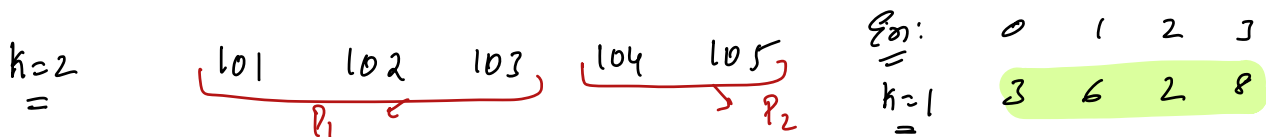
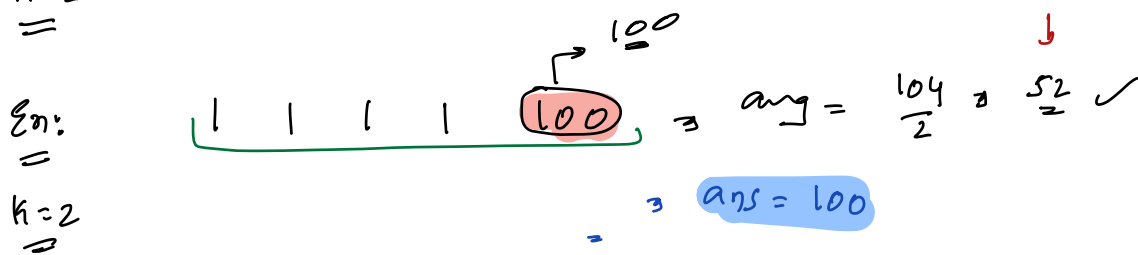
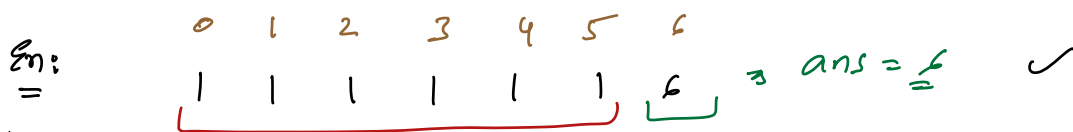
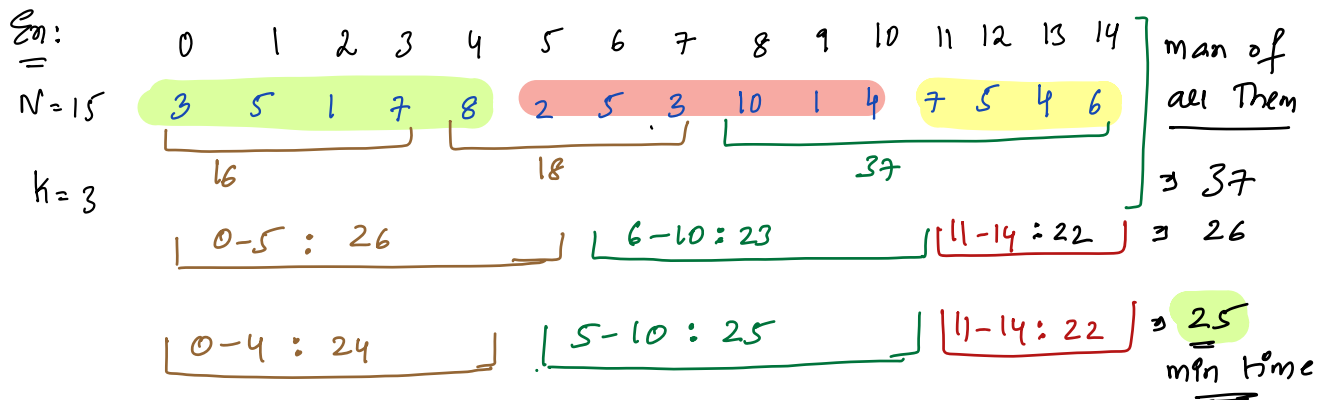


500 Given N tasks, k workers & time taken for each task
find min time in which we can complete all tasks. No Sorting

Note 1: A single worker can only do continuous set of tasks

Note 2: All workers start their assigned tasks at same time

Note 3: A task can only be assigned to a single person



19

Ex:

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	} Time <u>22</u>
N=15 :	3	5	1	7	8	2	5	3	10	1	4	7	5	4	6	
<u>k=4</u>	16				18				22				15			
	0-3 : 16				4-6 : 15				7-9 : 14				10-12 : 16		Tail rem	

Can we finish : 22 23 24 25 ...

