Thursday, 28 October 2021 12:43 PM

Q Maximize sum of array after k negations.

[2, -3, -1, 5, -4] , k = 2

output → 13

-4 → 4

[2, -3, -1, 5, 4]

-3 → 3

[2, 3, -1, 5, 4] → ans

→ 2 + 3 - 1 + 5 + 4 = 13

Q Job Scheduling

Each job has a deadline and a profit associated.

Eg ①

Job	Deadline	Profit
→ a	4	20
b	1	10
c	1	→ 40
d	1	30

c	a		
1	2	3	4

T Profit → 40 + 20 = 60

→ 1 job ⇒ 1 day
→ Job can be performed in any order
→ Not necessary to perform all jobs
→ Maximise profit.

Eg ②

Job	d	p
a	2	100
b	1	19
c	2	27
d	1	25
e	3	15

ans → 142

c	a	e

→ 100 + 27 + 15

Let's say

a	1	50
b	1	100
c	1	20

choose the job with max profit

now,

a	2	100
b	1	100
c	3	100

what should be the ordering here ??

a/b/c			c

(1 option)

(3 options)

For day 1, I have all the options.

* Always perform a job on its deadline day.

① Which job to choose first ?? → offers max profit

② When to prefer a job ?? → as close to the deadline as possible

① Sort the jobs as per profit
② Try to find a slot.

J	d	p
a	2	100
b	1	19
c	2	27
d	1	25
e	3	15

→

J	d	p
→ a	2	100
→ c	2	27
→ d	1	25
→ b	1	19
→ e	3	15

→

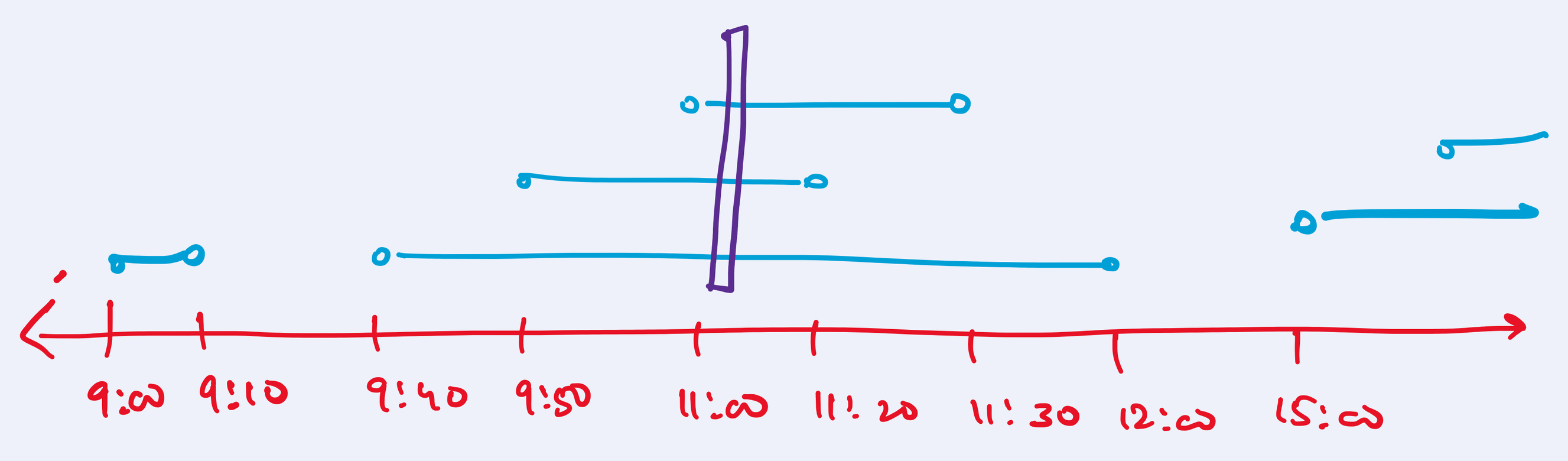
x | c | a | e | → 100 + 27 + 25 = 142

1 2 3

Q Min no of platforms required.

