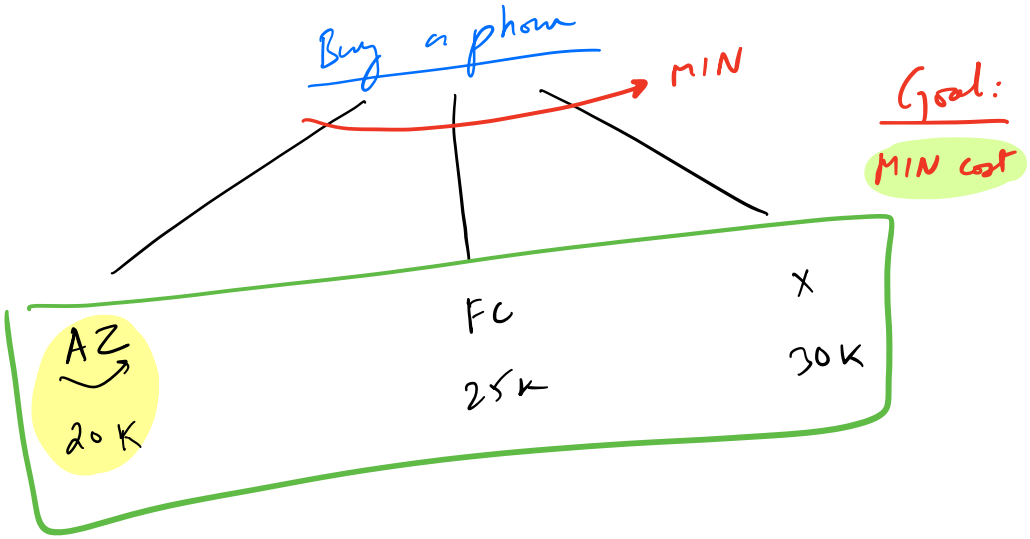
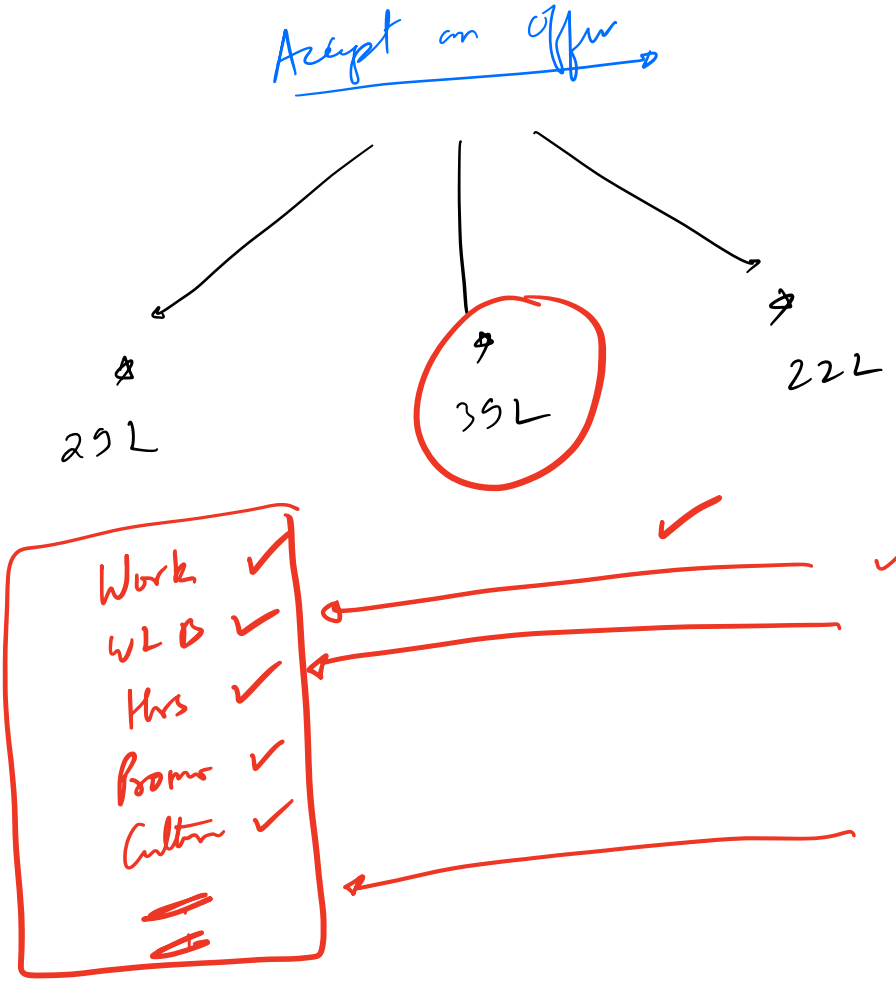
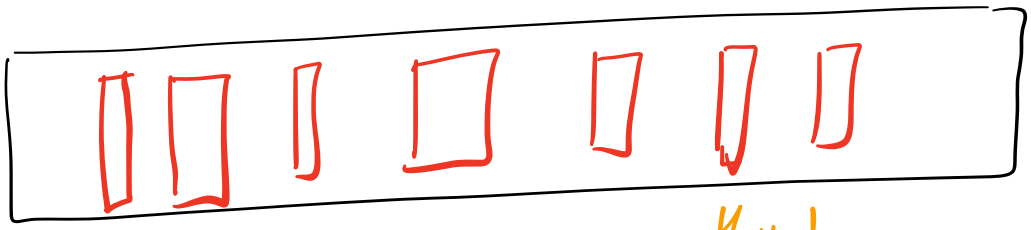
Greedy Algos



