Sorting

↳ What is sorting ?? → Sorting is a category of algorithms, which are used to re-arrange a bunch of objects in a specific manner.

↳ What is the brute force way to sort the given elements ??

$\boxed{A, B, C}$

rearrangement → permutation

↓ we go and calc all the possible permutations of the given data & the permutation which fulfills our requirement will be our ans.

A B C
A C B
B A C
B C A
C A B
C B A

→ $n!$ permutations in the case when all obj are distinct.

Sorting

- Comparison Based
- Counting Based.

What properties we look for in sorting algo.

(1) Time Complexity

(2) Space Complexity

(3) Inplace → when an algorithm sorts the same given list, without copying the data somewhere or mellow externally sorting it.

(4) Stablity → If the relative order of elements remains same even after sorting, then it is a stable sort.

$\underline{\underline{\text{ip}}} \rightarrow [2', 1', 3, 2'', 1'', 1''', 4, 2''']$

$\text{o/p} \rightarrow [1', 1'', 1''', 2', 2'', 2''', 3, 4]$

→ blue iphone11, black iph12, blue iph12, white iphXR

→ [white iphXR, blue iph11, black iph12, blue iph12]

Stable sort

(5) No of comparison

(6) No of Swaps

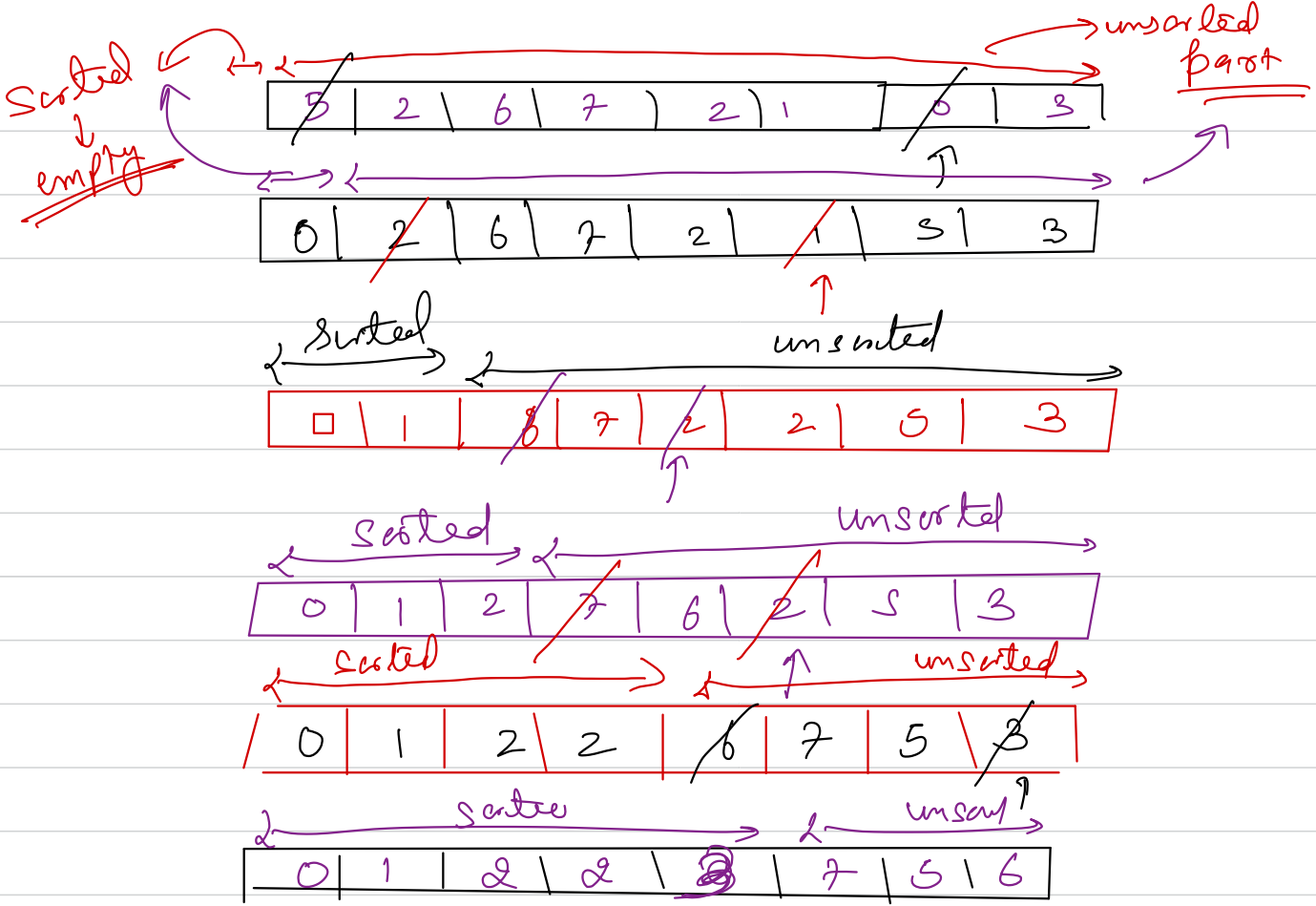
★ Selection Sort

Scenario → Sorting based on value of int.

1) You've 2 parts in a list → first part is sorted and second is unsorted.

2) The largest element of the sorted part is smaller than the smallest element of unsorted part.

→ find the min element from the unsorted region, and swap it with the first element of the unsorted region. This will expand the sorted region.


Sorted
empty
unsorted part
5 | 2 | 6 | 7 | 2 | 1 | 0 | 3
0 | 2 | 6 | 7 | 2 | 1 | 5 | 3
sorted
unsorted