!-randomly gave a number ↑

~~11~~ ~~12~~ ~~13~~ ~~14~~ ~~15~~ ~~16~~ : Can we finish in

k=4?

Binary Search

1) Target: min time

2) Search: [max(arr) Total sum]

Sum/k
|
min
|
max

3) Divide or not?

<u>lo</u>	<u>hi</u>	<u>mid</u>	
10	71	<u>40</u>	ans = 40, left h = mid - 1
10	39	<u>24</u>	ans = 24, left h = mid - 1
10	23	<u>16</u>	goto right l = mid + 1
17	23	<u>20</u>	goto right l = mid + 1
21	23	<u>22</u>	ans = 22 goto left
<u>21</u>	21	<u>21</u>	goto right l = mid + 1
<u>22</u>	21	<u>21</u>	ans = 22

// Pseudo Code

10:35 break

lo = (min(arr[])) hi = (sum of arr[])

ans = hi

while(lo <= hi) {

m = (lo + hi) / 2

if(check(m, arr[], N, k))

// Means we can finish in m time

ans = m;

hi = m - 1

else {

lo = m + 1

}

return ans;

l = min(arr[])

hp = sum of arr[]

n = number of Elements

where we are we are

doing $n = [hp - l + 1]$

TC: $\log_2 n \times O(N)$

TC: $N \log_2 n$

SC: $O(1)$

// bool check(int time, int arr[], int N, int k) {

s = 0, c = 0

for(i = 0; i < N; i++) {

s = s + arr[i]