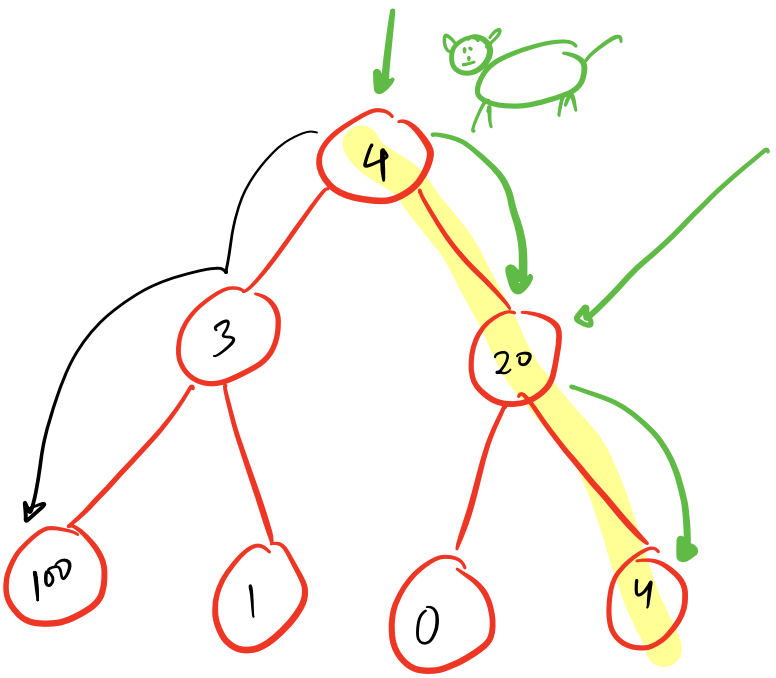


① Read 20 books → [quickly]



Choose books which are thin!

