```
Merge-sort(S):
 if $|S| \le 1$, then return S
 divide S into two subsets $S_1$, $S_2$, of equal size
    A = merge-Sort($S_1$)
    B = merge-Sort($S_2$)
    C = merge-Sort(A+B);  ☆
 return C;
```

function(A, B)

↳ (A, B)

Binary Search Tree (BST)

Quick Sort:

2 6 5 3 8 7 1 0

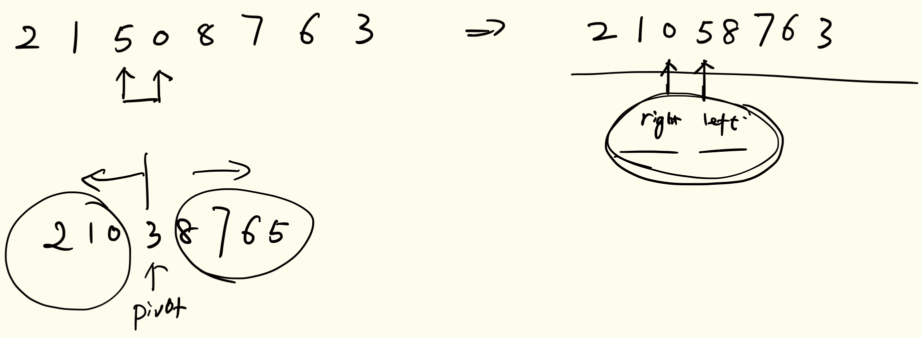
pivot

2 6 5 0 8 7 1 3

⇒ 2 item. 1st: from left which is the first item that is larger than pivot.

2nd: item from right which the first item is smaller than pivot

2 6 5 0 8 7 1 3 ⇒ 2 1 5 0 8 7 6 3



Worest $O(n^2)$

Averge $O(n\log n)$

Guess & Verify

$$T(n) = 2T\left(\frac{n}{2}\right) + \frac{n}{\log n} = O(n)$$

$$T(n) \le c\left(n - \frac{n}{\log n}\right)$$

$$\Rightarrow T\left(\frac{n}{2}\right) \le c\left(\frac{n}{2} - \frac{n/2}{\log n/2}\right)$$

$$\Rightarrow 2c\left(\frac{n}{2} - \frac{n/2}{\log n/2}\right) + \frac{n}{\log n} \le cn - c\,\frac{n}{\log n}$$

$$cn - \frac{cn}{\log n/2} + \frac{n}{\log n} \le cn - \frac{cn}{\log n} \qquad ⓒ$$

$$\Rightarrow cn - \frac{cn}{\log n - 1} + \frac{n}{\log n} \le cn - \frac{cn}{\log n} \qquad ☆$$

$$\Rightarrow \star \quad \frac{1}{\log n} \le \frac{c}{\log n - 1} - \frac{c}{\log n} \qquad c$$

$$\frac{c}{\log n - 1} \geq \frac{c+1}{\log n}$$

$$\frac{c}{c+1} \geq \frac{\log n - 1}{\log n} \in (0, 1)$$

$$1 \geq$$

$$\textcircled{c}$$

Concept

$$T(n) = a\,T\left(\frac{n}{b}\right) + f(n)$$

$$\rightarrow \text{Guess } O(n) \qquad \le C\,(n - f(n))$$

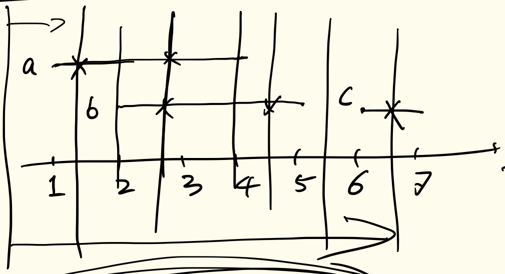
$$O(n^2) \qquad \le C\,(n^2 - f(n))$$

$$\begin{cases} T(n) \le c\,n. = O(n) \\ a\,T\left(\frac{n}{b}\right) \le dn = O(n) \end{cases}$$

Sweep Line.

Leetcode: ① Number of Airplane in the sky

② Building Outline.



a = [1 , 4]

b = [2 , 5]

c = [6 , 7]   2

What's max number of Airplane at one time

(1, 1)   (4, 0)

[1, T]	[4, F]
[2, T]	[5 F]
[6, T]	[7, F]

```
for loop for all intervals:
    List.add [ (i.start, 1) ]
    List.add [ (i.end, 0) ]
Sort (List)
for point in List:
        if p.flag == 1,   count++
        else:
            count --
        ans = Math.max(ans, count)
        return ans.
```

Count	1	1	2	2	3	2	2	1
ans	1	1	2	2	3	3	3	3

$O(n \log n)$