Problem 3 - ADA Rectangle (Programming) (15 points)

Problem Description

Given N points $(p_1, p_2 \cdots, p_n)$ on a 2-dimensional (2D) plane, we define two points p_i and p_j as a good pair if there exists a rectangle satisfying the following conditions:

1. The sides of the rectangle are parallel to the X-axis and Y-axis.
2. The rectangle contains these two points p_i, p_j only; no other points fall within this rectangle.

How many good pairs of points are there in total?

Input

The first line of the input contains one integer N, denoting the number of points.

The i^{th} of the following N lines contains two integers x_i, y_i, indicating the coordinates of the i^{th} point.

Test Group 0 (0 %)

- Sample Input.

Test Group 1 (20 %)

- $1 \leq N \leq 5 \times 10^2$
- $1 \leq x_i \leq N$
- $1 \leq y_i \leq N$
- All x_i are distinct.
- All y_i are distinct.

Test Group 2 (35 %)

- $1 \leq N \leq 3 \times 10^3$
- $1 \leq x_i \leq N$
- $1 \leq y_i \leq N$
- All x_i are distinct.
- All y_i are distinct.

Test Group 3 (45 %)

- $1 \leq N \leq 2 \times 10^5$
- $1 \leq x_i \leq N$
- $1 \leq y_i \leq N$
- All x_i are distinct.
- All y_i are distinct.

Output

Print the number of good pairs in a single line.

Sample Input 1

```
5
1 5
2 2
5 4
4 1
3 3
```

Sample Output 1

```
8
```

Sample Input 2

```
5
1 1
2 2
3 3
4 4
5 5
```

Sample Output 2

```
4
```

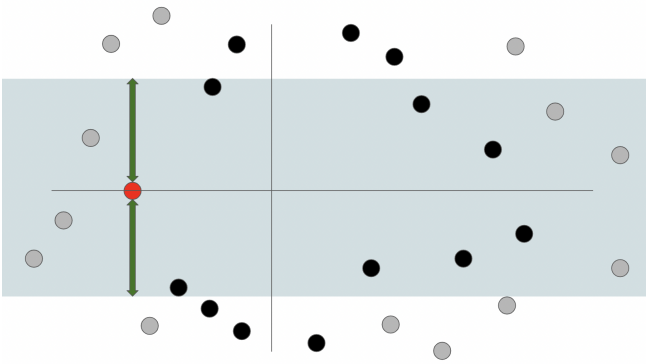
Sample Input 3

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

Sample Output 3

```
3
```

Hint



1. The good pairs in Sample Input 1 are (p_1, p_2), (p_1, p_3), (p_1, p_5), (p_2, p_4), (p_2, p_5), (p_3, p_4), (p_3, p_5), (p_4, p_5).
2. You are also encouraged to use the above picture to find inspiration. When conquering, you can use stacks to maintain the black points, and use binary search to identify how many black points are in the gray scope.
3. You can maintain a stack on each of the left and right sides to keep the "needed points", which are the points blocking the current point or forming a good pair with the current point you are focusing on.
4. Instead of `std::stack`, you can use `std::vector` to maintain stack and perform binary search algorithm on `std::vector` directly using `std::lower_bound`.
5. GL & HF (Good Luck and Have Fun).