Painter's Partition

Aggressive Cows

Holiday 11 - 15th Nov

Next Mon 13th

Next Wed 15th

1. Given N tasks, K workers. Time taken to complete i^{th} task → A[i].
Find minimum time to complete all tasks.

- One task can be only done by 1 person.
- A worker can only do continuous tasks.
- Workers work parallely

A = < 10, 10, 10, 10 > K = 2

ans → 20

No	Time
1	40
2	30

10 → P_1 ; 10, 10, 10 → P_2

2	20

10, 10 → P_1 ; 10, 10 → P_2

A = < 10, 20, 30, 40 > K = 2

ans = 60

No	Tim
1	100
2	90

10 → P_1 ; 20, 30, 40 → P_2

70 | 10, 20 (P_1) | 30, 40 (P_2)

60 | 10, 20, 30 (P_1) | 40 (P_2)

$A = \langle 10, 20, 30, 40 \rangle \quad K = 2$

2 ppl available : 100 amt of work

Idea 1 : Divide work equally among all painters

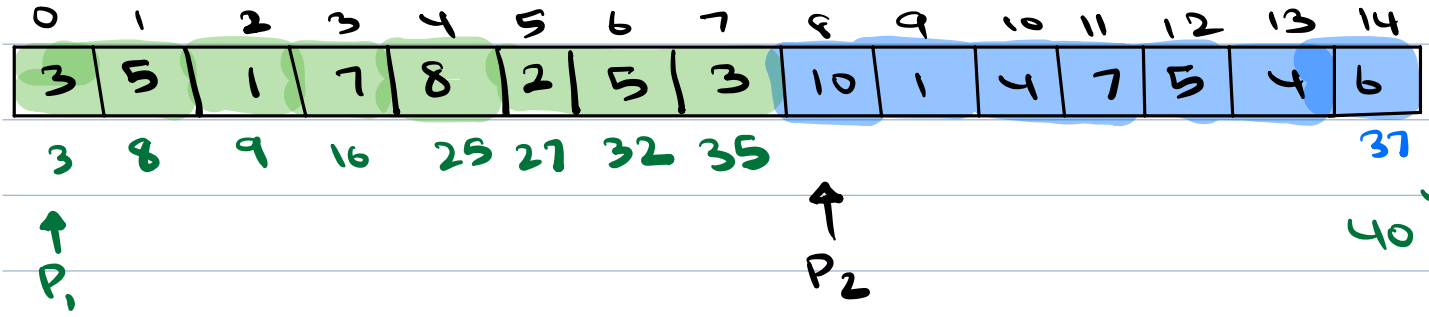
$$\text{Work} = \frac{\text{Total work}}{K}$$

$$\text{Work} = \frac{100}{2} = 50$$

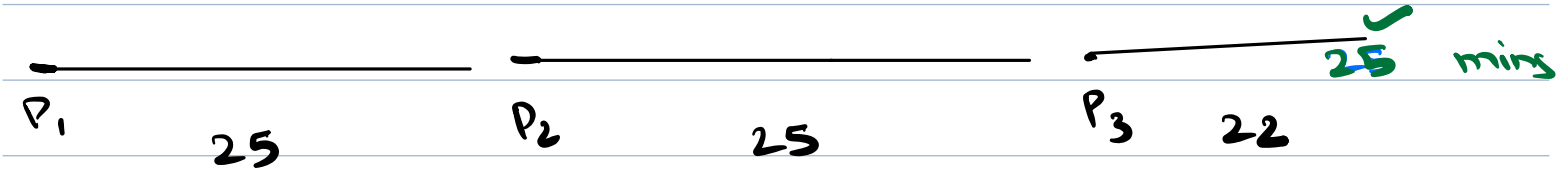
$A = \langle 10, 20, 30, 40 \rangle \quad K = 3$

