A. Rank List

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Another programming contest is over. You got hold of the contest's final results table. The table has the following data. For each team we are shown two numbers: the number of problems and the total penalty time. However, for no team we are shown its final place.

You know the rules of comparing the results of two given teams very well. Let's say that team a solved p_a problems with total penalty time t_a and team b solved p_b problems with total penalty time t_b. Team a gets a higher place than team b in the end, if it either solved more problems on the contest, or solved the same number of problems but in less total time. In other words, team a gets a higher place than team b in the final results' table if either $p_a > p_b$, or $p_a = p_b$ and $t_a < t_b$.

It is considered that the teams that solve the same number of problems with the same penalty time share all corresponding places. More formally, let's say there is a group of x teams that solved the same number of problems with the same penalty time. Let's also say that y teams performed better than the teams from this group. In this case all teams from the group share places $y + 1$, $y + 2$, ..., $y + x$. The teams that performed worse than the teams from this group, get their places in the results table starting from the $y + x + 1$-th place.

Your task is to count what number of teams from the given list shared the k-th place.

Input

The first line contains two integers n and k ($1 \le k \le n \le 50$). Then n lines contain the description of the teams: the i-th line contains two integers p_i and t_i ($1 \le p_i, t_i \le 50$) — the number of solved problems and the total penalty time of the i-th team, correspondingly. All numbers in the lines are separated by spaces.

Output

In the only line print the sought number of teams that got the k-th place in the final results' table.

Examples

input

```
7 2
4 10
4 10
4 10
3 20
2 1
2 1
1 10
```

output

```
3
```

input

```
5 4
3 1
3 1
5 3
3 1
3 1
```

output

```
4
```

Note

The final results' table for the first sample is:

- 1-3 places — 4 solved problems, the penalty time equals 10
- 4 place — 3 solved problems, the penalty time equals 20
- 5-6 places — 2 solved problems, the penalty time equals 1
- 7 place — 1 solved problem, the penalty time equals 10

The table shows that the second place is shared by the teams that solved 4 problems with penalty time 10. There are 3 such teams.

The final table for the second sample is:

- 1 place — 5 solved problems, the penalty time equals 3
- 2-5 places — 3 solved problems, the penalty time equals 1

The table shows that the fourth place is shared by the teams that solved 3 problems with penalty time 1. There are 4 such teams.

B. Polygons

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You've got another geometrical task. You are given two non-degenerate polygons A and B as vertex coordinates. Polygon A is strictly convex. Polygon B is an arbitrary polygon without any self-intersections and self-touches. The vertices of both polygons are given in the clockwise order. For each polygon no three consecutively following vertices are located on the same straight line.

Your task is to check whether polygon B is positioned strictly inside polygon A. It means that any point of polygon B should be strictly inside polygon A. "Strictly" means that the vertex of polygon B cannot lie on the side of the polygon A.

Input

The first line contains the only integer n ($3 \le n \le 10^5$) — the number of vertices of polygon A. Then n lines contain pairs of integers x_i, y_i ($|x_i|, |y_i| \le 10^9$) — coordinates of the i-th vertex of polygon A. The vertices are given in the clockwise order.

The next line contains a single integer m ($3 \le m \le 2 \cdot 10^4$) — the number of vertices of polygon B. Then following m lines contain pairs of integers x_j, y_j ($|x_j|, |y_j| \le 10^9$) — the coordinates of the j-th vertex of polygon B. The vertices are given in the clockwise order.

The coordinates of the polygon's vertices are separated by a single space. It is guaranteed that polygons A and B are non-degenerate, that polygon A is strictly convex, that polygon B has no self-intersections and self-touches and also for each polygon no three consecutively following vertices are located on the same straight line.

Output

Print on the only line the answer to the problem — if polygon B is strictly inside polygon A, print "YES", otherwise print "NO" (without the quotes).

Examples

input

```
6
-2 1
0 3
3 3
4 1
3 -2
2 -2
4
0 1
2 2
3 1
1 0
```

output

```
YES
```

input

```
5
1 2
4 2
3 -3
-2 -2
-2 1
4
0 1
1 2
4 1
2 -1
```

output

```
NO
```

input

```
5
-1 2
2 3
4 1
3 -2
0 -3
5
1 0
1 1
3 1
5 -1
```

2 -1
output
NO

C. Median

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

A *median* in an array with the length of n is an element which occupies position number $\lfloor \frac{n+1}{2} \rfloor$ after we sort the elements in the non-decreasing order (the array elements are numbered starting with 1). A median of an array $(2, 6, 1, 2, 3)$ is the number 2, and a median of array $(0, 96, 17, 23)$ — the number 17.

