→ Sorting Algorithms

3 classes
- Today
- Thursday
- Sunday

Pre-requisite →
- for / while
- basic recursion
- probability

Basic Sorting → Bubble sort / selection / insertion

merge sort / quick sort

Counting sort / Radix / Bucket

What is sorting ?? Given a set of objects arranging them in some particular order is called sorting

$\boxed{P}$

Polynomial time solvable algorithms

$O(n^a)$

$\boxed{NP}$

Non-polynomial time solvable but can be verified in polynomial time

$O(2^n)$

$[a_1 \ a_2 \ a_3 \ a_4 \ \cdots a_n] \longleftarrow$ You want to sort them

Brute force $\rightarrow$ get all possible

arrangement $\rightarrow$ permutation

permutation

$n! \rightarrow 10^{n}$

output $\rightarrow$ verify $\rightarrow O(n)$

bool var $\rightarrow$ false

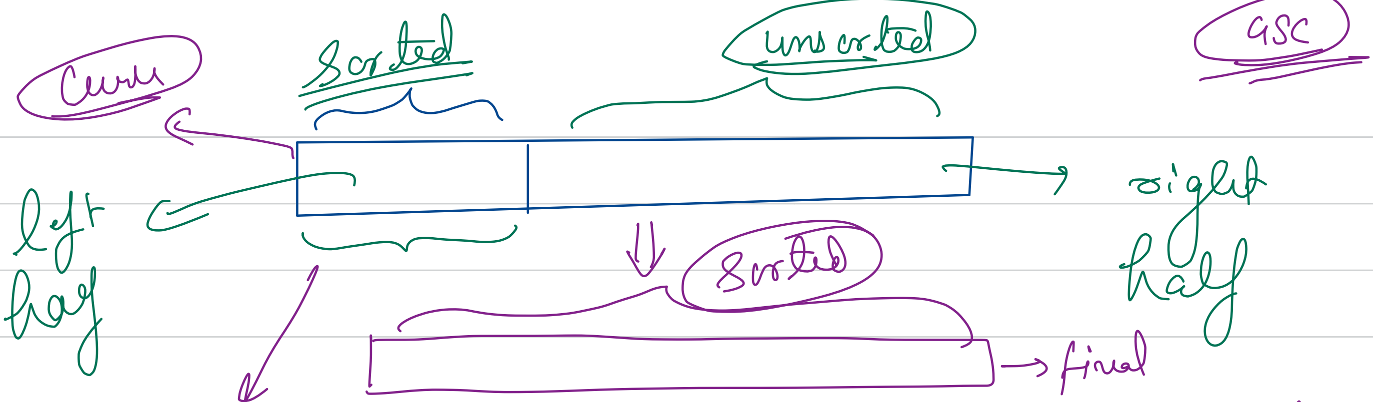
true

$\rightarrow$ [even] [odd]

$O(n)$

1

Intuition



1. We are given a situation that left part is sorted & right is unsorted.

2. All the elements of left part are smaller than all the elements of right part

I can add 'x', which should be just 1 larger than the largest element of sorted region.

viz the smallest element from unsorted.

Q. How can we find smallest element from unsorted region.

→ $\boxed{\text{Linear Search}}$

Selection Sort

sorted

unsorted

0, 2, 6, 7, 2, 1, 5, 3

min = 0

min_idx = 6

swap → i, min_idx

0, 1, 6, 7, 2, 2, 5, 3

0, 1, 2, 7, 6, 2, 5, 3

0, 1, 2, 2, 6, 7, 5, 3

0, 1, 2, 2, 3, 7, 5, 6

empty

How many comparisons it did?

$$N + (N-1) + (N-2) \cdots\cdots (1) \rightarrow \frac{n(n+1)}{2} \rightarrow O(n^2)$$

How many swaps it did?

$$1 + 1 + 1 \cdots\cdots \rightarrow \underline{\underline{O(n)}}$$

→ RAM

HDD

Flipkart Amazon product

seller

Iphone's sort $\rightarrow$ Price

X'_{40k}, X''_{40k}, X_{50k}, X_{55k}, X_{60k} - - - - - -

Stable Sorting

property

stability

$2', 1', 2'', 1'', 3, 4, 5', 5''$

after sorting the array, the relative ordering of the elements should not change.

