R, we define its **Gaussian blur** as the sum of all values $A[i][j] \cdot C^{|i-i_0|+|j-j_0|}$, where (i, j) go through all the cells of the mentioned square.

Below is a very rough piece of code that calculates that sum:

```
for(int i = -R; i <= R; ++i){
    for(int j = -R; j <= R; ++j){
        sum = sum + A[i_0 + i][j_0 + j] * pow(C, abs(i) + abs(j));
    }
}
```

Your task is to calculate the Gaussian blur of several such squares, all having the same radius R. As the resulting number can be very large, print its value modulo $1000000007 = 10^9 + 7$, which is a prime number. Be careful to use **long long** to avoid overflow where needed.

It is guaranteed that we will never consider a square which is not completely inside our matrix.

INPUT DATA:

In the first line, there are four integers M, N, R, and C, in that order.
Numbers M and N represent the number of rows and columns of our matrix and $3 \leqslant M, N \leqslant 1000$.
Number $1 \leqslant R \leqslant 200$ is the radius of the squares which will be considered (same R for all squares).
Number $1 \leqslant C \leqslant 1000$ is the number C from the problem statement (same C for all squares).

In each of the next M lines there are N integers $0 \leqslant x \leqslant 1000$, the values in the cells of our matrix.

In the next line, there is a single integer $1 \leqslant T \leqslant 100000$, the number of test cases.

In each of the next T lines, there are two integers $R \leqslant i \leqslant M - R - 1$ and $R \leqslant j \leqslant N - R - 1$, representing the center of a square with center (i, j) and radius R.

OUTPUT DATA:

Print T lines, one for each input test case. In each line, print one non-negative integer, the Gaussian blur of the corresponding square with radius R, modulo $1000000007 = 10^9 + 7$.

SCORING:

First test data cluster is worth 10 points and $M, N \leqslant 10$, $R \leqslant 3$, $C \leqslant 2$, $T \leqslant 10$.
Second test data cluster is worth 5 points and $T \leqslant 100$.
Third test data cluster is worth 5 points and $M, N \leqslant 100$, $R \leqslant 40$.
Fourth test data cluster is worth 20 points and $C = 1$.
Fifth test data cluster is worth 30 points and $M, N \leqslant 500$, $R \leqslant 100$.
Sixth test data cluster is worth 30 points and there are no additional constraints.

STEMgames

TECHNOLOGY ARENA

Problem 4: Find the square

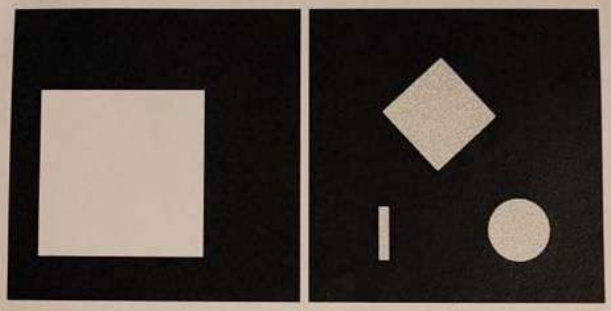
2 seconds / 1024 MiB / 100 points

You are given a grayscale image of resolution $M \times N$. The rows and columns of the image are indexed from 0. That means the top row has index 0, the bottom row has index $M-1$. The leftmost column has index 0 and the rightmost column has index $N-1$. Every pixel is represented by an integer between 0 and 255. The value 0 indicates pure black and 255 indicates pure white and the values in between are various shades of gray.

It is known that somewhere on the image, there is a single white square which you need to find. Square is a rectangle with all sides of equal length. Its size and color can vary, it might be tilted, viewed from the side or the image might be noisy. It is guaranteed that the correct answer is undoubtedly clear when the image is viewed by a human.

To find the square, print the pixel coordinates (row first, then column) of each of its four corner pixels in the image. You can print them in any order and an error of ± 1 pixel in row and column will be tolerated.

Below you can see a few images, the likes of which will appear in the test set. You can see that some cases look very easy, others much more tricky.



There will be 50 test cases in total, each worth 2 points. The goal is not to score 100 points, as there is no single correct algorithm for this type of problem. Doing simple things correctly is better than doing complicated things poorly. Given the time budget you have, try to cover as much cases as possible and score as high as possible.

INPUT DATA:

In the first line, there are two integers M, N in that order.
Numbers M and N represent the number of rows and columns of our image and $50 \leqslant M, N \leqslant 1500$.

In each of the next M lines there are N integers $0 \leqslant x \leqslant 255$, the values of each pixel of the input image.

OUTPUT DATA:

Print 4 lines, one for each corner of the white square. In each line, print two integers, the coordinates of one of the corners of the targeted square. Print row coordinate first, then column coordinate. Remember that rows and columns are indexed from 0. Your output will be considered correct if all coordinates differ from the official (human) output by at most 1 pixel on each axis.

SCORING:

There are 50 test cases, each worth 2 points. The cases will be of various sizes and difficulty.

TEST SAMPLES:

Input:

```
8 7
100 100 100 100 100 100 100
100 100 100 100 100 100 100
100 100 247 100 100 100 100
100 255 255 255 100 100 100
255 252 250 237 255 100 100
100 255 255 255 100 100 100
100 100 253 100 100 100 100
100 100 100 100 100 100 100
```

Output:

```
2 2
4 0
4 4
6 2
```

Explanation:

This is how approximately the zoomed-in image looks like. It is undoubtedly clear where the corners of the white square are located.