cow stalls
↳ aggressive

N=5

k=3

0	1	2	3	4	} locations
1	2	4	8	9	

$C_1 \xleftrightarrow{1} C_2 \xleftrightarrow{2} C_3$

$C_1 \xrightarrow{2} C_2 \xrightarrow{4} C_3$

$C_1 \xrightarrow{7} C_2 \xrightarrow{1} C_3$

$C_1 \xrightarrow{3} C_2 \xrightarrow{5} C_3$

min

1
↓
2
(1)

3
↓
maximum min dist b/w any two cows

$^5C_3$
= 10

Q given N cow stalls, M cows.

A[i] → position of each stall in asc order

N ≥ M

2 cows in [illegible]

Maximise the minimum distance b/w cows.

BF

$\{ {}^NC_M * M \}$

$\frac{n!}{m!(n-m)!}$

N ≥ M

3 cows

0	1	2	3	4
1+2	2	4+2=6	8 +4=12	9

1 $C_1 \xrightarrow{1} C_2 \xrightarrow{2} C_3$

ans 2 $C_1 \xrightarrow{3} \times\ C_2 \xrightarrow{4} C_3$

ans → 3 $C_1 \xrightarrow{3} \times\ C_2 \xrightarrow{4} C_3$

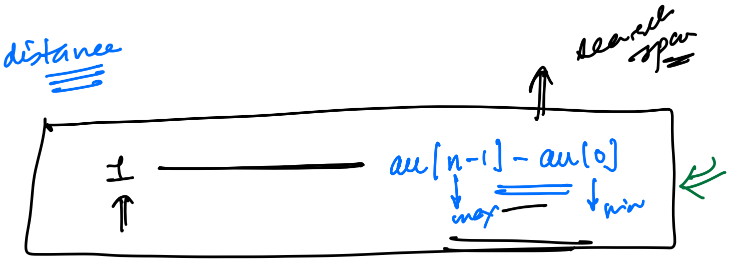
4 C_1 ✗ ✗ C_2 ✗

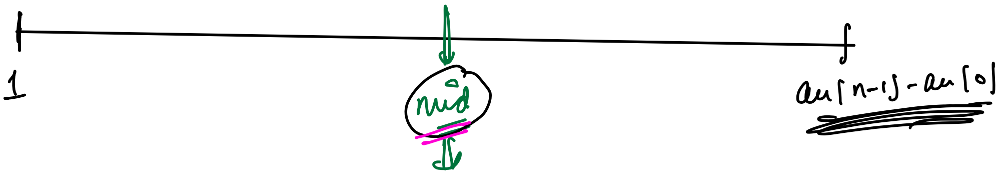
TTTTFFFFFF
⟶
BS

target = min distance

N/M ✗

Search space:





can I place my M cows with atleast a distance of mid

Yes → ans = mid, l = mid+1

No → l = mid-1

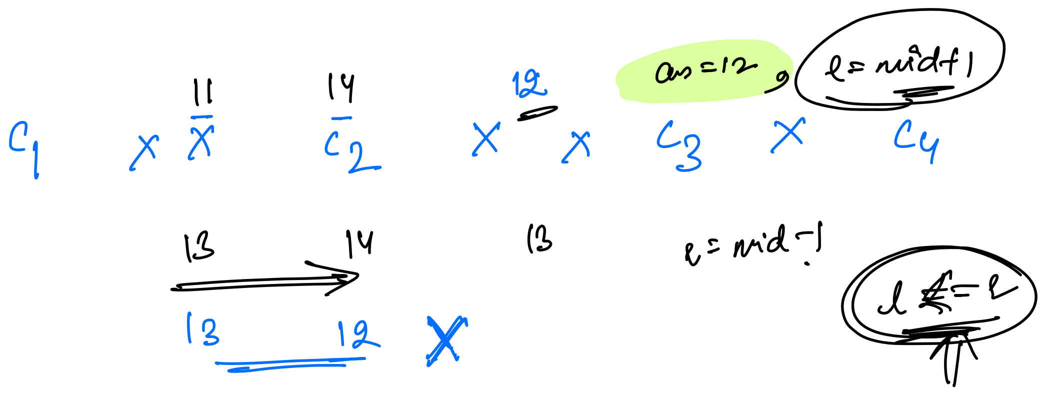
mid+1 ? mid+2? mid+3?

