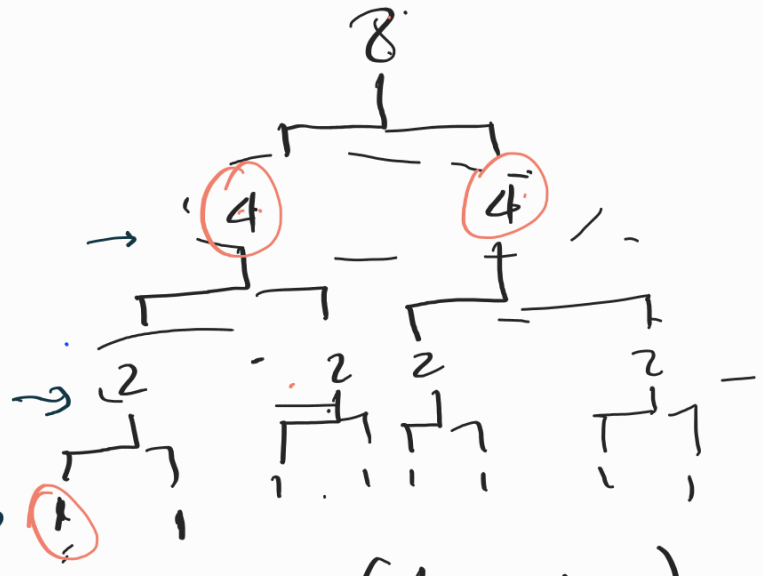
Merge sort:

$S = [7, 5, 2, 3, 0, 4, 1, 6]$ input w/ N_array

$S_1 = [7, 5, 2, 3]$



$(\log_2 n)$

$S_{11} = [7, 5]$

7
5
→ [5, 7]

$S_{12} = [2, 3]$

2
3
→ [2, 3]

→ [2, 3, 5, 7]

S_2 — [0, 1, 4, 6]

→ [0, 1, 2, 3, 4, 5, 6, 7]

Combine

Runtime: $2k$

l_1 & l_2 $\quad O(k)$.

$l_1 = [a_1, a_2, \ldots, a_k]$ ↑

$l_2 = [b_1, b_2, \ldots, b_k]$ ↑

$\rightarrow [a_1, a_2, b_1, \ldots]$.

Complexity

Combine: $O(n)$

$R * O(s)$

↳ $O(s) = O(n)$

$R = \log(n)$

complexity merge-sort: $\boxed{O(n \log(n))}$

n	2^n	$\log_2(2^n)$
5	32	5
4	16	4
3	8	3
2	4	2
1	2	1
0	1	0

$2^n = \underbrace{2 * \ldots * 2}_{n}$

$\log_2(k) \doteq$ # times ... we divide n by 2 until we get 1 as result.

$k/2/2/2\ldots = 1$

n times

Memory complexity :- extra memory.

w/o deep copy (pointers): $O(1)+n$

w/ 1. deep copy of original array $O(n)+n$

w/. 1 deep copy per recursive call: $O(n \log(n))$

$+ n$

Quick Sort:

[5, 9, 7, 2, 3, 10, 0, 4, 1, 8, 6, 11]

rand(0, 11)

pivot = 5

S_1 = [2, 3, 0, 4, 1]

S_{11} = [0, 1]

[1]

[0, 1]

[0, 1, 2, 3, 4]

S_{12} = [3, 4]

[3, 4]

[4]

$S_2 = [9, 7, 10, 8, 6, 11]$

$S_{21} = [7, 8, 6]$

[6] [7] → [6, 7, 8]

$S_{22} = [10, 11]$

[11] → [10, 11]

[6, 7, 8] + [9] + [10, 11] → [6, 7, 8, 9, 10, 11]

[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11]

Combine:

$S_{left} + pivot + S_{right}$ "+" = concat.

Complexity

R: log(n)

O(S): n

> $O(n \log n)$

12

[11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0].