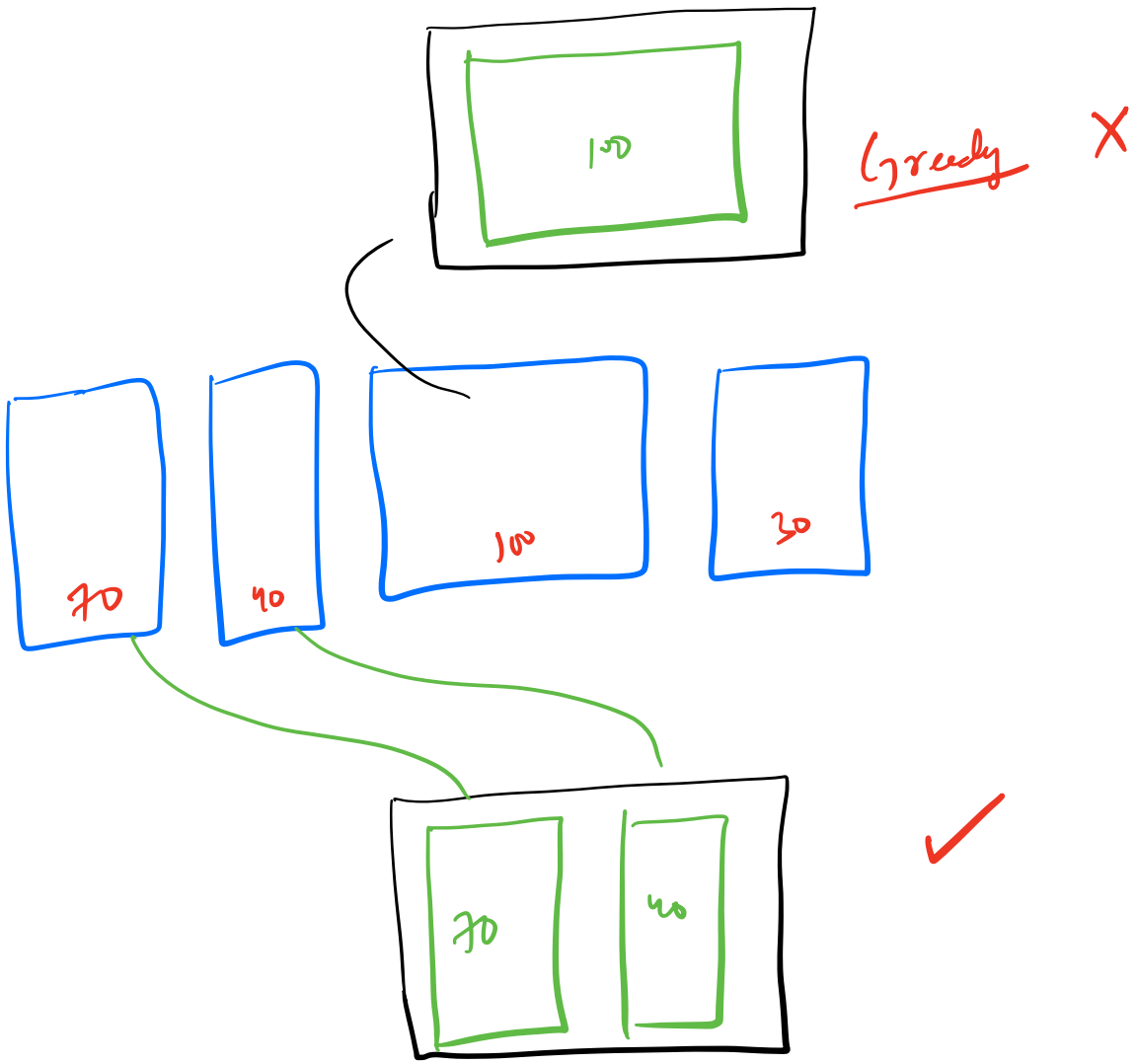
① Greedy strategies might not always lead to most optimal answers!



(Greedy)

2^8 // X

107



Greedy X

✓

Greedy:

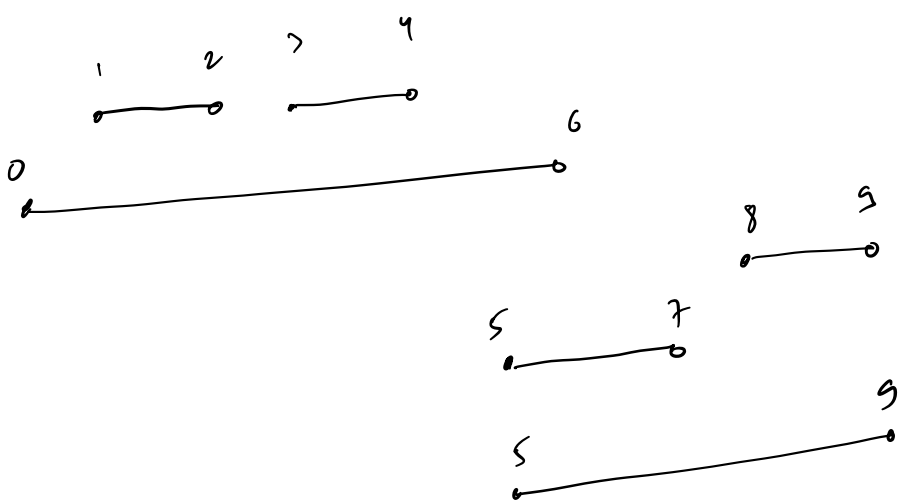
Think greedily at every step w/o considering future outcomes!