We define an expression $\lfloor \frac{a}{b} \rfloor$ as the integer part of dividing number a by number b.

One day Vasya showed Petya an array consisting of n integers and suggested finding the array's median. Petya didn't even look at the array and said that it equals x. Petya is a very honest boy, so he decided to add several numbers to the given array so that the median of the resulting array would be equal to x.

Petya can add any integers from 1 to 10^5 to the array, including the same numbers. Of course, he can add nothing to the array. If a number is added multiple times, then we should consider it the number of times it occurs. It is not allowed to delete of change initial numbers of the array.

While Petya is busy distracting Vasya, your task is to find the minimum number of elements he will need.

Input

The first input line contains two space-separated integers n and x ($1 \le n \le 500$, $1 \le x \le 10^5$) — the initial array's length and the required median's value. The second line contains n space-separated numbers — the initial array. The elements of the array are integers from 1 to 10^5. The array elements are not necessarily different.

Output

Print the only integer — the minimum number of elements Petya needs to add to the array so that its median equals x.

Examples

input

```
3 10
10 20 30
```

output

```
1
```

input

```
3 4
1 2 3
```

output

```
4
```

Note

In the first sample we can add number 9 to array $(10, 20, 30)$. The resulting array $(9, 10, 20, 30)$ will have a median in position $\lfloor \frac{4+1}{2} \rfloor = 2$, that is, 10.

In the second sample you should add numbers 4, 5, 5, 5. The resulting array has median equal to 4.

D. Shoe Store

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

The warehouse in your shop has n shoe pairs. Each pair is characterized by two integers: its price c_i and its size s_i. We know that on this very day all numbers s_i are different, that is, there is no more than one pair of each size.

The shop has m customers who came at the same time. The customer number i has d_i money and the size of his feet equals l_i. The customer number i can buy the pair number j, if $c_j \le d_i$, and also if $l_i = s_j$ or $l_i = s_j - 1$; that is, it is necessary that he has enough money to pay for the shoes. It is also necessary that the size of his feet equals to or is less by 1 than the size of the shoes he chooses.

Your task is to sell some customers pairs of shoes (a pair per person) so as to maximize the sum of the sold pairs c_j that is, the profit. It is guaranteed that each customer buys no more than one pair and each pair will be bought by no more than one customer.

Input

The first input line contains the only integer n ($1 \le n \le 10^5$) — the number of shoe pairs in the warehouse. Then n lines contain the descriptions of pairs of shoes as two integers c_i and s_i ($1 \le c_i, s_i \le 10^9$), the numbers are separated by a space. It is guaranteed that all numbers s_i are different.

The next line contains an integer m ($1 \le m \le 10^5$) — the number of customers in the shop. Next m lines contain the customers' descriptions as two integers d_i and l_i ($1 \le d_i, l_i \le 10^9$), the numbers are separated by a space.

Output

In the first line print the only integer — the maximum profit you can get from selling shoes. In the second line print an integer k — the number of shoe pairs you will sell. In the following k lines print the descriptions of the sold pairs — two space-separated integers where the first number is the customer's number and the second number is the number of the shoes the customer will buy.

You can print pairs of numbers "the customer's number and the shoes' number" in any order, the customers and the pairs of shoes are numbered starting from 1 in the order in which they are given in the input. If there are several optimal answers, you are allowed to print any of them.

Please do not use the %lld specificator to read or write 64-bit numbers in C++. It is preferred to use the `cin`, `cout` streams or the %I64d specificator instead.

Examples

input

```
3
10 1
30 2
20 3
2
20 1
20 2
```

output

```
30
2
2 3
1 1
```

input

```
3
10 4
20 5
30 6
2
70 4
50 5
```

output

```
50
2
2 3
1 2
```

E. Tetrahedron

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given a tetrahedron. Let's mark its vertices with letters A, B, C and D correspondingly.

An ant is standing in the vertex D of the tetrahedron. The ant is quite active and he wouldn't stay idle. At each moment of time he makes a step from one vertex to another one along some edge of the tetrahedron. The ant just can't stand on one place.

You do not have to do much to solve the problem: your task is to count the number of ways in which the ant can go from the initial vertex D to itself in exactly n steps. In other words, you are asked to find out the number of different cyclic paths with the length of n from vertex D to itself. As the number can be quite large, you should print it modulo 1000000007 ($10^9 + 7$).

Input

The first line contains the only integer n ($1 \leq n \leq 10^7$) — the required length of the cyclic path.

Output

Print the only integer — the required number of ways modulo 1000000007 ($10^9 + 7$).

Examples

input

```
2
```

output

```
3
```

input

```
4
```

output

```
21
```

Note

The required paths in the first sample are:

- D - A - D
- D - B - D
- D - C - D