0	1	2	3	4	5	6	7	8
2	6	11	14	19	25	30	39	43

arr[n-1]-arr[0]

M=4

l	r	mid	
1	41	21	r = mid-1
C1 X X X X C2 X X X			
1	20	10	ans = 10, l = mid+1
C1 X X C2 X C3 X C4			
11	20	15	l = mid-1
C1 X X X C2 X X C3 . .			



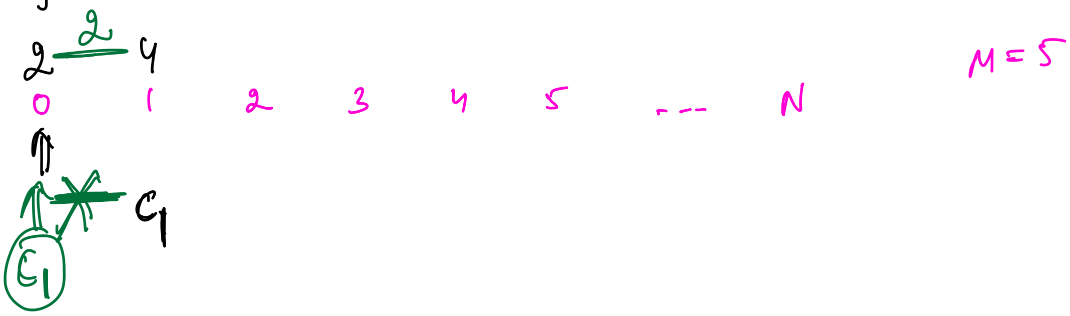
```
~~arr[0]~~
int l= 1 , r= arr[n-1] - arr[0];

while( l<=r)
{        int  mid = (l+r)/2;

         if( check(  mid ))
         {
                ans = mid
                l= mid+1 ;
         }
         else        r= mid-1;
}
```

log(range) * N

```
bool check( arr, mid, M, n)
{
        int  last_placed= arr[0];
           int  cow =1;
        for( int  i=1; i<n; i++)
        {
              if( arr[i] - last_placed >=mid)
              {        cow++;
                       last_placed=arr[i];
                   if( cow ==M) return true;
```

arr[i]-last_pla
>=mid

```
                }
            }
        eturn false;
    }
```

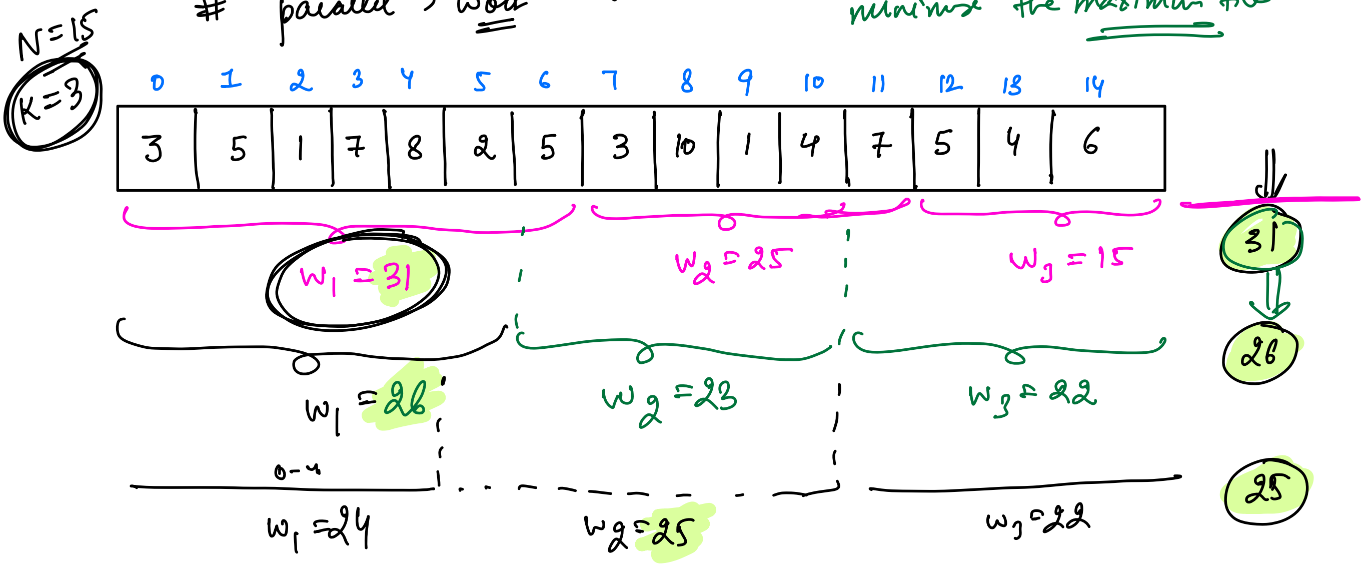


Q Given N Tasks, K workers.

array → A[i] = time taken to complete i^{th} task. Find minimum time to complete all tasks.