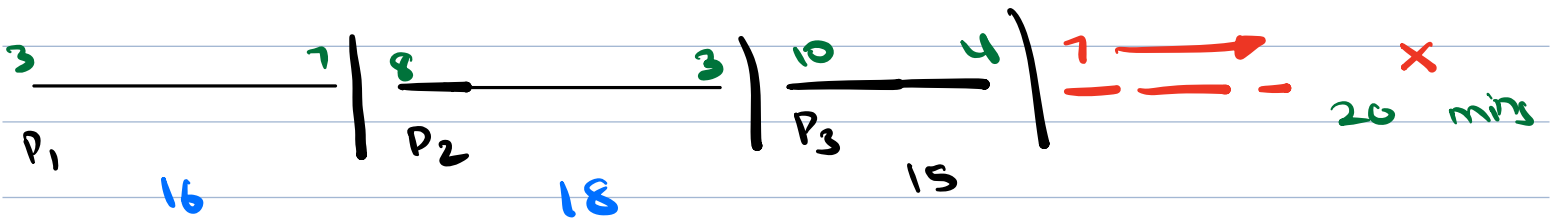
$\text{Work} = 100/3 = 33.33$

N = 15 K = 3

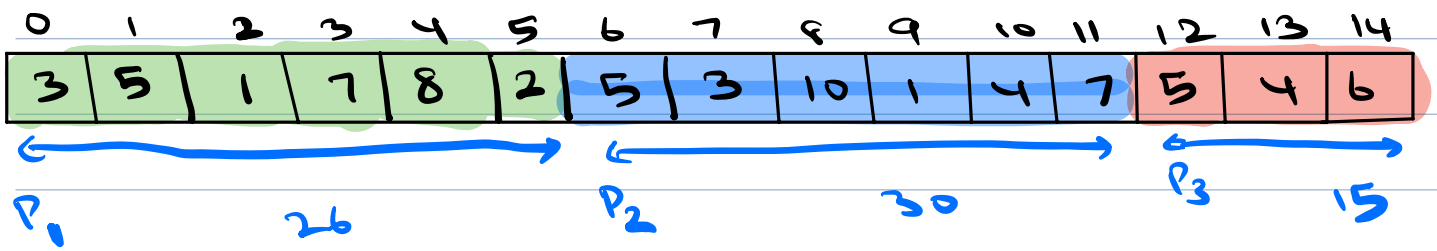


40 mins ✓



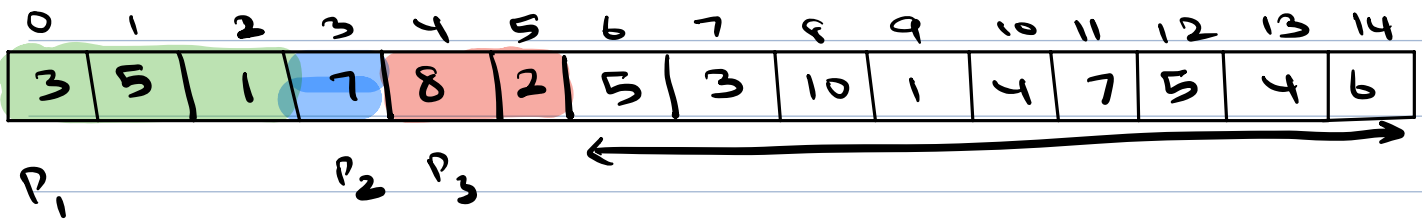


K = 3



Can we do all task in 30 mins ? (≤ 30)

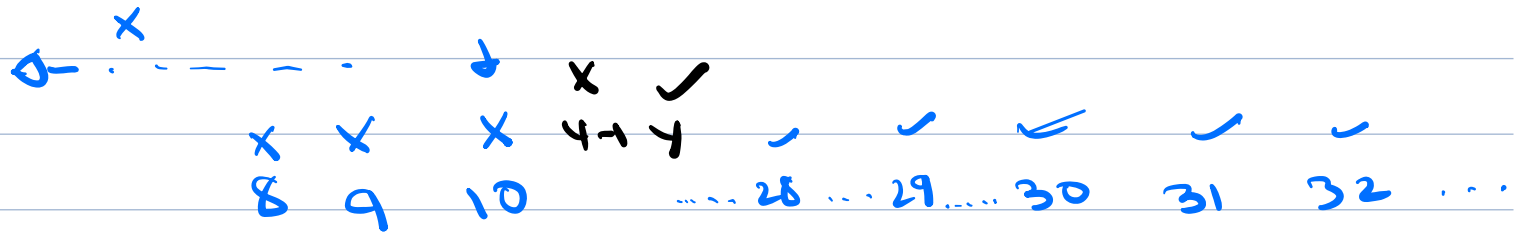
30 ✓ 31 ✓ 32 ✓ 33 ✓ 34 ✓ ...