Activity Selection Problem

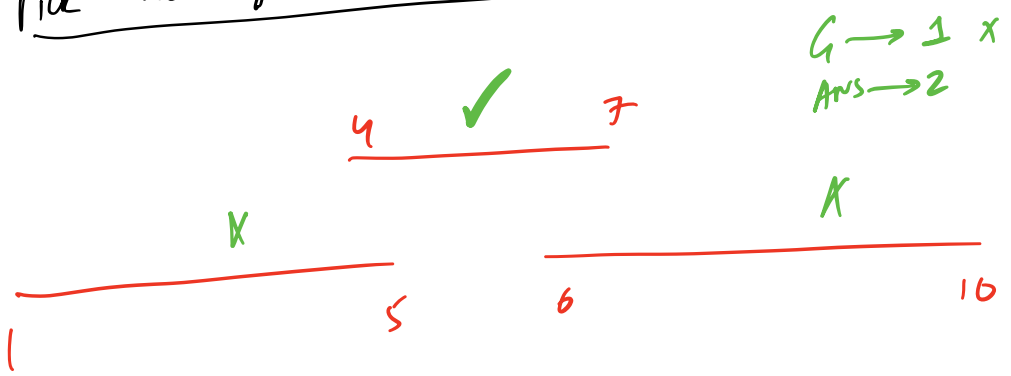
Given N activities → [S, E] S < E

1. At a time, you can perform only 1 ACT-
2. Find the MAX no. of activities you can perform!

	0	1	2	3	4	5
S	1	3	0	8	5	5
E	2	4	6	9	7	9

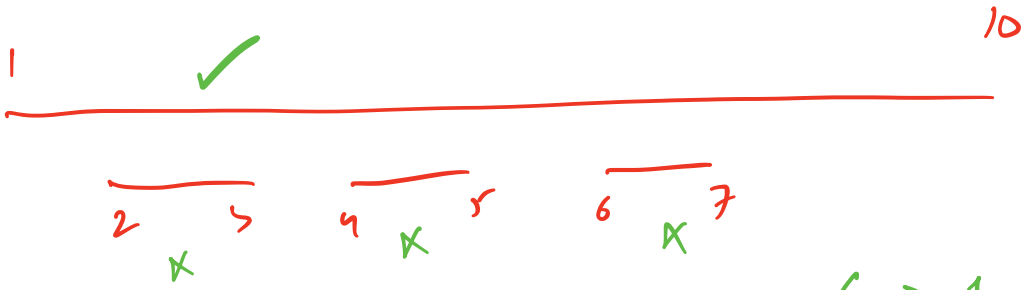


1) Pick activity with the least duration first



G → 1 ✗
Ans → 2

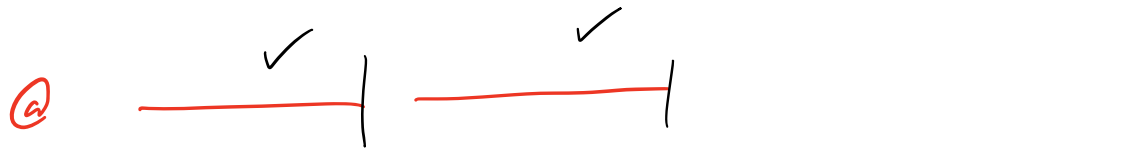
2) Pick activity that starts early

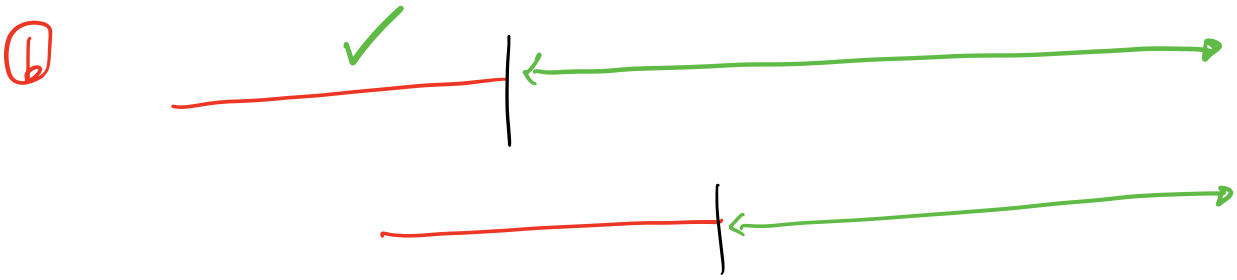


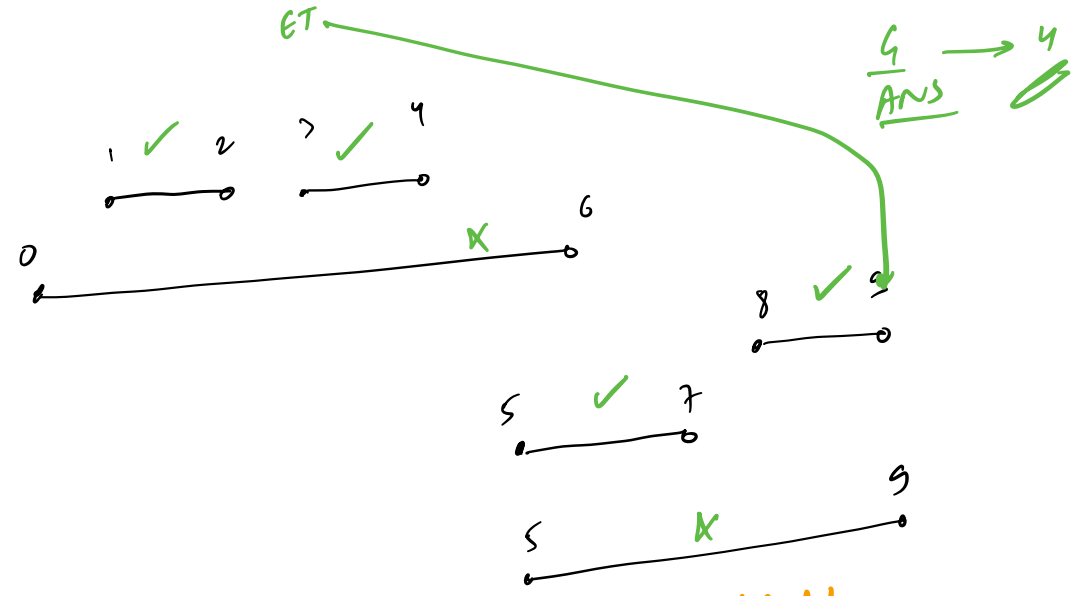
G → 1

Ans → 3

3) Pick ACT with smallest End time first







1) Sort using ET → NlgN

2) Iterate on the ACTIVITIES → N
& check if $ST_i < ET_j$ the last ACT chosen
→ true : X
→ false : ✓ ET = ET_i

TC: O(N lgN)

SC: O(1)

JOB SCHEDULING

Given N jobs → reward
→ deadline

