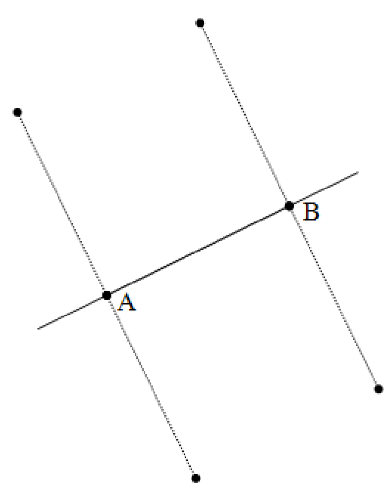
Rectangle

Spoiler

Solution of SQUARE

For start, let's solve a similar–looking easier task: find the area of the largest square.

All we have to do is pick two points A and B and find the coordinates of two other points in such a way that all four points form a square. There are two valid pairs of points — on one side of the line AB and on the other. For each of the pairs, we check whether both points actually exist. This check can be done in either $O(\log n)$ (if we store the points in a tree or use binary search) or $O(1)$ (if we hash each given point initially).



We have to check every pair of points, hence the total complexity is $O(n^2)$.

Solution of RECTANGLE

First, we notice that rectangles (and only rectangles!) have the following property: all four distances from the point of intersection of their diagonals to each vertex are equal.

For each pair of points (let's denote these points by A and B) we store a record R containing the following values:

$R.x = (A_x + B_x)/2$

$R.y = (A_y + B_y)/2$

$R.d = (A_x - B_x)^2 + (A_y - B_y)^2$

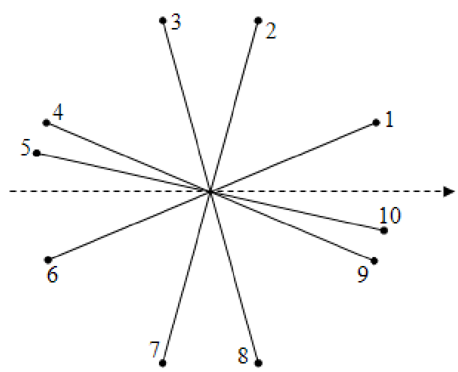
$R.\alpha$ — the angle between x axis and the segment AB

Note that $(R.x, R.y)$ is the midpoint of the segment AB. We will see later that it is not in fact necessary to divide $A_x + B_x$ and $A_y + B_y$ by two.

We sort these $\frac{n(n-1)}{2}$ records by (x, y), then by d and then by α value.

Suppose we have two records, R_1 and R_2, with $R_1.x = R_2.x$, $R_1.y = R_2.y$ and $R_1.d = R_2.d$. According to what was said before, this means that four points which produced these two records form a rectangle. Moreover, its area is equal to $\frac{1}{2}\sqrt{d}\sqrt{d}\sin\alpha = \frac{1}{2}d\sin\alpha$.

When the records are sorted, we can process each set of records having equal x, y and d values separately. Illustration of one such set would look like this:



We have to choose two of these segments in such a way that the rectangle would have the largest possible area. As d is fixed, we have to maximize $\sin\alpha$. In other words, we want the angle between two chosen segments to be as close as possible to 90 degrees.

This can be achieved using two cursors. Let's say the segments are numbered from 1 to M anti-clockwise, as can be seen in the drawing (in our case $M = 10$). At first, both cursors point to the first segment. Now, while the current angle is less than 90 degrees, we increase the second cursor. After some time, the angle becomes more or equal than 90 degrees. This is the time when we start moving the first cursor — we move it forward until the angle becomes less or equal than 90 degrees. Then we move the second cursor etc. We finish when the first cursor points to the segment whose y coordinate is less than zero (the segment number 6 in our case).

This last part can also be done by simply checking every pair or segments — it will work slower but still not exceed the time limit.

As we have to sort $O(n^2)$ records, the total time complexity is $O(n^2 \log n)$.

A final note: of course, it is not necessary to store the exact value of the angles if we want to avoid dealing with real numbers. We just need to know two things:

1. which of two angles is larger;
2. whether the difference between two angles is smaller or larger than 90 degrees.

Inefficient solution

The $O(n^3)$ solution is based on the solution of square problem. We consider every three of the given points and do the following:

1. Check whether they form a right triangle;
2. If they do, we calculate what the coordinates of the fourth vertex would be (in order for all four vertices to form a rectangle);
3. Check if there exists a given point at these coordinates.

Each of these steps takes $O(1)$ time.

Alternative solution

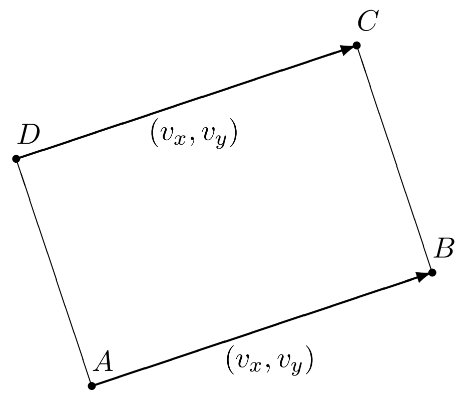
Below is a short description of a similar approach which yields better performance.

