Welcome ☺

Agenda: Greedy Algo.
3 ques^ns.

Greedy → Max Profits / Min Loses

Aman	Mukesh
Iphone ⇒ 70K	Iphone ⇒ 60 K ✓

(10 K Shoes 5K) ⇒ depends on multiple factors.

Q1 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. And toy can only be bought if current time of i^{th} toy < A[i]

Buy toys s.t sum of beauty is maximized.

eg.

$$\begin{array}{cccccccc} & & 0 & 1 & 2 & 3 & 4 & \\ A & [& 3 & 1 & 3 & 2 & 3 &] \\ B & [& 6 & 5 & 3 & 1 & 9 &] \end{array}$$

	toy	A[i]
t=0	4	9
t=1	0	6
t=2	2	3
		18

	toy	A[i]
t=0	1	5
t=1	0	6
t=2	4	9
		20

Q

$$\begin{array}{cccc} & 0 & 1 & \\ [& 1 & 2 &] \\ [& 3 & 1500 &] \end{array}$$

Ans = 1503

⇒ Select toys not based just on the beauty of it but select them w.r.t sale end time

		0	1	2	3	4	5	6	
A	[	1	3	5	3	3	5	5	]
B	[	5	2	4	7	1	3	8	]

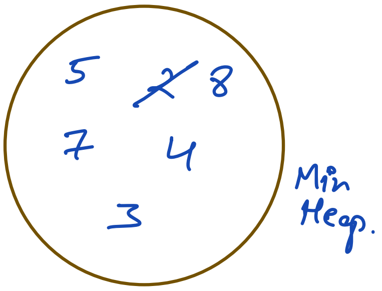
Step 1 ⇒ Sort array

		0	1	2	3	4	5	6	
A	[	1	3	3	3	5	5	5	]
B	[	5	2	7	1	4	3	8	]

very less beautiful.

high beauty toy
⇒ Purchase it
⇒ remove smallest toy

	toy	B[i]
t=0	0	5
t=1	1	2
t=2	2	7
t=3	4	4
t=4	5	3
t=5		



Code

```
1) sort. w.r.t to A
   t=0
   for ( i→ 0 to N-1)
   {
       if ( A[i] > t ) {
           Minheap.insert ( B[i])
              t++
       }
       else {   // sale end time ≥ t
           if ( Minheap.root < B[i] ) {
               Minheap. getMin ()
               Minheap. insert ( B[i])
           }
       }
   }
```

T.C → O(N log N)
SC → O(N)

Q2 Candy Distribution

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

a> Every child has atleast 1 candy.

b> Children with higher marks must have more candies than their neighbours. $i-1 < i > i+1$

Find min candies req. to do this.

eg.

A :	[1	5	2	1]
C →	1	~~1~~ ~~111~~ ~~2~~3	~~1~~ 2	1

1) $\forall i$, $C[i] \geq 1$

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

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

Quiz eg:

	[4	4	4	4	4]
C	1	1	1	1	1

count = 5

eg:

A :	[1	6	3	1	10	12	20	5	2]
C :	1	~~1~~ ~~2~~ 3	~~1~~ 2	1	~~1~~ 2	~~1~~ 3	~~1~~ 4	~~1~~ 2	1

Cond.2 →

← cond3

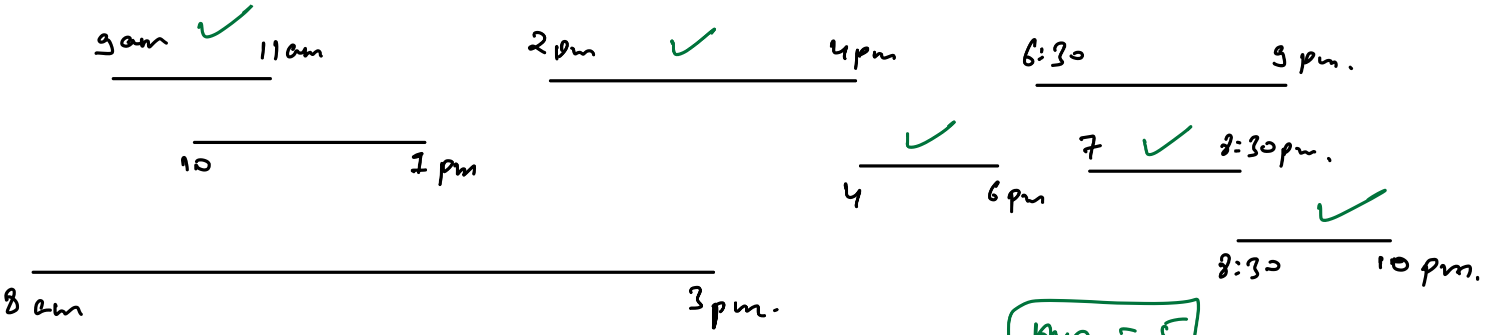
Code

```
∀i C[i] = 1
for ( i → 1 to N-1) {
    if ( A[i] > A[i-1])   C[i] = c[i-1]+1
}
```

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

T.C → O(N)
S.C → O(N)

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



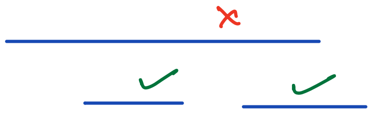
$\boxed{\text{Ans} = 5}$

Quiz

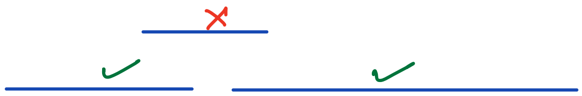
S	[1	5	8	7	12	13]
E	[2	10	10	11	20	19]

Ans = 3

Start Time X



Duration. X



End Time → Early start time & less duration

⇒ Early End Time

Code

```
1. // Sort based on End Time.

    ans = 1          e = E[0]

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

T.C → O(NlogN)

S.C → O(1)