- At any time, a single job can be performed
- A job takes 1 unit of time to complete

MAXIMIZE the total amount of reward!

Jobs	deadline	Reward
a	3	100 ✓
b	1	19 ✗
c	2	27 ✓
d	1	25 ✗
e	3	30 ✓

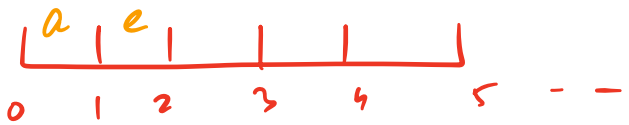
157

1) Pick Jobs with highest Rewards → Priority

Jobs	deadline
a	3
b	1
c	2
d	1
e	3

Reward	
100	→ 1 ✓
19	→ 5 ✗
27	→ 3 ✗
25	→ 4 ✗
30	→ 2 ✓

130

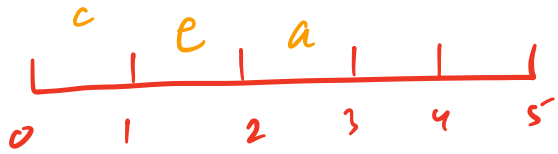


2) Pick Jobs with highest rewards & put them as close as possible to their deadline →

Jobs	deadline
a	3
b	1
c	2
d	1
e	3

Reward	Points
100	→ 1 ✓
19	→ 5 ✗
27	→ 3 ✓
25	→ 4 ✗
30	→ 2 ✓

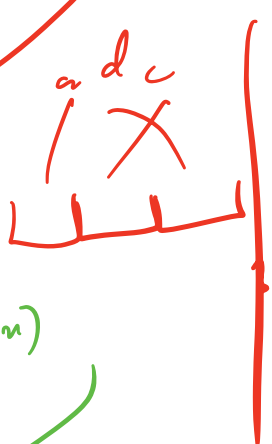
157 ✓

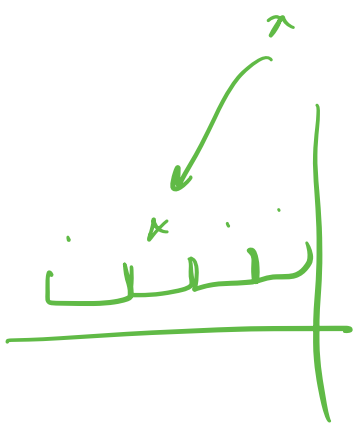


II Sort on the basis of deadline → (Asc)

Jobs	deadline	Reward	priority
a	3	100	4 ✓
b	1	19	1 ✗
c	2	27	3 ✓
d	1	25	2 ✓
e	3	30	5