Can we do all task in 10 mins ? (≤ 10)

X X X X

8 9 10



BS on time

Target - Min time to paint all boards

Search space -

s → max of boards[]
lowest time to paint all board
(1 board → each painter)

e → sum of boards[]
highest time to paint all boards
(1 painter has to paint all boards)

P_1 P_2 P_3 P_4 P_5 . . . .

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

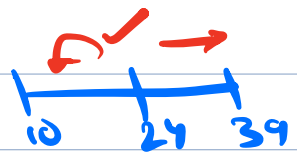
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
3	5	1	7	8	2	5	3	10	1	4	7	5	4	6

N = 15 K = 4

	s	e	mid	
①	10	71	40	ans = 40 left

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

N = 15 34 37

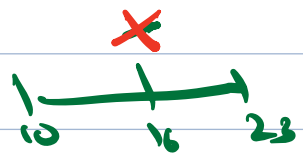


②	10	39	24	ans = 24 left

K = 4

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

P_1 24 P_2 21 P_3 20 P_4 6

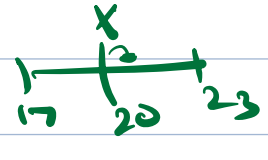


③	10	23	16	right

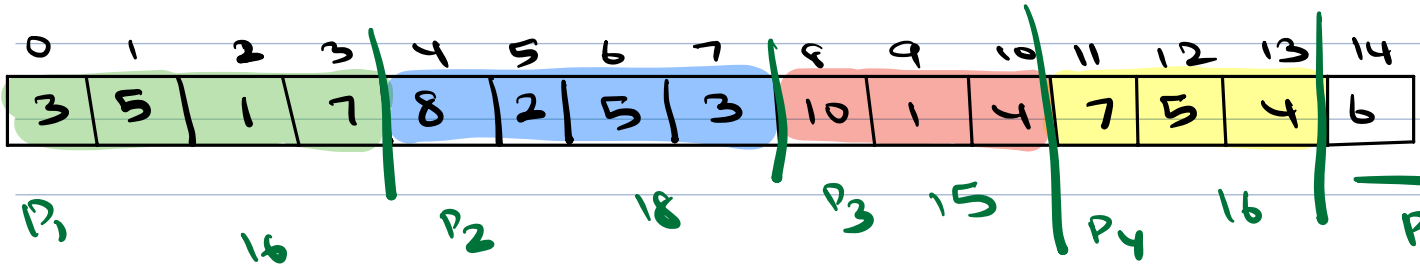
K = 4

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

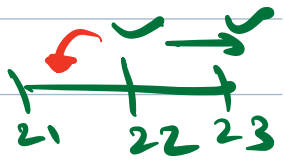
P_1 16 P_2 15 P_3 13 P_4 16 P_5 X

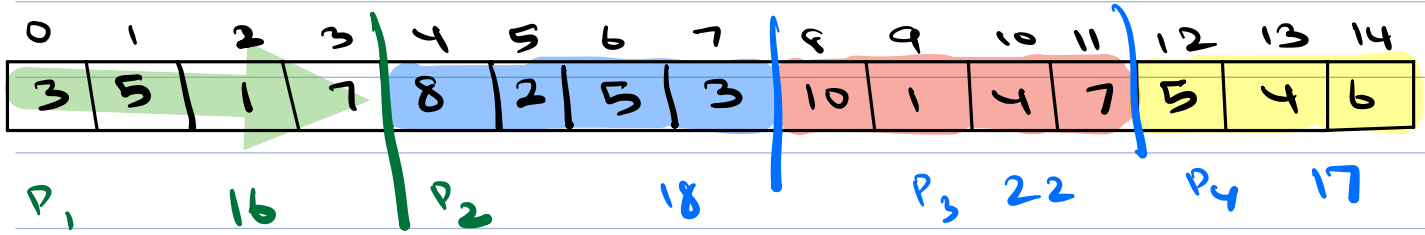


④	17	23	20	X	right

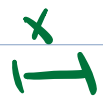