if(s > time) { c++; s = arr[i]; }

}

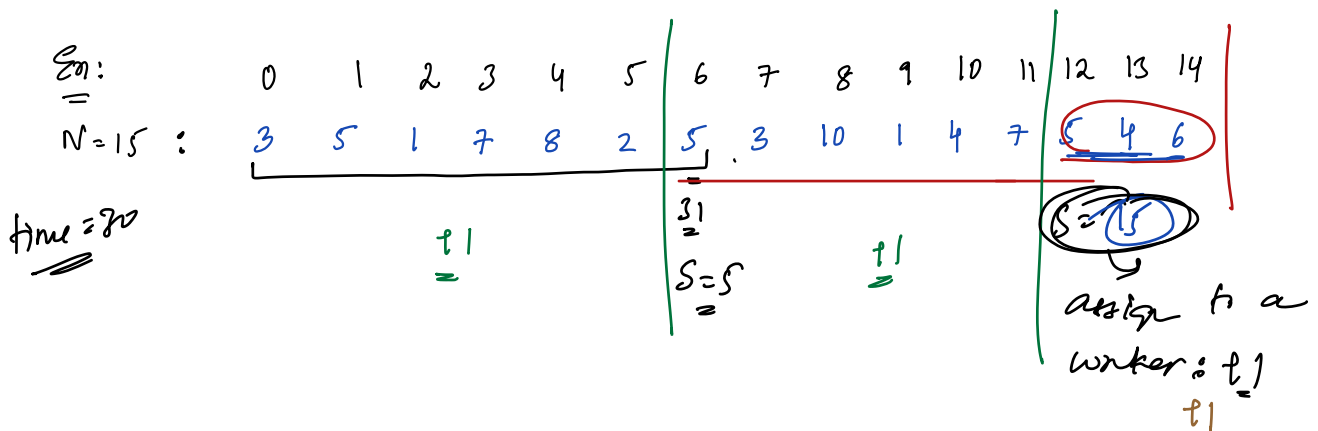
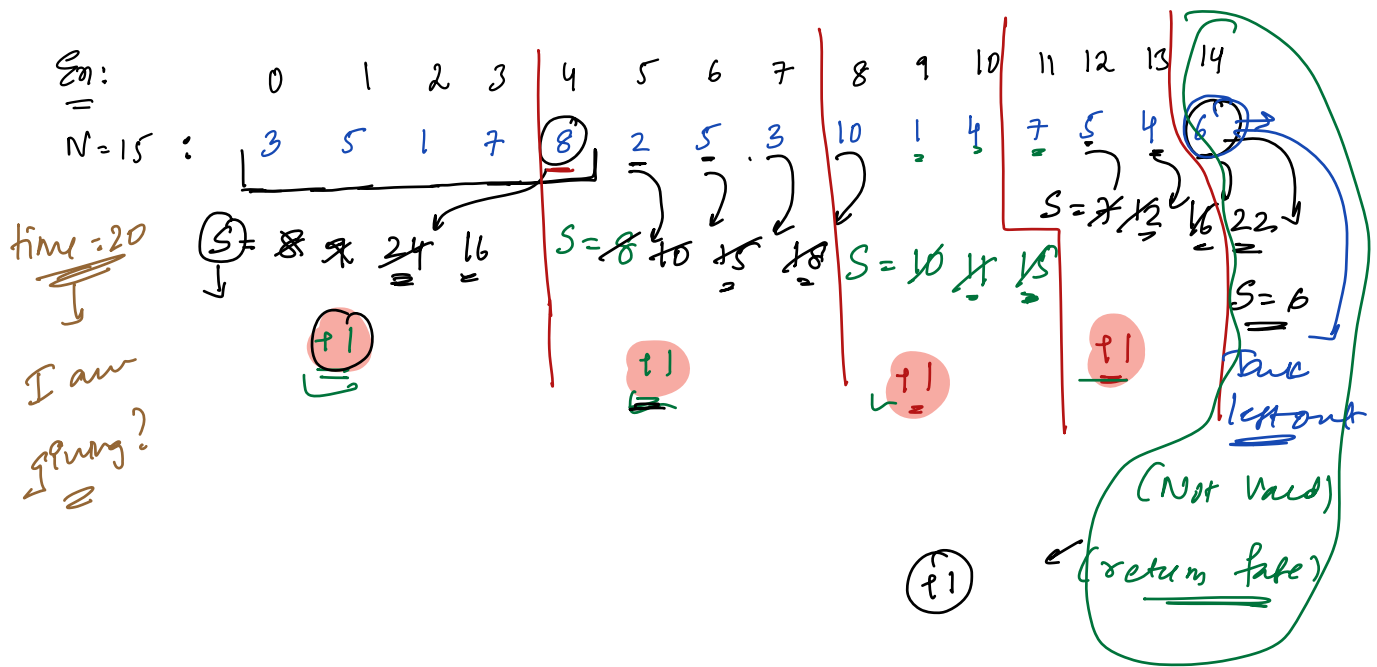
if(s != 0) { c++; }

if(c <= k) { return True }

else { return False }

TC: $O(N)$

SC: $O(1)$



$$\rightarrow (N) \Rightarrow (key^N_2)$$

$$l1 = \text{max}(C) \quad hp = \text{sum}(C)$$

$$Q = (hp - lo + 1)$$

log₂ n

$$C = (3)$$

$$3 < 4$$

20) Given N Cows & M Stalls, all M stalls are on a line at different locations. Place all N Cows such a way min distance between any 2 cows is maximized