t = 3



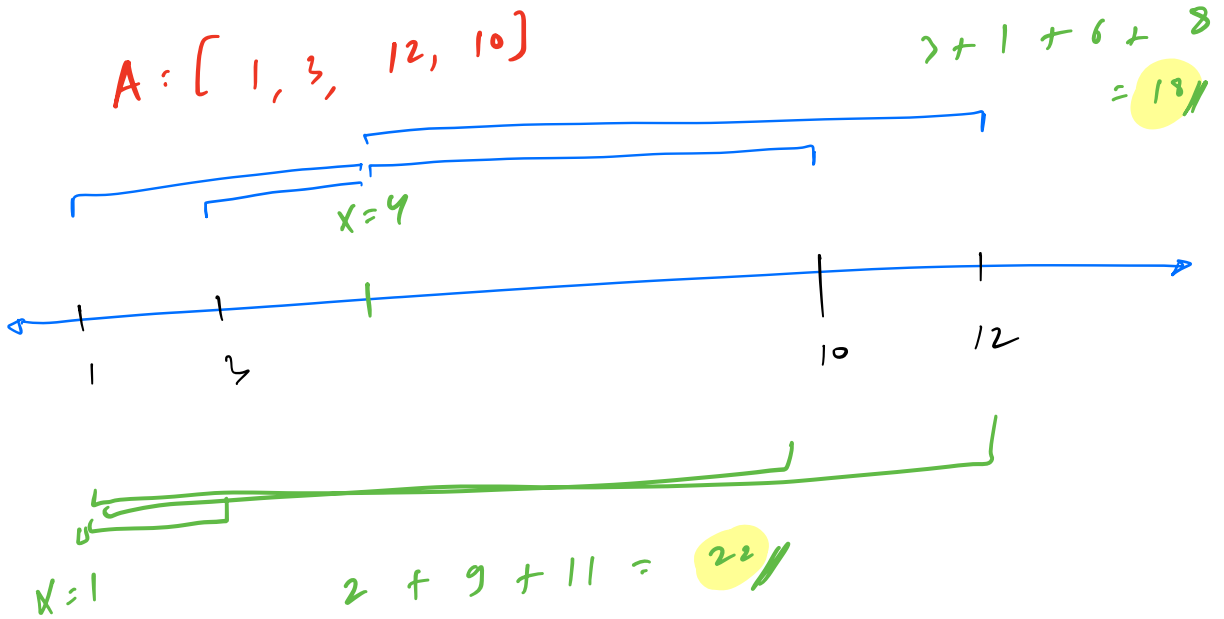


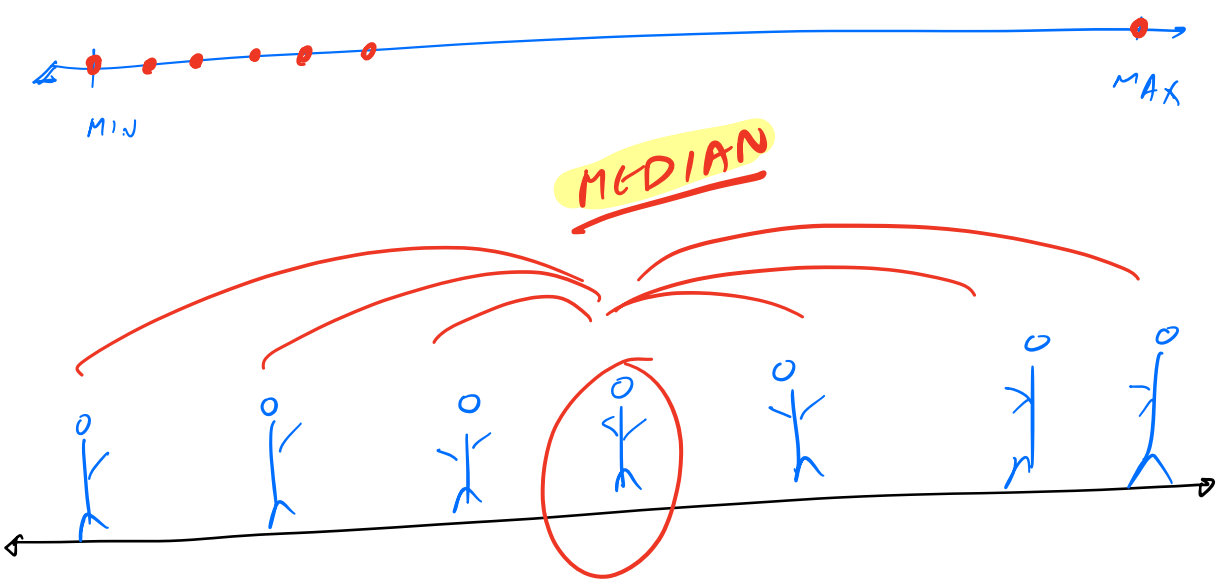
insert(x)