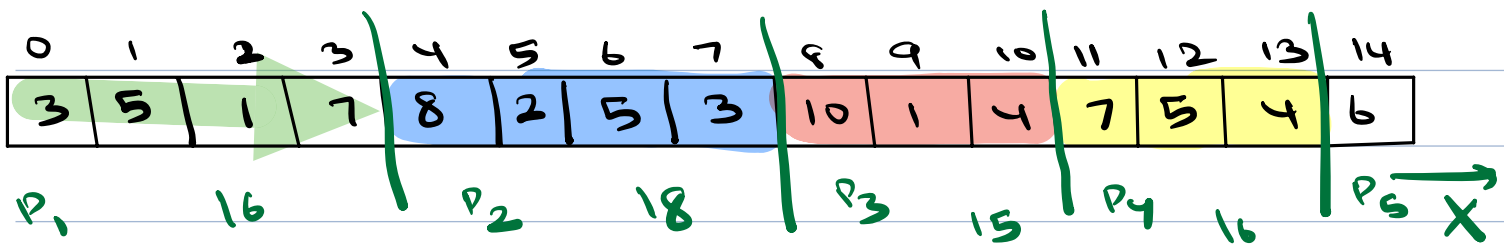
(5) 21 23 22 ans = 22 left



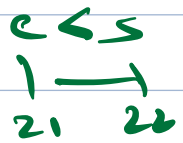


(6) 21 21 21 X right





(7) 22 21 Break



ans = 22

```
int time(int boards[], int n, int K){
    // BS on time
    s = maxEle(arr)
    e = sum(arr)

    while(s <= e){
        mid = (s+e)/2

        // checking if all jobs can be completed in ≤ mid time
        if(check(mid, boards, n, K)){
            ans = mid
            e = mid-1      // left
        }
        else{
            s = mid+1      // right
        }
    }
    return ans
}
```

```
bool check (int m, int boards[], int n,
            int k){
    cur = 1      work = 0
    for (i=0; i<n; i++){
        if (work + board[i] <= m){
            work = work + board[i]
        }
        else {
            cur++      work = board[i]
        }
    }
    if (cur > k)
        return false
    else
        return true
}
```

k = 4

TC → log (range) × time taken for feasibility check

TC = O(log (sum(dc) - max(dc) × N)

check fn

Break → 10:30

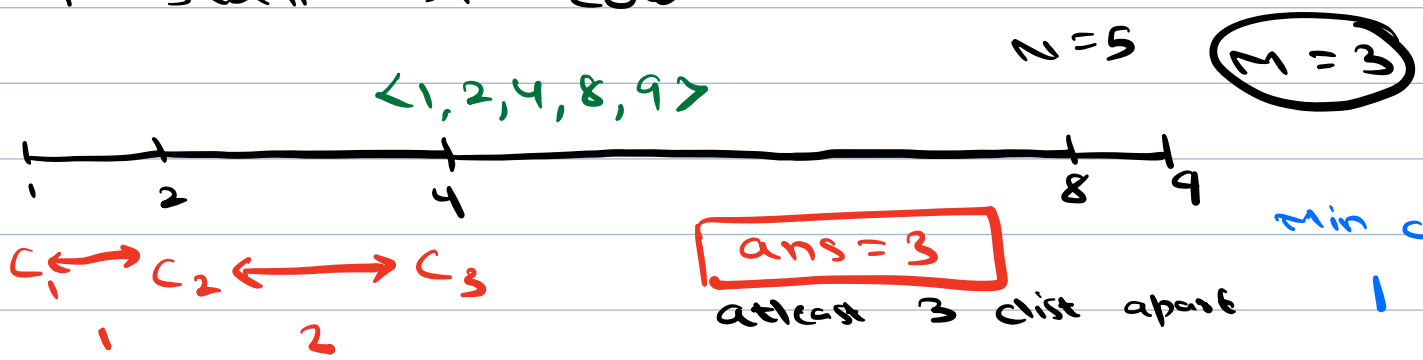
2. Given N stalls, and M cows. $N \geq M$