Note: In a stall only 1 cow can be present

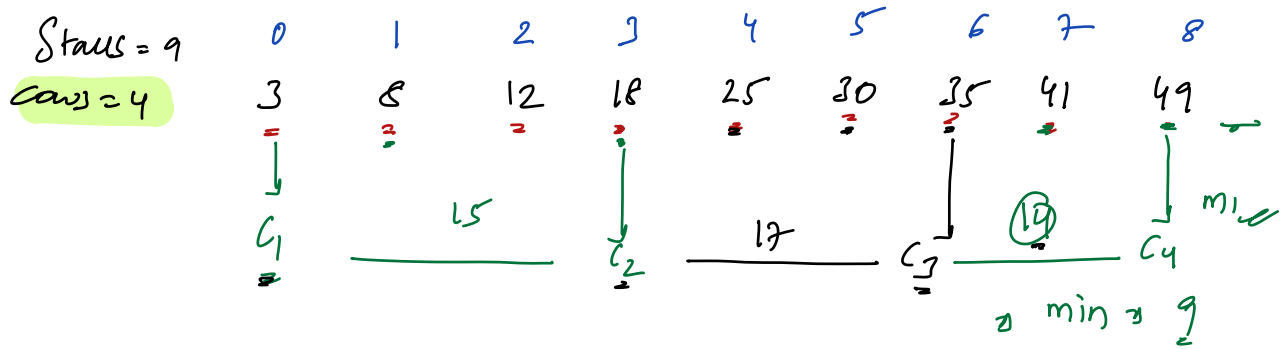
Note: All Cows have to be placed. {stalls sorted}

Ex1:

	0	1	2	3	4	min
Stalls = 5	1	2	4	8	9	
Cows = 3	C_1	C_2	C_3			1
	C_1		C_2	C_3		3
	C_1			C_2	C_3	1
						ans = 3

Ex2:

	0	1	2	3	4	5	6	7	8	min
Stalls = 9	2	6	11	14	19	25	30	37	43	3
Cows = 4	C_1	C_2	C_3	C_4						9
	C_1		C_2	C_3	C_4					12
	C_1			C_2	C_3	C_4				



At least 20 distance between 2 cows

* 21 22 23 24 25 -
X X X X X X X X

7 = 8

At least 8 distance between 2 cows ✓

ans = _____

4 5 6 7 8
←

Target : maintain

Search space : [Min adj diff, last - first]