We try to solve the following task: find the number of right triangles whose vertices are from the given set of points. This task was used in *Croatian COCI 2007 Olympiad*. Using cursor approach, this task can be solved in $O(n^2 \log n)$ time. However, for every right triangle we find we have to check for the fourth vertex (as we did in *Inefficient solution*). Thus, the total complexity would be $O(n^2 \log n + t)$ where t is the number of right triangles. In practice, $t \leq n^2 \log n$ for $n \leq 1{,}500$ and therefore this solution is good enough to receive full score.

See: `http://www.hsin.hr/coci/archive/2007_2008/contest2_tasks.pdf`

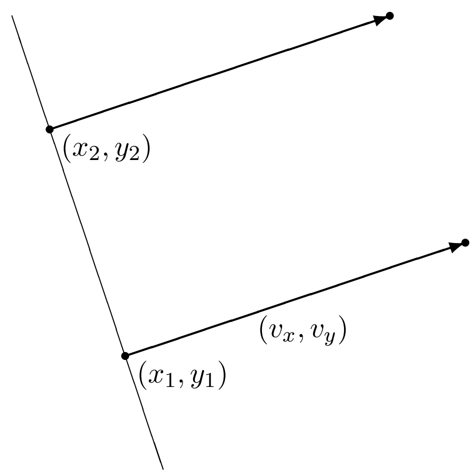
Yet another solution sorting vectors

Let us consider a rectangle on the plane and pick its vertex having minimum y coordinate (if two vertices have the same y coordinate, we pick the leftmost vertex). Let's denote this vertex by A. It is easy to see that vertex A is connected to a neighboring vertex B such that $B_x - A_x \geq 0$ and $B_y - A_y > 0$. In other words, for vector $\overrightarrow{AB} = (v_x, v_y)$ we have $v_x \geq 0$ and $v_y > 0$. Also note that the vector formed by other two vertices, $\overrightarrow{DC}$, is equal to $\overrightarrow{AB}$.



Now, let's check every pair of the given points and collect vectors which have this property ($v_x \geq 0$ and $v_y > 0$). We can sort these vectors so that we can process the sets of equal vectors easily.

Let's consider one such set of equal vectors and assume these vectors are (v_x, v_y). Any two vectors of the set form a parallelogram. Let's figure out when they form a rectangle. For this purpose, let's pick two vectors and assume the coordinates of their origins are (x_1, y_1) and (x_2, y_2). Sure enough, to form a rectangle, the vectors (v_x, v_y) and $(x_2 - x_1, y_2 - y_1)$ have to be perpendicular, which means that their *scalar product* has to be zero: $(x_2 - x_1)v_x + (y_2 - y_1)v_y = 0$. In other words, $x_1v_x + y_1v_y = x_2v_x + y_2v_y$. So we come to the following conclusion: two equal vectors form a rectangle if and only if their v-values are equal where the v-value of a vector (v_x, v_y) with the origin at (x_1, y_1) is defined as $x_1v_x + y_1v_y$.



Now, we can either check all rectangles and choose the largest one or we can make an optimization based on the following observation. First, we note that all equal vectors can be further divided into groups of vectors having the same v-value. Let's consider one such group. Of course, any two vectors of this group form a rectangle. In fact, origins of these vectors are located on a line which is perpendicular to the vectors. Therefore the largest rectangle is obtained from two vectors with their origins lying on extreme ends of this line. We can find these extreme vectors by simply taking a vector which has the minimum x coordinate of its origin and a vector which has the maximum x coordinate of its origin. Note that this line where the origin points lie cannot be vertical because at the very beginning we chose the vectors with $v_y > 0$.

In this solution, we must sort $O(n^2)$ vectors, therefore the total complexity is $O(n^2 \log n)$.

Test data overview

#	n	Triangles	Rectangles	Answer	Optimal	MaxCoord	Remarks	Points
0	8	22	3	10	1	3	Square	0
1	11	33	4	588	1	72		1
2	25	136	3	1223068	1	4806	Orthogonal	1
3	55	313	1	2071510376	1	786713	Square	1
4	102	192	23	786080	1	6013		2
5	199	48687	7349	1537360689312250	2	96803808	Square	2
6	309	74883	6458	139656460	1	9688	Orthogonal	2
7	353	1980	435	6967659886146	1	99811664		2
8	404	59376	6297	1615667124561765	1	93505480		3
9	448	9449	1	343200	1	6276		3
10	500	51042	10895	296329760	1	100000000		3
11	636	30092	271	1140700	1	9103		3
12	707	930325	154583	1865324372100	1	1465826		3
13	805	3613	112	1223032004876	1	2214859		4
14	902	3448	1	70301	1	8197	Orthogonal	4
15	1006	33132	528	19542192522058220	1	99547443		5
16	1103	22258	1549	4108806711522	4	1983220	Orthogonal	6
17	1191	22910	2761	199338750	1	10000		7
18	1325	125905	638	53801600	1	9408		8
19	1404	1014171	33590	3853519190576640	1	97381415		10
20	1488	2469480	339472	892144747132240	1	99994017		10
21	1500	3117	88	2053316907252	1	51198452		10
22	1500	5381228	1023027	5883783690585240	1	79730583		10

Legend:

n — the number of points.
Triangles — the number of right triangles with vertices in the given points.
Rectangles — the number of rectangles with vertices in the given points.
Answer — the area of the largest rectangle.
Optimal — the number of rectangles which have the maximum area.
MaxCoord — the maximum absolute value of coordinates of the points.
Remarks: Square — the largest rectangle is a square.
Remarks: Orthogonal — the sides of the largest rectangle are parallel to coordinate axes.