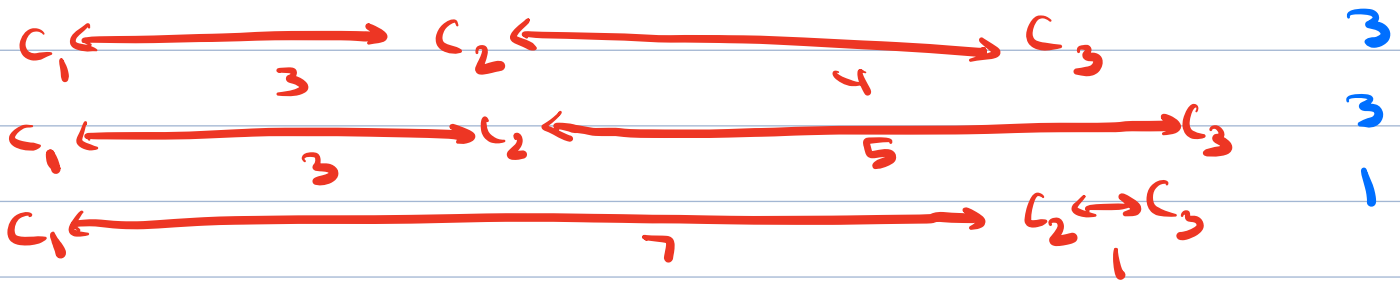
Position of each stall is given in A[N] in asc order.

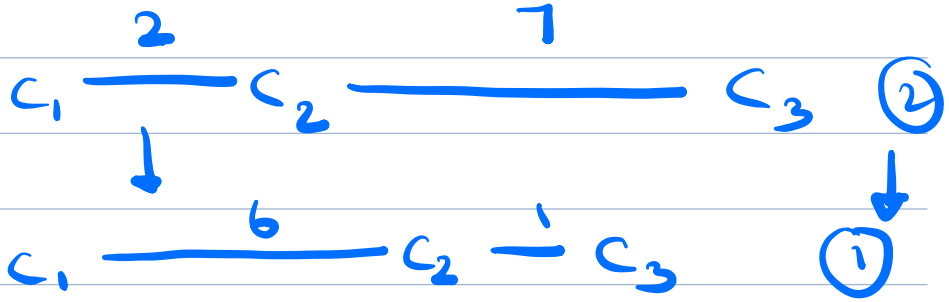
Maximise the closest distance b/w cows.

min

1 stall → 1 cow







N = 9, M = 4

ans = 12

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

Min dist

$C_1 \xleftrightarrow{4} C_2 \xleftrightarrow{5} C_3 \xleftrightarrow{3} C_4$ → 3

$C_1 \xleftrightarrow{9} C_2 \xleftrightarrow{8} C_3 \xleftrightarrow{11} C_4$ → 8

$C_1 \xleftrightarrow{12} C_2 \xleftrightarrow{16} C_3 \xleftrightarrow{13} C_4$ → 12

✗ $C_1 \xleftrightarrow{17} C_2 \xleftrightarrow{20} C_3 \xleftrightarrow{<13} C_4$ (X) → 13

M = 3

min dist

cnt = 2 : $C_1 \xleftrightarrow{8} C_2$ → 8

cnt = 3 : 1 — 9, 5, $C_1 \xleftarrow{4} \xrightarrow{4} C_2$, C_3 → 4

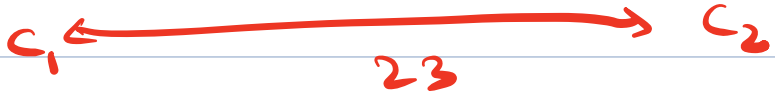
Stall at 5 ?