Divide 2 half & discard

BS

lo: hi mid
4 46 25
h = mid - 1

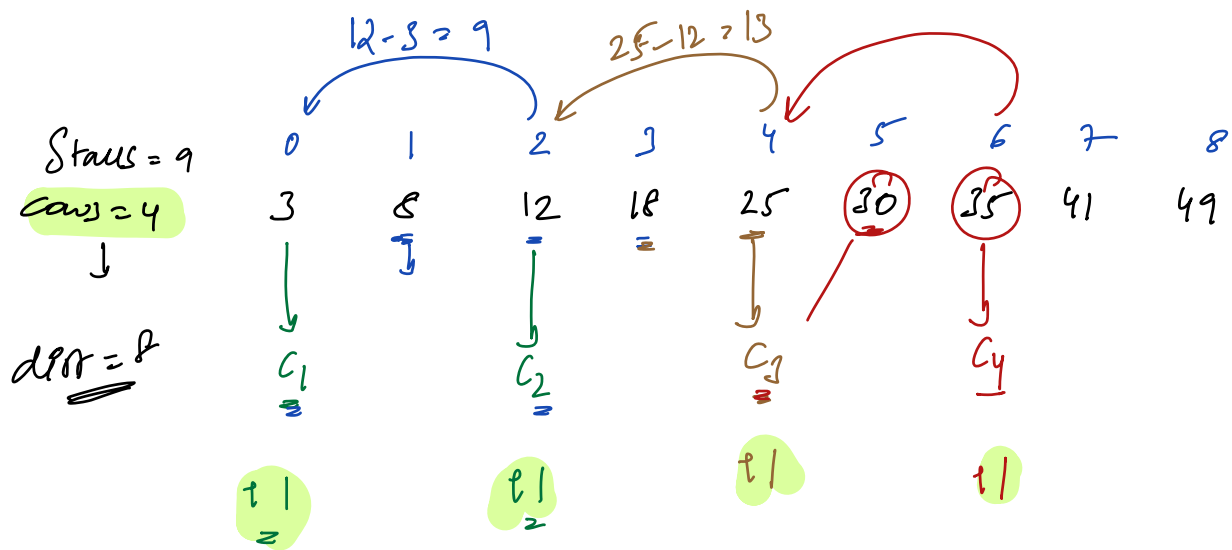
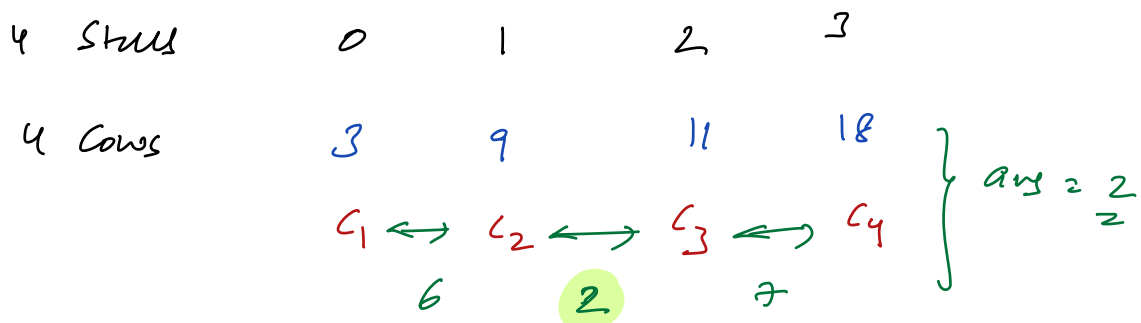
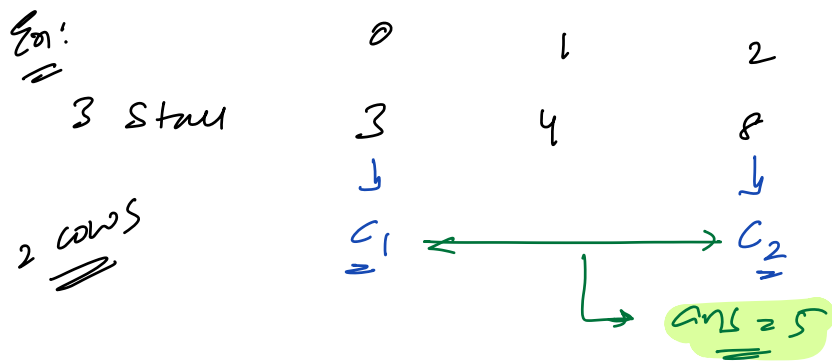
4 24 14
ans = 14
l = mid + 1

15 24 14
h = mid - 1

15 18 16
h = mid - 1

15 15 15
h = mid - 1

15 14 {Break}



// Pseudo code :

```

lo = (min adjo dist)  hi = (last - first)  ans = —
while (lo <= hi) {
    mid = (lo + hi) / 2
    if (check(mid, arr[], N, C)) {
        ans = mid; // we can atleast place at mid cows
        lo = mid + 1;
    }
    else {
        hi = mid - 1;
    }
}
return ans;

```

min distance we need to place them
min dist^o we need to place them
no. of cows we need to place

$hi = (last - lo + 1)$
 $\log_2 N \times [N] : TC$
 $SC : O(1)$

bool check(dls, arr[], N, cows)

```

lastcow = A[0], c = 1
for (i = 1; i < N; i++) {
    if (A[i] - lastcow >= dls) {
        // we can place here
        lastcow = A[i]
        c++
        if (c == cows) { return true; }
    }
}
return false;

```

$TC : O(N)$

TC $\Rightarrow \left\{ \begin{array}{l} \rightarrow \text{Allocate Books} \\ \text{Painter problem} \end{array} \right\}$

(Day 8 clon)