- # One task can only be done by 1 person
- # A worker can only do contenuous tasks
- # parallel → work

minimise the maximum time



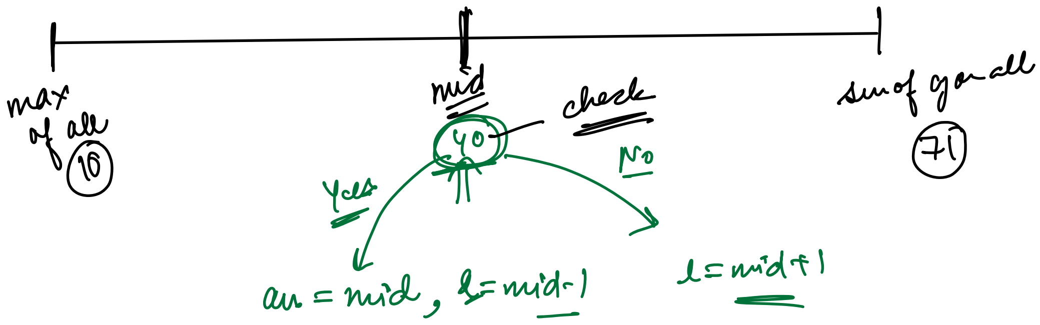
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
3	5	1	7	8	2	5	3	10	1	4	7	5	4	6

target = time to complete the task

search space = l — h

l ↓ max of all

h ↓ sum of your array



0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
3	5	1	7	8	2	5	3	10	1	4	7	5	4	6

k=3

w_1 w_2

l	r	mid
10	71	40
10	39	24
25	39	32
25	31	28
25	27	26

→ if I can complete 15 tasks in 40 min

→ l= mid+1 with 3 work

→ r= mid

```
bool check ( int arr[], int mid, int k, int n)
{
    int worker =1;
    int current_time = arr[0];
    for( int i=1; i<n; i++)        size
    {
        if( current_time + arr[i] > mid)
        {
            worker++;
            current_time = arr[i];
        }
        else {
            current_time += arr[i];
        }
    }
             req
    if( worker <= k)   return true;
    else  return false.          actual reson
}
```

S.C: O(1)

T.C:

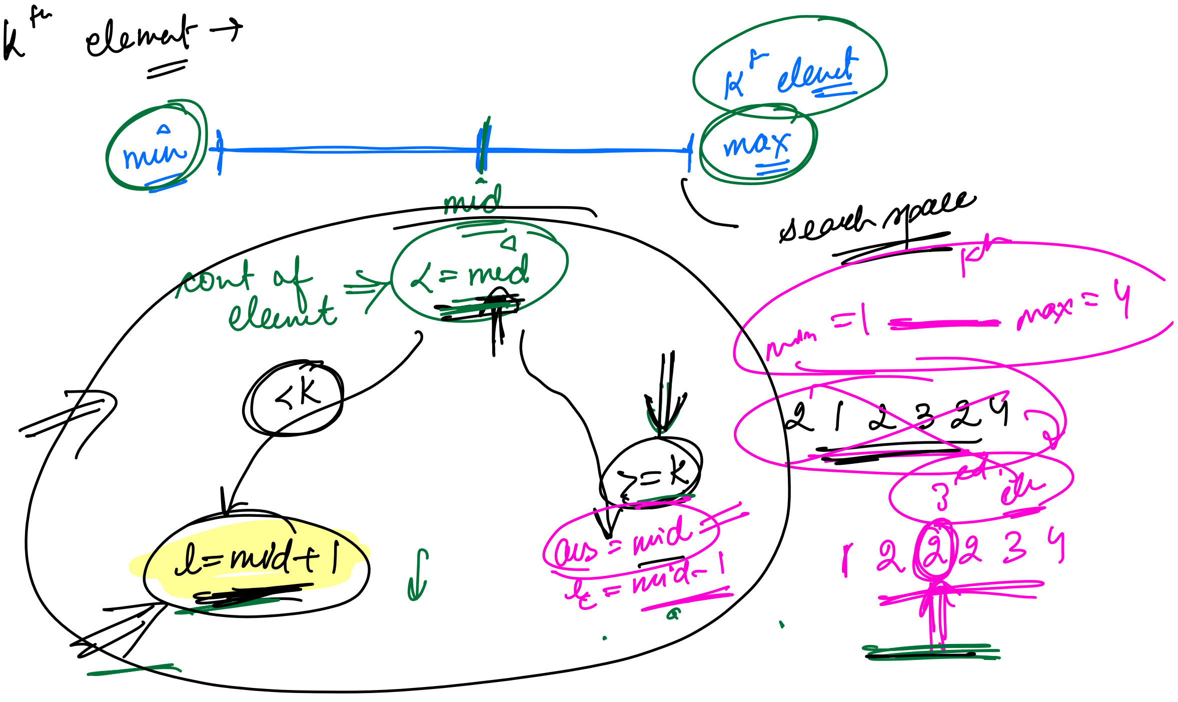
{log(range)×N}

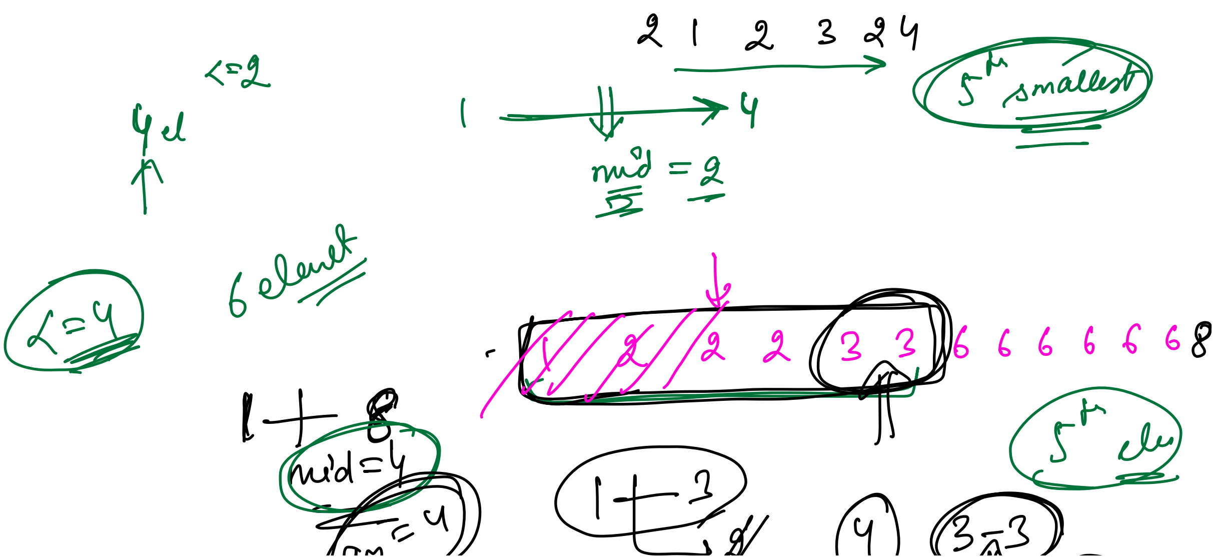
k^th price ≈ k^th smallest number → extra space ✗
→ modify ✗

Doubt

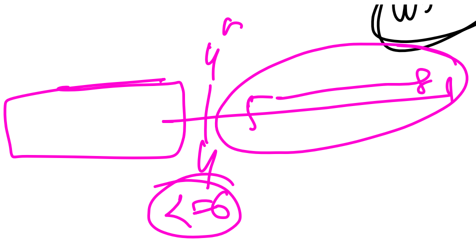
sort
then arr[k-1]

k^th element →





6



1 2 2 3 3 4 5 . . .

1 2 2 4 4 4

median of a matrix   row are sorted

Nth ele in 2 sorted arry

Median of 2 sorted ary

extra

NlogM

1	3	7	9	16
2	5	8	9	10
1	4	8	6	10
12	14	20	24	26
7	8	9	10	11

median → sorted order → N/2 th ele

$\frac{N*M}{2}$ ← K th small

25

$1 \le \_\_\_ \le 10^9$

≤ = mid

l = mid+1

ans = mid
r = mid-1