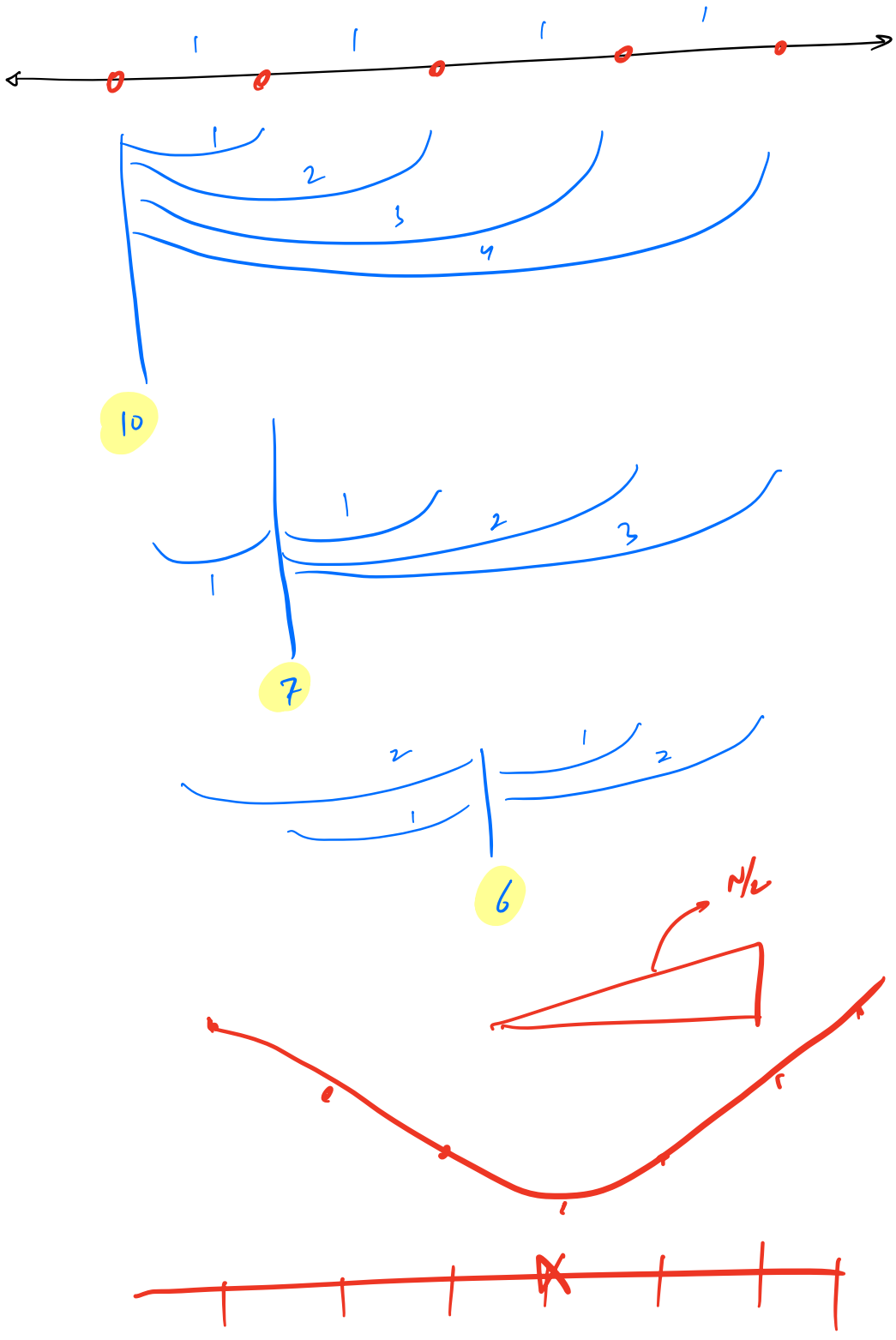
delMin() getMin()