0 | 1 | 6 | 7 | 2 | 2 | 5 | 3
sorted
unsorted
0 | 1 | 2 | 7 | 6 | 2 | 5 | 3
sorted
unsorted
0 | 1 | 2 | 2 | 6 | 7 | 5 | 3
sorted
unsorted
0 | 1 | 2 | 2 | 3 | 7 | 5 | 6

TC → $n + n-1 + n-2 + n-3 - - - - - - 3 + 2 + 1$

→ $\frac{n(n+1)}{2}$ → $\frac{n^2}{2} + \frac{n}{2}$

→ $O(n^2)$ worst Case

$\Omega(n^2)$ Best Case

$\Theta(n^2)$ Avg

applications

SC → $O(1)$ const

Inplace → Yes

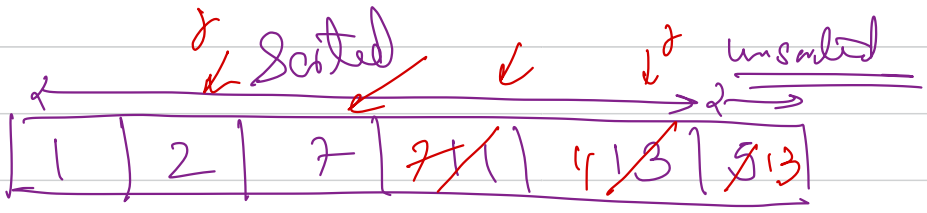
Stability → No

# of comparison → n^2

# of swaps → n swaps

★ Insertion Sort.

1) You've 2 parts in a list → first part is sorted and second is unsorted.



sorted unsorted

25, 17, 31, 13, 2

17, 25, 31, 13, 2

17, 25, 31, 13, 2

13, 17, 25, 31, 2

2, 13, 17, 25, 31

$(n-1) \times n$

→ pick the first element of the unsorted region, & insert it at it's correct position.

TC → Worst → reverse sorted list $O(n^2)$

Avg → $\Theta(n^2)$

Best → almost sorted array → $\Omega(n)$

SC → $O(1)$

Inplace → Yes

Stability → Yes

# of comparison → n^2

# of swaps → n^2

* Bubble Sort → Bubbling up that bigger value are first settled

14	33	27	35	10

after every iteration, biggest no. of unsorted region is placed at it's correct pos.

unsorted

n	14	33	27	35	10	→ 1st iterate
		27	33	10	35	
n-1	14	27	33	10	35	sorted
			10	33		2nd iteration
n-2	14	27	10	33	35	3rd iterate
		10	27			
n-3	14	10	27	33	35	4 more
	10	14				
	10	14	27	33	35	

TC → $n + n-1 + n-2 \text{ - - - - - - } 3+2+1$

→ $O(n^2)$ worst case

→ 1 2 3 4 5 → flag = False

Best Case → already sorted → $\Omega(n)$

Avg → $\Theta(n^2)$

SC → $O(1)$

Stability → Yes

Inplace → Yes

# of comparison → n^2

# of swaps → n^2

worst cases

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