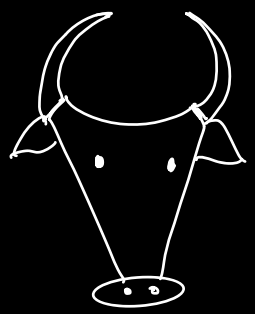
Google Aggresive Cows



Cow / Bull

(Maximize the min possible val)

Given

① A sorted array of true integers having the positions of rooms where we can keep the cows, (A[N])

② K → count of cows

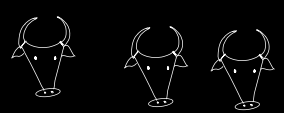
K <= N

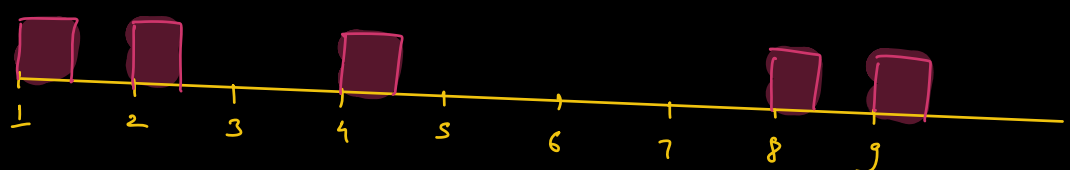
Return the max val of minimum possible distance b/w any two cows.

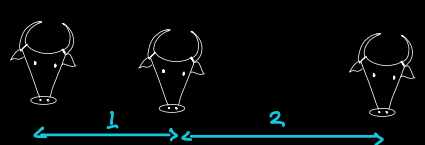
(Maximize the distance b/w the closest cows)

A : 1, 2, 4, 8, 9 ←

K = 3







Brute Force

Try all possible combinations

No of ways to place K cows in N positions

$= {}^{N}C_{K} \quad \left(\frac{N!}{K!\,(N-K)!}\right)$

→ Iterate over all $^{N}C_{K}$ combinations

$O(N) \leftarrow O(K)$ ← keep updating the minDist

↳ dist b/w the closest cows.

TC : $O(N!)$

$\sqrt{N} \longrightarrow$ $1 \times 1 = N$?

$2 \times 2 = N$?

$3 \times 3 = N$?

⋮

target : max i

such that $i \times i <= N$

100

$1 \times 1 = 1$ ✓

$2 \times 2 = 4$ ✓

$3 \times 3 = 9$ ✓

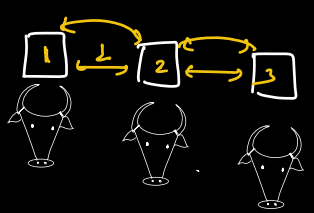
⋮

$9 \times 9 = 81$ ✓

$10 \times 10 = 100$ ✗

Target ⟶ Distance b/w the closest cows.

ans_{max}	$= A[N-1] - A[0]$
ans_{min}	$= 1$ ✓



linear

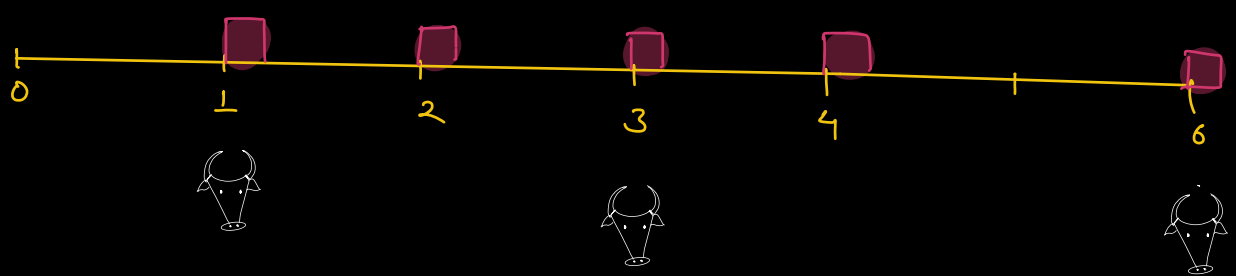
```
for (d = ans_max ; d >= ans_min ; d--) {
    // Check if we can place K cows maintain a
       min distance of d
    if (check(d, A, K)) {  ------> O(N)
        ret d;
    }
}
```

TC: O(R N)

R: Range of ans [1, A[i]_max]

A : 1, 2, 3, 4, 6 (r at 1, l at 6)

K = 3

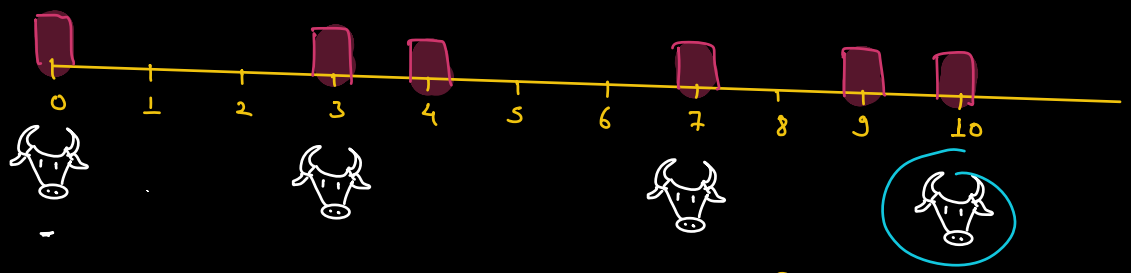


$ans_{max} = 5$

$ans_{min} = 1$

d	Check(d, A, 3)
5	✗
4	✗
3	✗
2	✓

A : 0, 3, 4, 7, 9, 10 K = 4



ans max = 10
ans min = 1

```
boolean check(d, A[], K){
// Returns true if it is possible to place
   K cows maintaining the min dis of at least d.
   (dis b/w any two adjacent cows >= d)

   int prevPos = A[0];
   int cowsPlaced = 1;

   for(i = 1; i < N; i++){
      if(A[i] - prevPos >= d){
         cowsPlaced++;
         PrevPos = A[i];
      }
      if(cowsPlaced == K)
         ret true;
   }
   ret false;
}
```