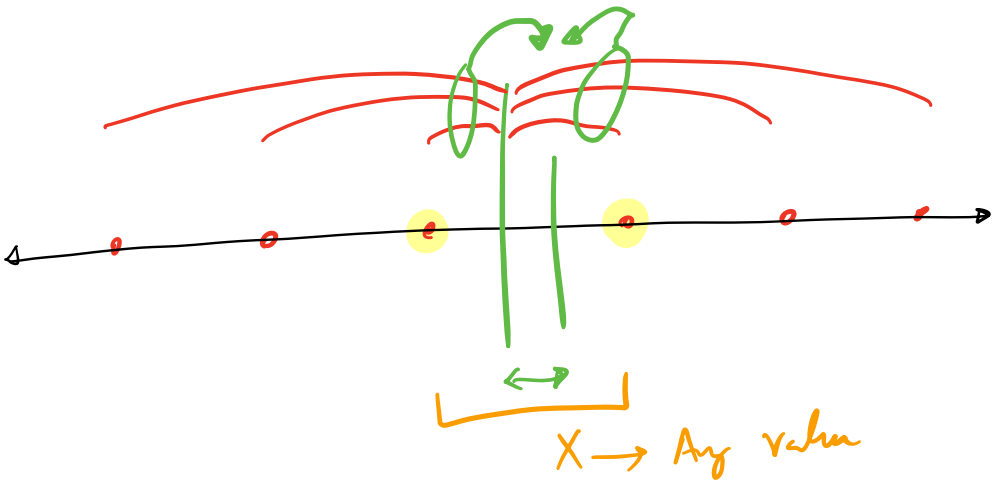
MIN HEAP

Q Given an Array. find X s.t. $\sum |A_i - X|$ is MINIMIZED!






1
1
1
1
1
2
3
4
10
1
2
3
1
7
2
1
1
2
6
N/2



(9) . . X . X . . X X

Q N children → Ratings.
You want to give candies to all children!

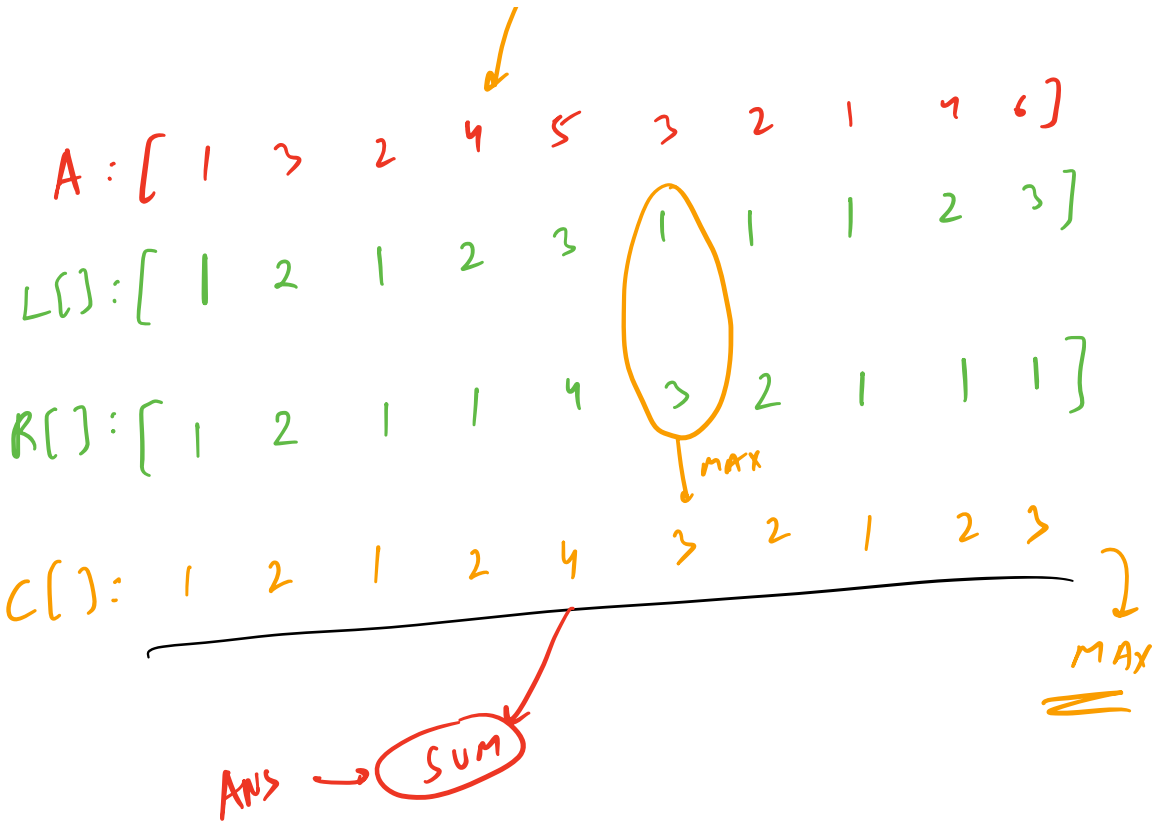
→
1) Everyone should get ≥ 1 candy.
2) Child with higher rating than it's neighbour should get more candies than that neighbour.

GOAL: MINIMIZE the total # of candies distributed!

A: [1, 5, 2, 1]

X C: [5 10 7 5]

C: [1 3 2 1]


A : [1 3 2 4 5 3 2 1 4 6]
L[] : [1 2 1 2 3 1 1 1 2 3]
R[] : [1 2 1 1 4 3 2 1 1 1]
max
C[] : 1 2 1 2 4 3 2 1 2 3
MAX

Ans
SUM