Sorting 3

Thursday, 4 March 2021 8:59 PM

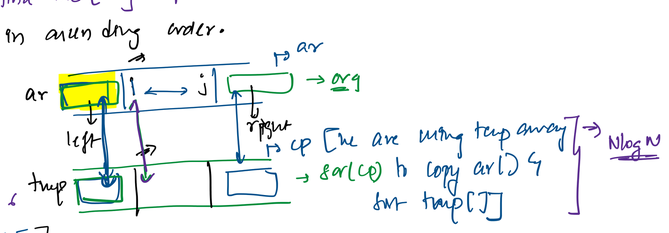
1a) Given an array of n elements, find the [length of smallest subarray] which when sorted entire is sorted in ascending order.

Ex1: 0 1 2 3 4 5 — 4 5 10 8 11 12 → 2

Ex2: 0 1 2 3 4 5 6 — 2 6 4 8 10 9 15 → 5

4 6 8 9 10

2 4 6 8 9 10 15



→ Nlog N

If we iterate from left first mismatch will be at i → [i - j]

ar[i] != temp[i]

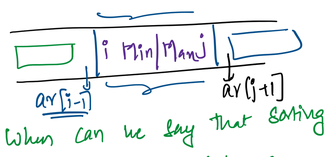
If we iterate from right first mismatch will be at j ⇒ length = j - i + 1

ar[j] != temp[j]

[TC: [NlogN + N] SC: O(N)]

sorting — finding i & j

Sub:



→ When can we say that sorting from i-j will sort entire subarray

1) All elements from (0, i-1) is sorted

2) " " " (j+1, n-1) is sorted

2 6 4 8 10 9 15

3) If ar[i-1] less than = min (ar[i-j])

4) If ar[j+1] greater than = max (ar[i-j])

final ans:

ar[]: 2 6 4 8 10 9 15 — i ⟷ j — sorted

[2 < = min = 4]

Max = 10 ≤ 15

ar[] → 0 1 2 3 4 5 6 7 8 9 10 11 — 2 4 6 8 11 7 19 5 10 18 24 30

i j

[update min & max value] ← i — [min = 5 M = 19] → [update min & max value]

i value

[1-6]

0 1 2 3 4 5 6 7 8 — 2 4 20 3 4 15 3 19 25 30

m = 4 M = 20 | m = 3 M = 20

2 ≤ 3, 20 ≤ 25

ar: 0 1 2 3 4 5 6 7 8 — 2 4 20 4 5 7 24 25 30

Pseudo Code:

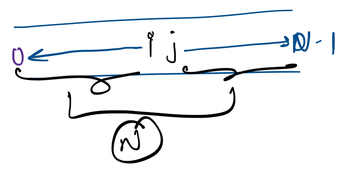
Given an array can we get i,j ⟺ i ≤ j, j ≤ i

Iterate from [i - j] or [j - i] & get Mv & mv

(i == 0 || j == N-1) Edge Cases

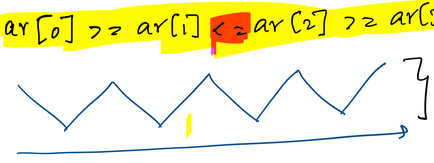
```
while(ar[i-1] > mv || ar[j+1] < Mv){
    while(i>0 && ar[i-1] > mv){
        i--;
        Mv = max(ar[i], Mv)      mv = min(ar[i], mv)
    }                            upd
    update max &
    while(j < N-1 && Mv > ar[j+1]){
        j++;
        Mv = max(ar[j], Mv)      mv = min(ar[j], mv)
    }
}
```

→ { i → 0, j → N-1 } TC: O(N) SC: O(1)



2a) Given an array construct a wave array

ar[0] >= ar[1] <= ar[2] >= ar[3] <= ar[4] - -



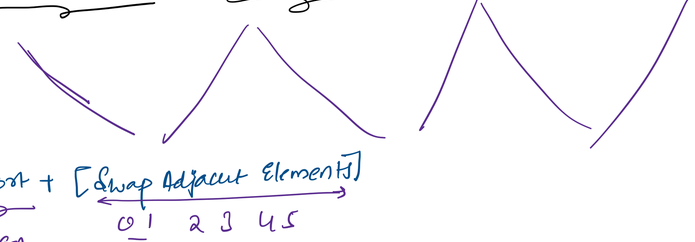
} wave array — Array looks exactly like wave

Ex: 3 6 9 8 3 4 7

Sort ⇒ 3 3 4 6 7 8 9

→ ar[0] ≤ ar[1] ≤ ar[2] ≤ ar[3] ≤ ar[4] ≤ ar[5]

ar[1] >= ar[0] ≤ ar[3] >= ar[2] ≤ ar[5] >= ar[4]