dis to be checked ↑ , No of cows ↑

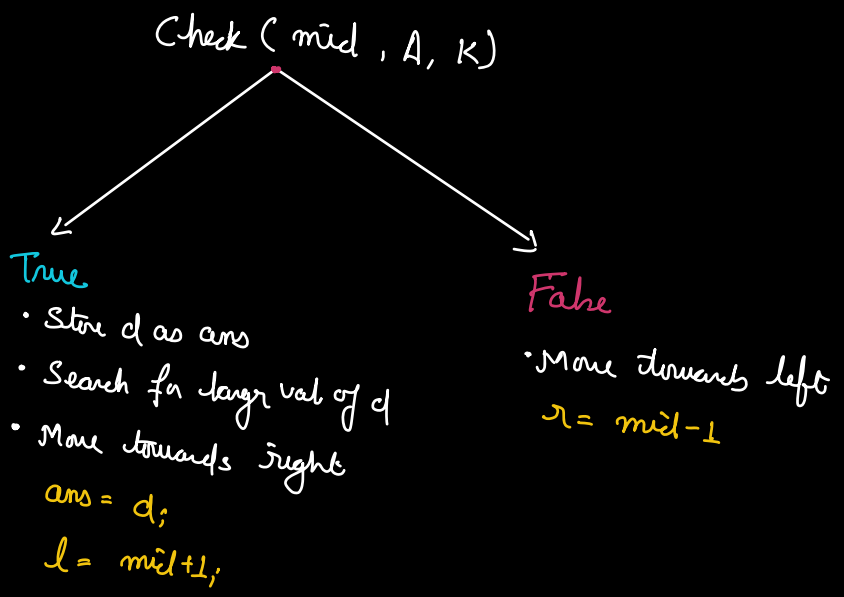
d	Check(d, A, 4)
→ 10	X
9	X
8	X
7	X
6	X
5	X
4	X
3	✓

A[i] - PrevPos >= d

TC : O(N)

target : Distance b/w the closest cows.

Search Space : $[1, A[i]_{max}]$



TC : $O(N \log R)$ — R: Range of the ans.

SC : $O(1)$

Break till 10:40 p

Google (Minimize the max)

Q Given

- N tasks & K workers
- An array A[] of size N

  A[i] ⟶ time required to complete the i^{th} task

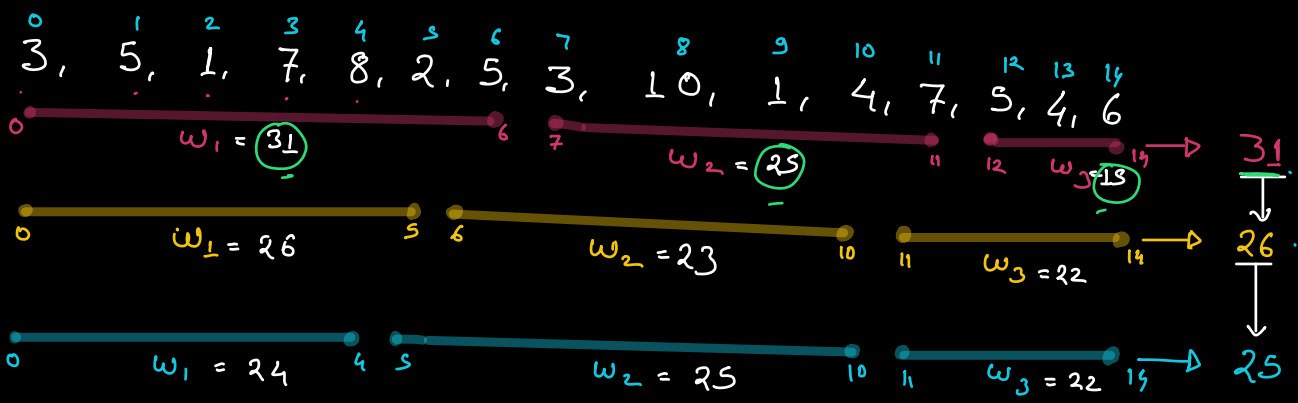
→ One task can only be performed by one worker.

→ A worker can only perform tasks which are contigous to each other

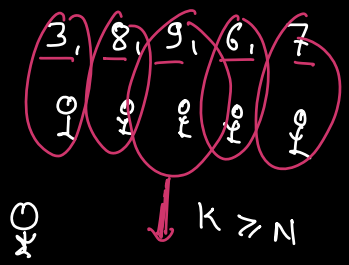
→ All workers can do their tasks in parallel.

Find the minimum amount of time required to complete all the tasks by the worker team.

K = 3



ans_{min}	=	max of A[]
ans_{max}	=	Sum of A[]



linear

```
for( time = ans_min ; time <= ans_max ; time ++) {
    if( check ( time, A, K)) {
        ret time ;
    }
}
```

TC : O(RN)

Target : time

Search Space : [A_{max} A_{sum}]

check (mid, A, K)

TC : O(N log R)

SC : O(1)

True
- Save mid as ans
- Move towards left to find a smaller time.

r = mid - 1 ;

False
- Move towards right & try a larger val

l = mid + 1

Check (time , Arr , k) { } →