1) Why Sorting ?
2) Problems.
3) Comparator.

SORTING ?

Arranging ~~integers~~ data/objects in ~~asc/desc~~ specific order based on some parameter.

Ex

1) 3, 8, 9, 14, 19 ⇒ Sorted in ascending order.

2) 19, 14, 9, 8, 3 ⇒ Descending order.

3) 1, 3, 9, 6, 12 ⇒ Sorted in asc. order based on the no. of factors.

	↓	↓	↓	↓	↓
No. of factors	1	2	3	4	6

* SORTING METHOD

`Sort(arr, arr+n)` ⇒ C++

`Arrays.sort(Arr)` ⇒ Java.

TC: $O(N \log N)$

SC: Depends on Sorting Algorithm.

$O(1) \rightarrow O(\log N) \rightarrow O(N)$

Why Sorting ?

↳ Sorting makes searching faster.

Q: Given an Array of size N, We can remove one array element at a time.

Cost of removal = Sum of elements in Array just before removal of this element.

Find the minimum cost of removal of all the array elements.

A : {2, 1, 4}

Remove 2 : [2, 1, 4] : Cost = 2 + 1 + 4 = 7

Remove 1 : [1, 4] : Cost = 1 + 4 = 5

Remove 4 : [4] : Cost = 4

Cost = **16**

Remove 4 : [2, 1, 4] : Cost = 2 + 1 + 4 = 7

Remove 2 : [2, 1] : Cost = 2 + 1 = 3

Remove 1 : [1] : Cost = 1

Cost = **11**

Quiz A : {4, 6, 1}

Remove 6 : {4, 6, 1} : Cost = 4 + 6 + 1 = 11

Remove 4 : {4, 1} : Cost = 4 + 1 = 5

Remove 1 : {1} : Cost = 1

Cost = 17.

Eg [a, b, c, d] (0, 1, 2, 3)

Remove a : {a, b, c, d} : Cost = a + b + c + d : PS[3]

Remove b : {b, c, d} : Cost = b + c + d ⇒ PS[3] - PS[0]

Remove c : {c, d} : Cost = c + d ⇒ PS[3] - PS[1]

Remove d : {d} : Cost = d ⇒ PS[3] - PS[2]

Total Cost = a + 2b + 3c + 4d

↑ Largest element

↑ Smallest element.

a > b > c > d.

```
int minCost(arr[], n){
    Cost = 0
    Sort(Arr) // Sort in descending order.
    for(i=0; i<N; i++){
        Cost += (i+1) * A[i]
    }
    return cost;
}
```

TC: $O(N \log N) + O(N) \Rightarrow O(N \log N)$

SC: Depends on sorting algo.

Remaining content on 3rd Nov, 9:00 Pm