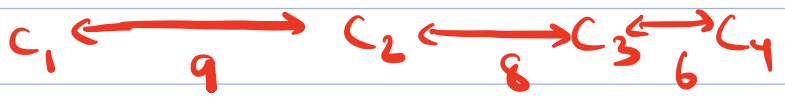
BF → $\frac{\text{Total dist}}{\text{cows}}$ ✗

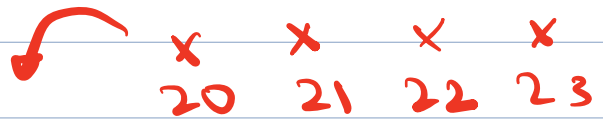
BS → dist

N = 9 M = 4

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

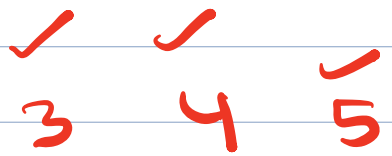






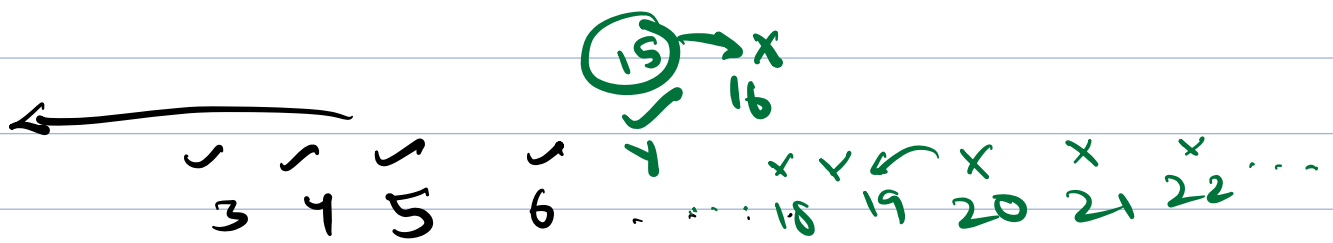
Check if we can place all cows atleast 20 dist apart?

dist ≥ 20



Check if we can place all cows atleast 5 dist apart?

dist ≥ 5



Target → Min dist b/w any 2 cows should be max

Max closest dist

Search space →

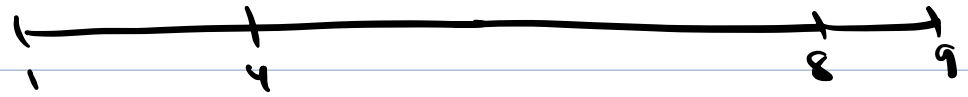
S → Min dist b/w any 2 adjacent stalls

e → Max dist of closest cows

A[N-1] - A[0]

