21/8/2023

Greedy

Greedy → Max profit / Min loss.

A
iphone → 70K

B
iphone → 60K ✓

(10K Shoes 5K) → multiple factors

Q There is a limited time sale going on for toys,

A[i] → sale end time for i^{th} toy
B[i] → beauty of i^{th} toy.

It takes 1 unit of time to buy a toy & toy can only be bought if current time for i^{th} toy ≤ A[i].

Buy toys s.t. sum of beauty is maximized.

	1	2	3	4	5
A =	3	1	3	2	3
B =	6	5	3	1	9
T =	1	0		2	

	toy	B[i]
T=0	5	9
T=1	1	6
T=2	3	3
T=3		18

Ans = 5+6+9 = 20

	1	2
A =	1	2
B =	3	1500
T = →	0	1

Ans = 1503

Select toys w.r.t sale end time.

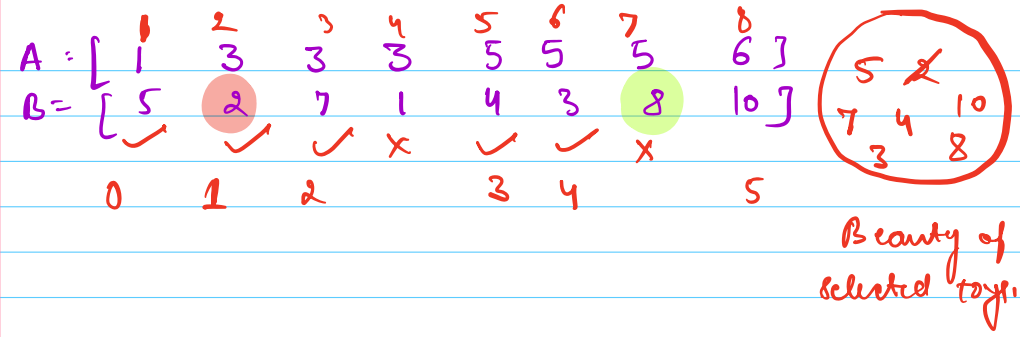
	1	2	3	4	5	6	7
A =	1	3	3	3	5	5	5
B =	5	2	7	1	4	3	8
T =	0	1	2		3	4	5

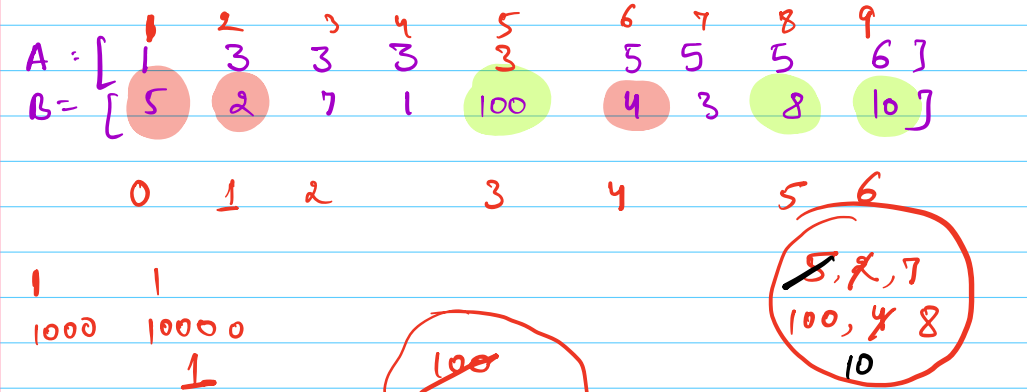
→ high beauty toys ⇒ purchase it ⇒ remove a previously purchased toy.

min beauty toy.

min heap.

Ans = 5+2+7+4+3
= 21






```
// sort wrt A[i]
t = 0
for i -> 0 to (N-1) {
    if (A[i] > t) {
        h.insert(B[i]);
        t++;
    }
    else { // sale ended for the product
        if (h.peek() < B[i]) { // replace with min beauty toy already bought.
            h.getMin();
            h.insert(B[i]);
        }
    }
}
Ans = Σ elements present in heap.
```

TC: O(N log N)
SC: O(N)

Q2 There are N children with marks, teacher has to give them candies. s.t.

a) Every child has atleast one candy.
b) Children with higher marks have more candies than their neighbours $(i-1)$ i $(i+1)$

Find min candies required to do so.

