Big O Notation

$a = \underbrace{[\cdots]}_{N}$ → a list of N integers

f_4 {
- $f_1(a) \Rightarrow$ lt $a[0] \Rightarrow O(1)$
- $f_2(a) \Rightarrow$ sum $[a] \Rightarrow O(N)$
- $f_3(a) \Rightarrow$ pair $[a] \Rightarrow O(N^2)$

$[1,2,3] \Rightarrow [1,1], [1,2], [1,3]$
$[2,2], [2,1], [2,3]$
$[3,3], [3,1], [1,3]$

The speed of the algorithm is very depend on the size of input

$$f_4(a) = O(N^2 + N + 1) = O(N^2)$$

$O(1)$

$O(\log(N))$

$O(N)$

$O(N \log N)$

$O(N^2)$, $O(N^3)$, $O(N^4)$

$O(2^N)$

$O(N!)$

worse time complexity ↓

☆ In coding interview, the time complexity is always considered the worst case.

Example:

1. $O(25) = O(1)$
2. $O(2N) = O(N)$
3. $O(N^2 + 2N) = O(N^2)$
4. $O(N^3 + \log(N) + 3) = O(N^3)$
5. $O(N+M) = O(N+M)$ ☆ N and M are different, do not drop