07/02/2026
Saturday

Aggressive cows (min dist between 2 cows) is max

arr[] = [0 3 4 7 10 9] cows = 4

these are all stalls (stops)

first sort these

0 3 4 7 9 10

first C_1 $\xleftarrow{3}$ C_2 $\xleftrightarrow{1}$ C_3 $\xleftrightarrow{3}$ C_4 min 1

Here all cows but we want max(min)

* U needs to place all cows in stalls in a way to get max(min)

0 3 4 7 9 10

$C_1 \xleftarrow{3} C_2 \xleftrightarrow{4} C_3 \xleftrightarrow{2} C_4$ min is 2

but we want max

0 3 4 7 9 10

$C_1 \xleftarrow{3} C_2 \xleftrightarrow{4} C_3 \xleftrightarrow{3} C_4$ min is 3

but we still increase to find min

0 3 4 7 9 10

$C_1 \xleftarrow{4} C_2 \xleftrightarrow{5} C_3$

not possible

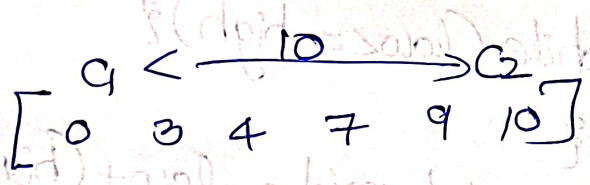
So answer is 3

suppose take

arr[] = [0 3 4 7 9 10] cows = 2

to get max(min)

I will place 1 cow in 1st stall and another cow in last stall



function to check whether we can place the cows (such that we need to get max) all ; imp

```
boolean function checkPlaceCows(arr, dist, cows)
{
    int countcows = 1;
    last = arr[0];
    for(int i = 1; i < arr.length; i++)
    {
        if (arr[i] - last >= dist)
        {
            countcows++;
            last = arr[i]
        }
    }
    if(Countcows >= cows)
        return true
    else
        return false;
}
```

```
public int solve(int[] a, int cows)
{
    int low=1;
    int high= max-min;   // max in array a, min in array a
    while(low<=high){
    {
        int mid= low+(high-low)/2
        if(canWePlaceCows(a,mid,cows))
        {
            low=mid+1;
        }
        else{
            high=mid-1;
        }
    }
    return low;
}
```

✓ ✓ ✓ X X X X X X X

low 1 2 3 4 5 6 7 8 9 10 high

high low

at Last high reaches the possible one