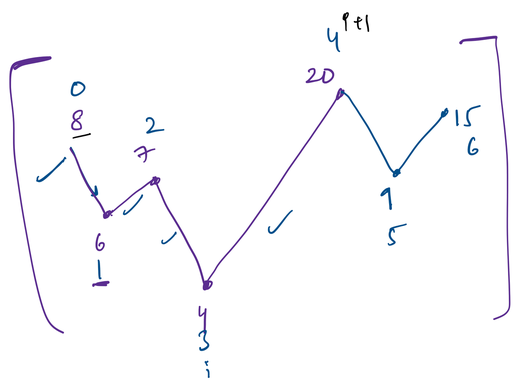
→ Sort + [Swap Adjacent Elements]

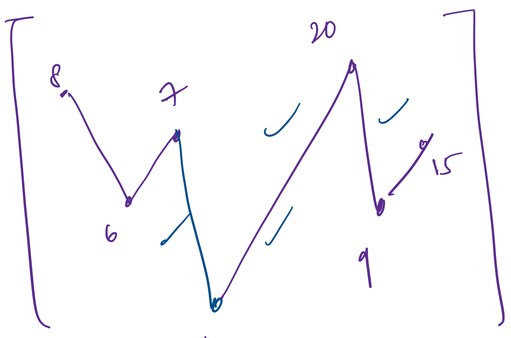
nlogn — 0 1 2 3 4 5 (n)

TC: O(n log n) + n

SC: N

Ex: 8 7 6 4 9 20 15





8 > 7 > 6

{8 > 6 < 7}

wave

4 < 9 < 20

4 < 20 > 9

```
for(int i=0; i<n-1; i++){
    if(i%2==0){ ⇒ Expected order
        if(ar[i] < ar[i+1]){
            swap(ar[i], ar[i+1])
        }
    } else {
        if(ar[i] > ar[i+1]){
            swap(ar[i], ar[i+1])
        }
    }
}
```

ar[i] >= ar[i+1]

TC: O(N)

SC: O(2)

Quick Sort } Recursion Time Complexity, Master's Theorem

Quick Select

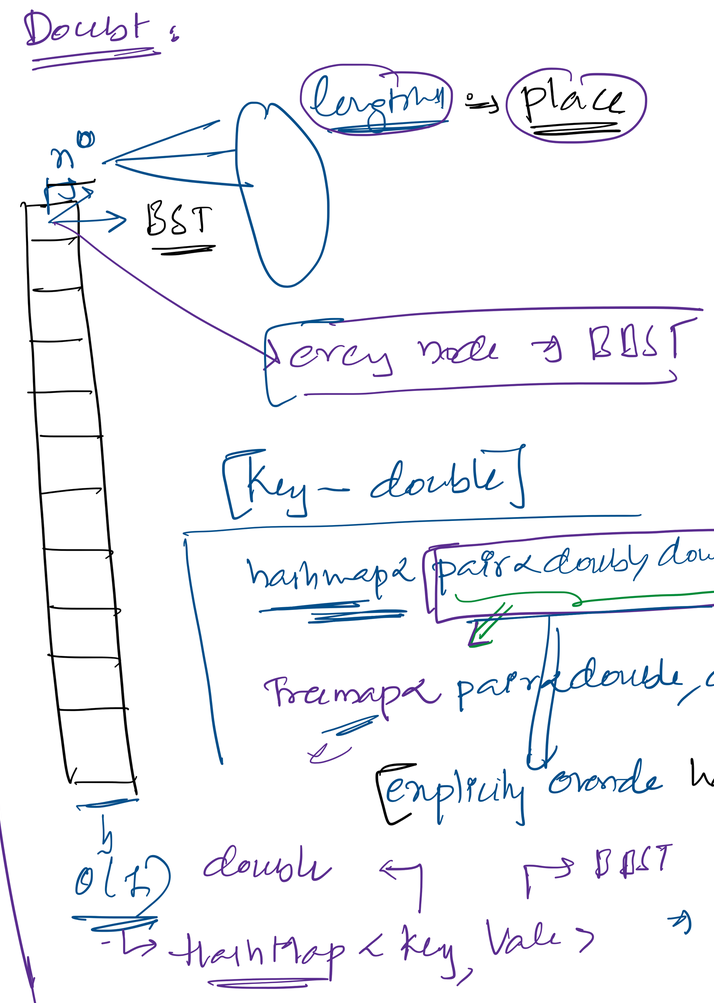
Max - Min in all subarray

Count Sort

Radix Sort

8 → 10:30

Doubt:



[Key - double]

hashmap< pair< double, double >, value > *] hashtable

Treemap< pair< double, double >, value >

10 30

31

32

33

[explicity overide hashfunctn]

O(1) double ← → BBST

→ HashMap< key, value >

BBST< key, value >

double → latitude

BBST< key, value >

double → longitude

log & log

HashMap< double, HashMap< double, place >>