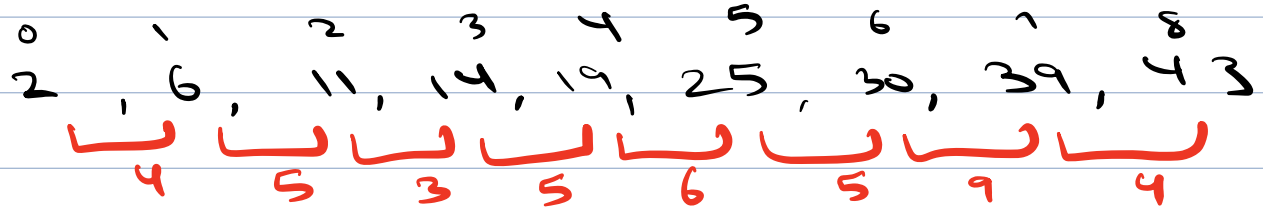
N = 4

S = 1

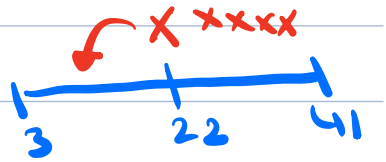


e =

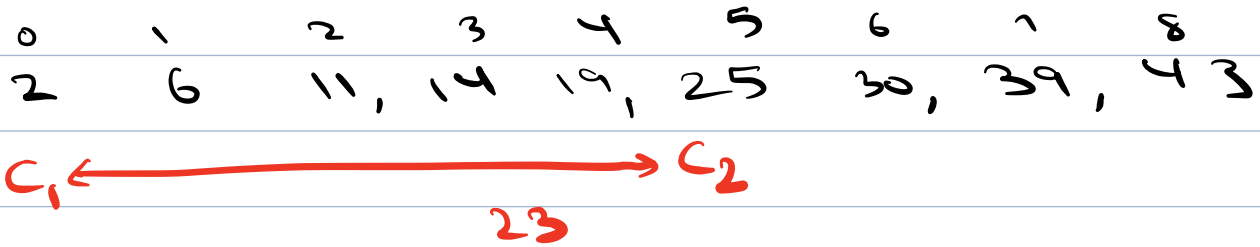
N = 9 M = 4



	s	e	mid	
①	3	41	22	left

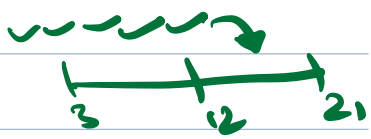


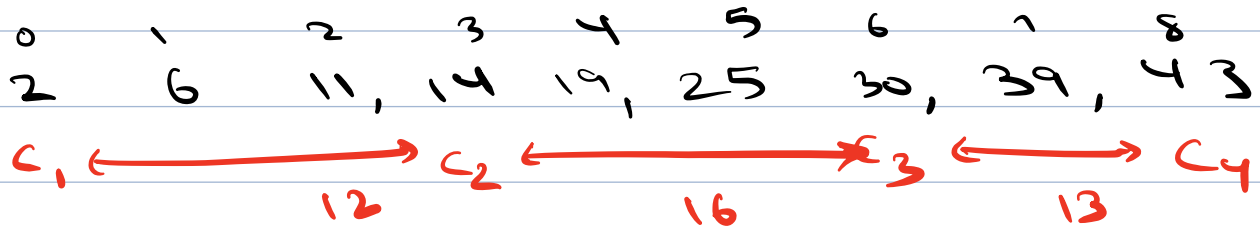
can you place cows atleast 22 dist apart?



