Codeforces Round #329 (Div. 2)

Problem B: Anton and Lines

Prof. Edson Alves

Faculdade UnB Gama

Codeforces Round #329 (Div. 2) – Problem B: Anton and Lines

Problema

The teacher gave Anton a large geometry homework, but he didn't do it (as usual) as he participated in a regular round on Codeforces. In the task he was given a set of n lines defined by the equations $y = k_i x + b_i$. It was necessary to determine whether there is at least one point of intersection of two of these lines, that lays strictly inside the strip between $x_1 < x_2$. In other words, is it true that there are $1 \leq i < j \leq n$ and x', y', such that:

- $y' = k_i x' + b_i$, that is, point (x', y') belongs to the line number i;
- $y' = k_j x' + b_j$, that is, point (x', y') belongs to the line number j;
- $x_1 < x' < x_2$, that is, point (x', y') lies inside the strip bounded by $x_1 < x_2$.

You can't leave Anton in trouble, can you? Write a program that solves the given task.

Entrada e saída

Input

The first line of the input contains an integer n $(2 \leq n \leq 100000)$ – the number of lines in the task given to Anton. The second line contains integers x_1 and x_2 $(-1000000 \leq x_1 < x_2 \leq 1000000)$ defining the strip inside which you need to find a point of intersection of at least two lines.

The following n lines contain integers k_i, b_i $(-1000000 \leq k_i, b_i \leq 1000000)$ – the descriptions of the lines. It is guaranteed that all lines are pairwise distinct, that is, for any two $i \neq j$ it is true that either $k_i \neq k_j$, or $b_i \neq b_j$.

Output

Print "Yes"(without quotes), if there is at least one intersection of two distinct lines, located strictly inside the strip. Otherwise print "No"(without quotes).

Exemplo de entradas e saídas

Sample Input

```
4
1 2
1 2
1 0
0 1
0 2

2
1 3
1 0
-1 3
```

Sample Output

```
YES

NO
```

Solução com complexidade $O(N \log N)$

- A busca completa, que verifica todos os pares de segmentos de reta, tem complexidade $O(N^2)$, o que leva ao TLE, pois $N \leq 10^5$

Solução com complexidade $O(N \log N)$

- A busca completa, que verifica todos os pares de segmentos de reta, tem complexidade $O(N^2)$, o que leva ao TLE, pois $N \leq 10^5$
- Contudo, é possível determinar as possíveis interseções no intervalo (x_1, x_2) com um algoritmo *sweep line*

Solução com complexidade $O(N \log N)$

- A busca completa, que verifica todos os pares de segmentos de reta, tem complexidade $O(N^2)$, o que leva ao TLE, pois $N \leq 10^5$
- Contudo, é possível determinar as possíveis interseções no intervalo (x_1, x_2) com um algoritmo *sweep line*
- Os segmentos devem ser ordenados, em ordem crescente, pelo valor da coordenada y do segmento no ponto x_1

Solução com complexidade $O(N \log N)$

- A busca completa, que verifica todos os pares de segmentos de reta, tem complexidade $O(N^2)$, o que leva ao TLE, pois $N \leq 10^5$
- Contudo, é possível determinar as possíveis interseções no intervalo (x_1, x_2) com um algoritmo *sweep line*
- Os segmentos devem ser ordenados, em ordem crescente, pelo valor da coordenada y do segmento no ponto x_1
- Em caso de empate, deve-se ordenar pela coordena y no ponto x_2

Solução com complexidade $O(N \log N)$

- A busca completa, que verifica todos os pares de segmentos de reta, tem complexidade $O(N^2)$, o que leva ao TLE, pois $N \leq 10^5$
- Contudo, é possível determinar as possíveis interseções no intervalo (x_1, x_2) com um algoritmo *sweep line*
- Os segmentos devem ser ordenados, em ordem crescente, pelo valor da coordenada y do segmento no ponto x_1
- Em caso de empate, deve-se ordenar pela coordena y no ponto x_2
- Cada segmento deve ser processado uma única vez, nesta ordem

Solução com complexidade $O(N \log N)$

- A busca completa, que verifica todos os pares de segmentos de reta, tem complexidade $O(N^2)$, o que leva ao TLE, pois $N \leq 10^5$
- Contudo, é possível determinar as possíveis interseções no intervalo (x_1, x_2) com um algoritmo *sweep line*
- Os segmentos devem ser ordenados, em ordem crescente, pelo valor da coordenada y do segmento no ponto x_1
- Em caso de empate, deve-se ordenar pela coordena y no ponto x_2
- Cada segmento deve ser processado uma única vez, nesta ordem
- Deve-se manter o registro da maior coordenada y em x_2 já encontrada (inicialmente, este valor deve ser igual a $-\infty$)

Solução com complexidade $O(N \log N)$

- A busca completa, que verifica todos os pares de segmentos de reta, tem complexidade $O(N^2)$, o que leva ao TLE, pois $N \leq 10^5$
- Contudo, é possível determinar as possíveis interseções no intervalo (x_1, x_2) com um algoritmo *sweep line*
- Os segmentos devem ser ordenados, em ordem crescente, pelo valor da coordenada y do segmento no ponto x_1
- Em caso de empate, deve-se ordenar pela coordena y no ponto x_2
- Cada segmento deve ser processado uma única vez, nesta ordem
- Deve-se manter o registro da maior coordenada y em x_2 já encontrada (inicialmente, este valor deve ser igual a $-\infty$)
- Se a coordenada y em x_2 do segmento a ser processado for menor do que a maior já encontrada, significa que houve uma interseção com algum dos segmentos já processados

Solução com complexidade $O(N \log N)$

```cpp
#include <bits/stdc++.h>

using namespace std;
using ll = long long;

const ll oo { 1000000000000000000LL };

struct Line
{
    ll k, b;

    ll eval(ll x) const { return k*x + b; }
};

bool solve(ll x1, ll x2, vector<Line>& lines)
{
    sort(lines.begin(), lines.end(), [&](const Line& r, const Line& s) {
        if (r.eval(x1) != s.eval(x1))
            return r.eval(x1) < s.eval(x1);
```

Solução com complexidade $O(N \log N)$

```
            return r.eval(x2) < s.eval(x2);
    });

    auto max_y = -oo;

    for (const auto& r : lines)
    {
        auto y = r.eval(x2);

        if (y < max_y)
            return true;

        max_y = max(y, max_y);
    }

    return false;
}
```