A : 9:00 9:40 9:50 11:00 15:06 18:00

D : 9:10 12:00 11:20 11:30 19:00 20:00

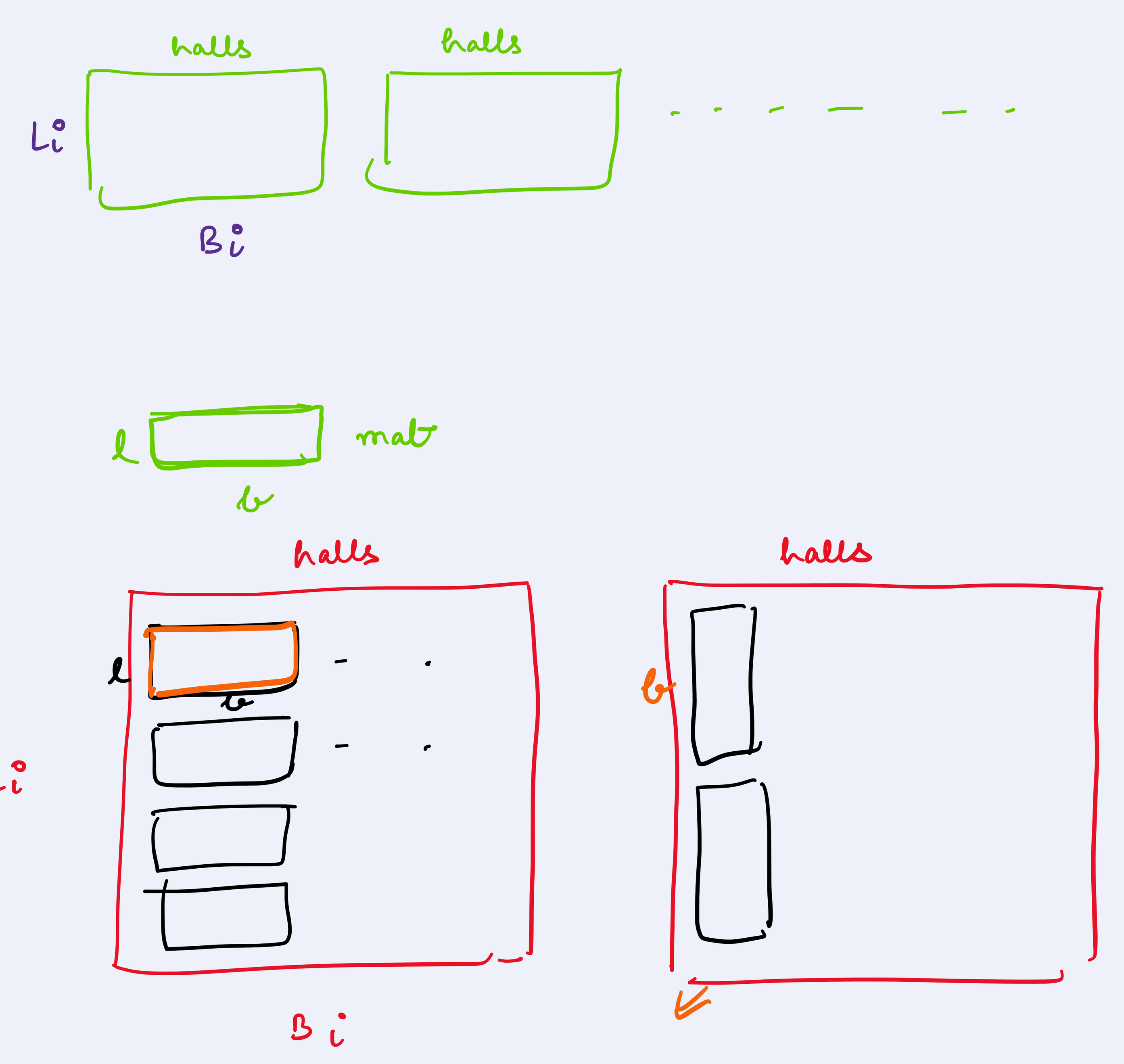


Max no. of platforms we need at a time is the max no. of trains at a time on the platform.

① Sort on the basis of arrival time
② Sort on the basis of destination time

→ Similar to meeting room problem.

Q → Codechef (Ramdev)



$$\sum_{i=0}^{n-1} \max\left(\frac{L_i}{l} \times \frac{B_i}{b}, \frac{L_i}{b} \times \frac{B_i}{l}\right)$$

```
mat → l, b.
result = 0

for (i=0; i<n; i++) {
        int Li = scn.nextInt();
        int Bi = scn.nextInt();
        result += max((Li/l)*(Bi/b), (Li/b)*(Bi/l));
```