A = [1 5 2 1]
C ⇒ [1 3 2 1]

Ans = 1+3+2+1 = 7

A = [4 4 4 4 4]
C = [1 1 1 1 1]

Ans = 5.

1) ∀ i C[i] >= 1

2) if A[i] > A[i-1]
   C[i] > C[i-1]
   C[i] = C[i-1] + 1

3) if A[i] > A[i+1]
   C[i] > C[i+1]
   C[i] = max(C[i+1]+1, C[i])

A = [1 5 2]
C = [1 ~~1~~ 2 1] ⇒ Ans = 4

	1	2	3	4	5	6	7	8	9
A =	1	6	3	1	10	12	20	5	2
C =	1	~~1~~ 2	~~1~~ 2	1	~~1~~ 2	~~1~~ 3	~~1~~ 4	~~1~~ 2	1

L→R

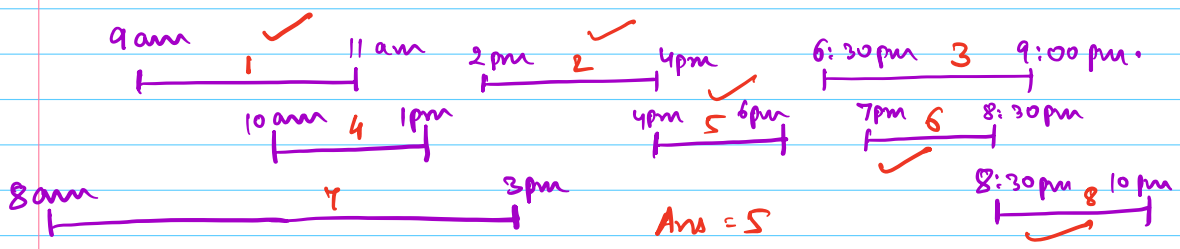
R→L

```
if C[i] = 1.
for i -> 1 to (N-1) {
    if (A[i] > A[i-1]) {
        C[i] = C[i-1] + 1;      <- greedy.
    }
}
ans = C[N-1];
for i -> (N-2) to 0 {
    if (A[i] > A[i+1]) {
        C[i] = max(C[i+1]+1, C[i]);
    }
    ans += C[i]
}
return ans;
```

TC : O(N)
SC : O(N)

Break till 8:49 am IST.

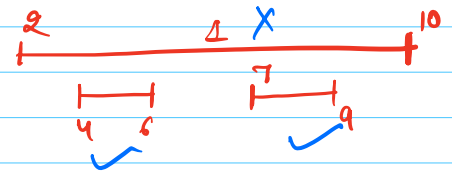
Q3 Given N jobs with start & end time. And max jobs that can be completed if only one job can be done at a time.



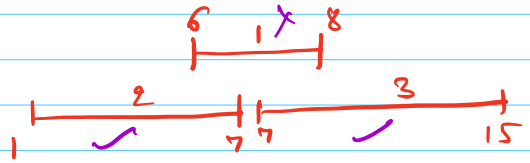
	1	2	3	4	5	6
S =	1	5	8	7	12	13
E =	2	10	10	11	20	19

Ans = 3

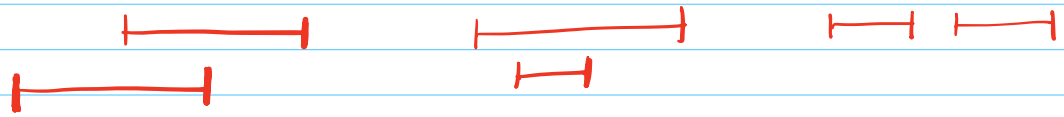
1) Start time



2) Duration



3) End time → early start time & less duration



```
// sort w.r.t E[i]   // sorting
ans = 1 , e = E[0]

for i -> 1 to (N-1) {
    if (S[i] >= e) {
        ans++;
        e = E[i]
    }
}

return ans;
```

TC: O(N log N)
SC: O(1)