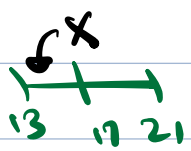
M = 4

②	3	21	12	ans = 12	right





③	13	21	17	X	left



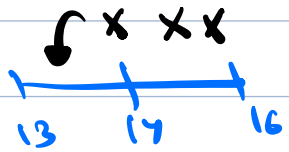
0 1 2 3 4 5 6 7 8

2 6 11, 14 19, 25 30, 39, 43

C_1 ⟷ 17 ⟷ C_2 ⟷ 20 ⟷ C_3

Not able to place 4th cow

(4) 13 16 14 left ✗ dist ≥ 14

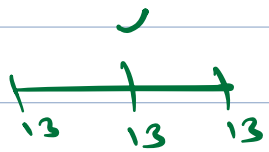


0 1 2 3 4 5 6 7 8

2 6 11, 14 19, 25 30, 39, 43

C_1 ⟷ 17 ⟷ C_2 ⟷ 20 ⟷ C_3

(5) 13 13 13 ✗ left



0 1 2 3 4 5 6 7 8

2 6 11, 14 19, 25 30, 39, 43

C_1 ⟷ 17 ⟷ C_2 ⟷ 20 ⟷ C_3

(6) 13 12 break

ans = 12

Nlog N ⟨ Sort stalls

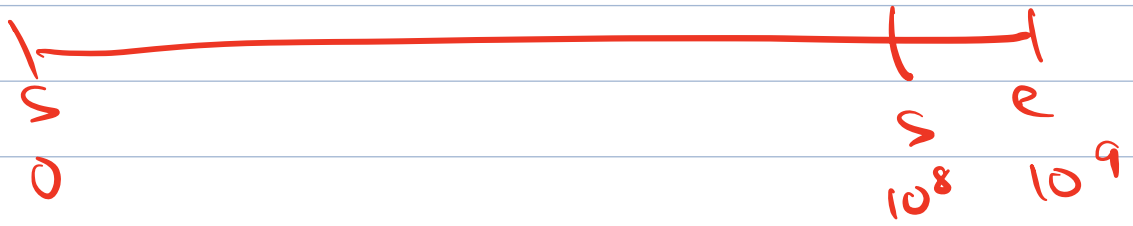
```
int maxdist(int dist[], int N, int M) {
    s = min(dist[i+1] - dist[i]) ∀ i
    e = dist[N-1] - dist[0]

    while (s <= e) {
        mid = (s+e)/2 // s + (e-s)/2
        // can we place cows atleast mid dist apart?
        if (check(mid, dist, n, m)) {
            ans = mid
            s = mid+1        // right
        }
        else {
            e = mid-1        // left
        }
    }
    return ans
}
```

```
bool check (int mid, int dist[], int n, int m){
    cur = 1        position = dist[0]
    for (i=1 ; i < n ; i++) {
        if (dist[i] - position >= mid) {
            cur++
            position = dist[i]
            if ( cur == m)
                return true
        }
    }
    return false
}
```

TC : $O\left(\log\left(\begin{matrix} \text{max} \\ \text{diff} \\ \text{b/w} \\ \text{dc} \end{matrix} - \begin{matrix} \text{diff} \\ \text{b/w any} \\ \text{2} \\ \text{adjacent} \\ \text{c(.)} \end{matrix} \right) \times N \right)$

$\log(s-e)$ $+ N + N \log N$



int s
int e

int mid = $\frac{s+e}{2}$ $\quad \frac{10^8 + 10^9}{2}$

overflow

mid = $s + \frac{(e-s)}{2}$

$s + \frac{e}{2} - \frac{s}{2}$

$\frac{s}{2} + \frac{e}{2} = \frac{(s+e)}{2}$

---

paint all boards)

place all cows

Task to do

1-2 constraint

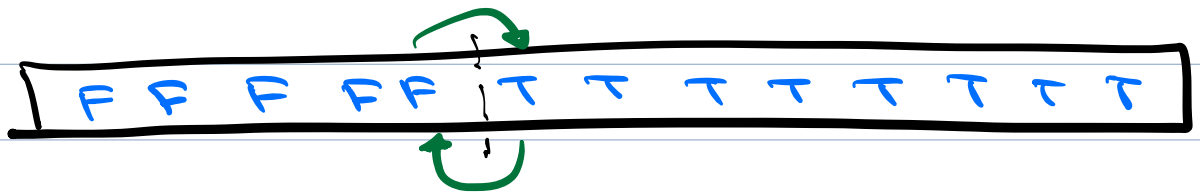
Minimize or Maximize

① K painters

M cows

② BS → Monotonic search space

Monotonicity - come up with a condn where part of the search satisfies condn and other part doesn't satisfy condn, we can use BS to find pivot point.



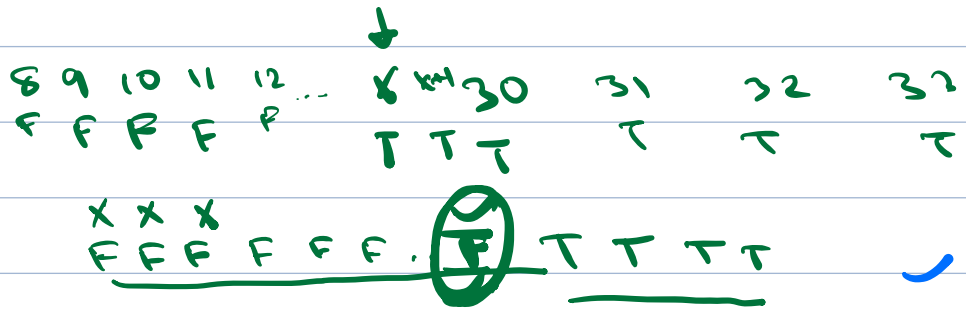
T T T T T T T F F F F

① 1 2 4 6 | 8 10 20 30 50
T T T T | F F F F F

arr[i] ≤ 6

② 4 8 2 1 6 30 20 10 50
T F T T T F F F F

arr[i] ≤ 6


8 9 10 11 12 ... k k+1 30 31 32 37
F F F F F T T T T T T T
X X X
F F F F F F F T T T T T