$1', 1'', 2', 2'', 3, 4, 5', 5''$

Selection sort is not stable → $4', 2, 3, 4'', 1$

In place

NO ✗✗

Sorted $1, 2, 3, 4'', 4'$

Can we make it stable?: → instead of swapping

we can insert

Swap

$O(n^2)$ swaps

$4', 2, 3, 4'', 1$ → $x = 1$

$1, 4', 2, 3, 4''$

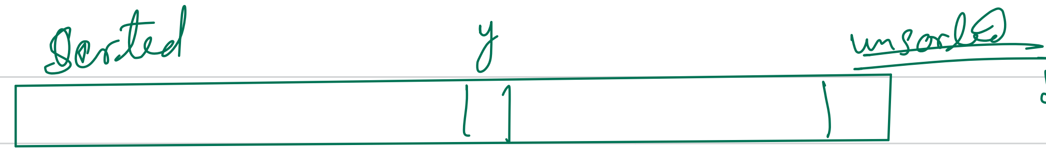
Space Complexity $\rightarrow$ $O(1)$

Time Complexity $\rightarrow$ $O(n^2)$ $\rightarrow$ worst

$\Omega(n^2)$ $\rightarrow$ best

$\Theta(n^2)$ $\rightarrow$ avg

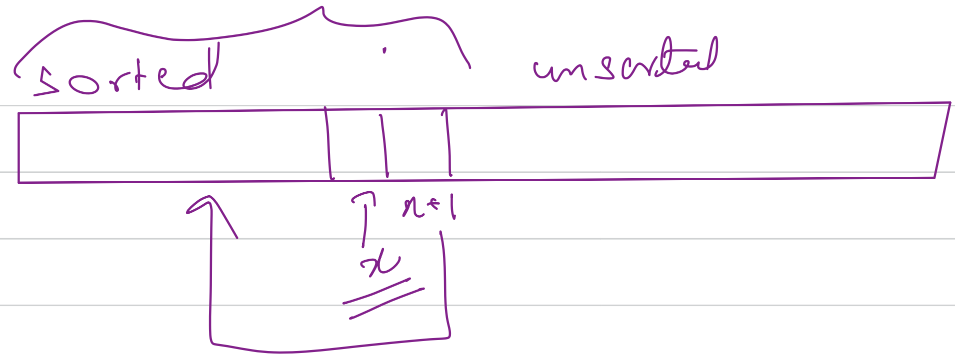
→



not at its correct pos

① We are given a situation that left part is sorted & right is unsorted.

$$\underbrace{17 \quad 18 \quad 25 \quad 33}_{\text{sorted}} \quad \underbrace{64}_{\text{unsorted}}$$



- 10 cards of hearts

Left

Right

aa, 5, 6, 9, 10

6, 7, 4, 3, 2, 8

sorted ← [13, ~~17~~ 13, ~~25~~ 17, ~~37~~ 25, ~~2~~ 37] → unsorted

Key = ~~2~~ 9

Comparison → $\underline{\underline{O(n^2)}}$

Swaps → $\underline{\underline{O(n^2)}}$

→ worst case is when array is $\underline{\underline{DESC}}$

Best case

→ comparison → $\Omega(n)$

→ swaps → $\Omega(n)$

5 4 3 2 1 → 1

4 5 3 2 1 → 2

3 4 5 2 1 → 3

⋮

n-1

I try to find correct pos of 2 before 2

https://www.cs.usfca.edu/~galles/visualization/ComparisonSort.html

Time complex → $\Omega(n)$

$O(n^2)$

$\Theta(n^2)$

Space complex → $O(1)$

Inplace

Insertion sort is Stable Sorting

The part of finding the correct pos can be optimised by Binary Search. → comparison → $O(\log n)$

Swapping for x is still $O(n)$

$$O\left(n\left(\log n + n\right)\right) \longrightarrow$$

$$\Downarrow$$

$$\underline{\underline{O(n^2)}}$$

Bubble Sort

Bubbling Up

Comparison

compare adjacent elements & swap if they are in wrong order.

14, 33, 27, 35, 10

14, 27, 33, 35, 10

14, 27, 33, 10, 35

14, 27, 10, 33, 35

14, 10, 27, 33, 35

10, 14, 27, 33, 35

★ at any i^{th} iteration, the i^{th} largest element will be at the i^{th} last position on array.

Stable → Yes

Inplace → Yes

Comparisons → $n-1 + n-2 + n-3 \ldots\ldots 1 \rightarrow O(n^2)$

Swaps → $O(n^2)$

already sorted Best Case → # Comparison $\Omega(n)$ Swaps → $\Omega(1)$

$2', 1', 2'', 1''$

$1', 2', 2'', 1''$

$1', 2', 1'', 2''$

5 4 3 2 1

Time $\rightarrow$ $O(n^2)$ $\Theta(n^2)$ $\Omega(n)$

Space $\rightarrow$ $O(1)$