[10, ..., 0].

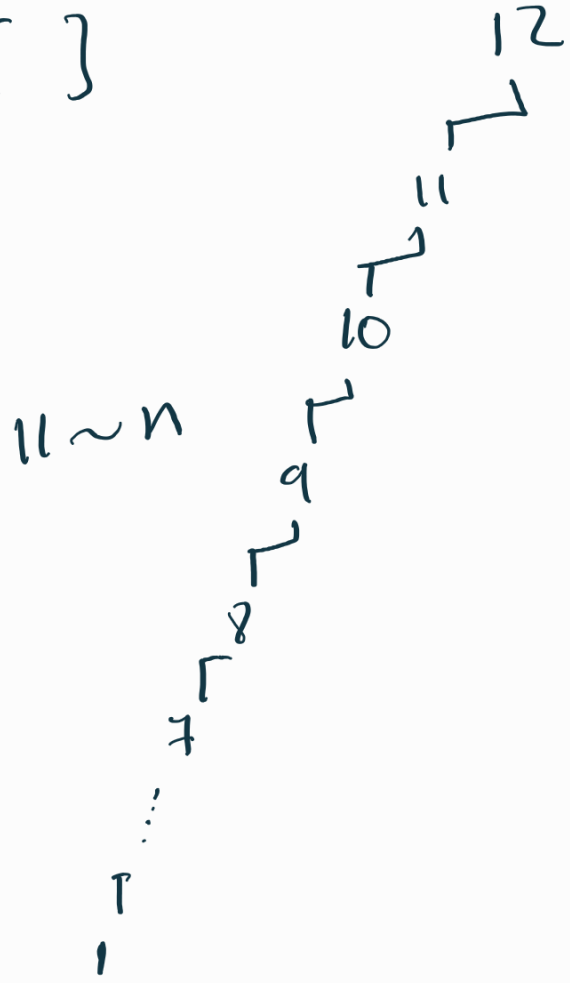
[9, ..., 0]

[8, ... 0]

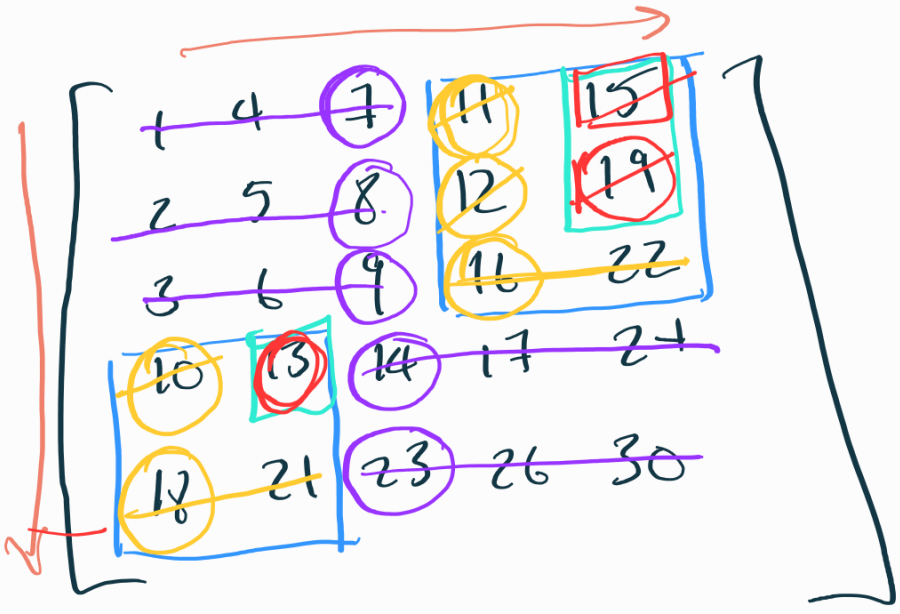
[7]

[]

[ ]



$O(n^2)$



target: 13

+1 python: