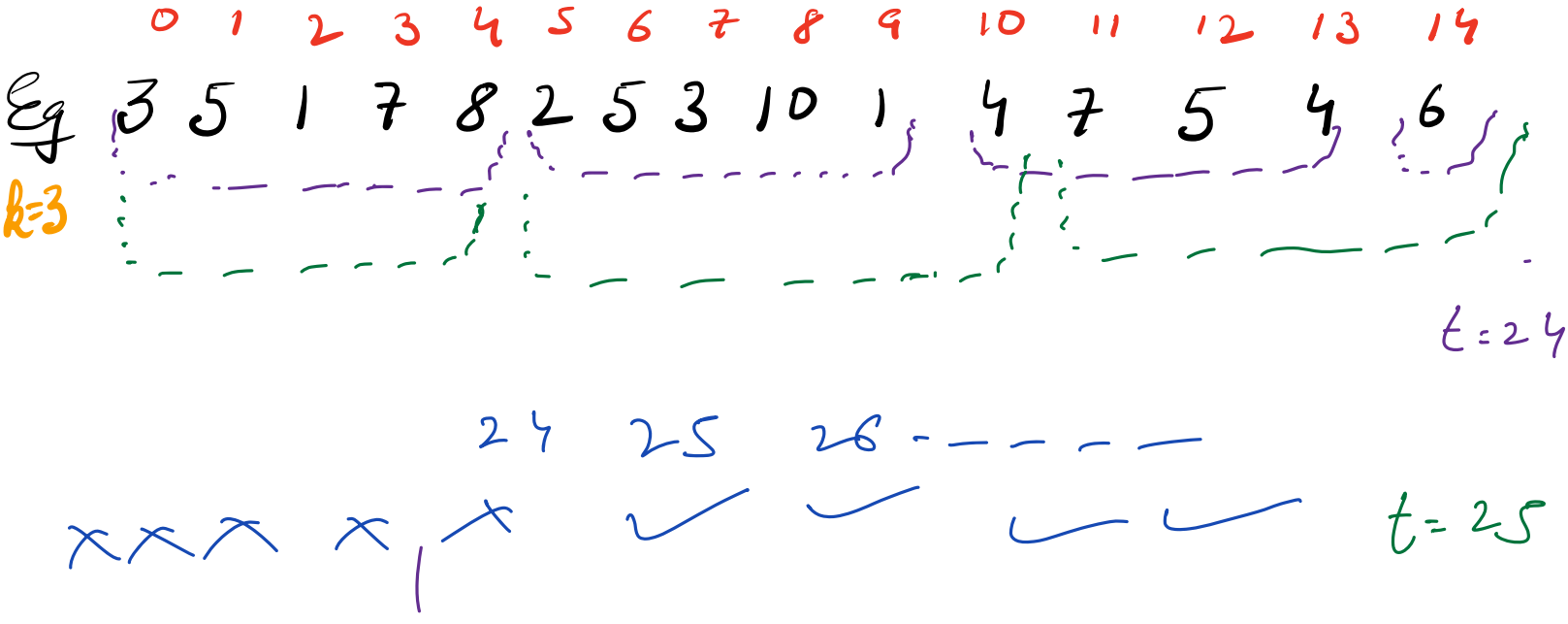
01) Given N tasks, K workers & time taken for each task, find min time to complete all tasks.

Note: Single worker can only do continuous set of tasks

Note: All workers start at same time = 0

Note: A task is only assigned to 1 worker



Idea: Binary Search

Can you finish all tasks in 30 min?

Yes

Can you finish all tasks in 10 mins?

NO

8 9 10 - - - - - - - 30 31 32

F F F T T T

Search: Target : Min time to finish all taks with k workers

l = max(arr) h = sum(arr)

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 = 4

l	h	m
10	71	40
10	39	24
10	23	16
17	23	20
21	23	22
21	21	21
22	21	STOP!!!

ans = 40

ans = 24

ans = 22

Code

```
int workers (int time[], int N, int K){
    l = max(all)   h = sum(all)
    while (l <= h){
        m = (l+h)/2
        if ( check( time, N, K, m)){
            ans = m ,    h = m-1
        }
        else {
            l = m+1
        }
    }
    return ans
}
```

TC: $O(n \log(sum-max+1))$

SC: $O(1)$

max sum

1 mid → O(n)

(sum - max + 1) ⇒ n × log(sum - max + 1)

```
bool check (int m, int time[], int N, int k){
    s=0        c=1
    for (i=0; i<N; i++){
        s = s + time[i]
        if (s > m){
            c++
            s = time[i]
            if (c > k) return false
        }
    }
    return true
}
```

TC: O(n)

SC: O(1)

3	5	7	4

s=7 s=4 m=10

Q2 Given N cows & M stalls, all stalls are on x-axis at diff location. Place all N cows in such a way such that min distance b/w any 2 cows is maximised

Note 1 One stall has only 1 cow

Note 2 All cows have to be placed

	0	1	2	3	4
Eg	1	2	4	8	9
	C_1		C_2		C_3

cows = 3

ans = dist = 3

	0	1	2	3	4	5	6	7	8
Eg	2	6	11	14	19	25	30	39	43
	C_1			C_2			C_3		C_4

cows = 4

ans = 12

Idea : Binary Search

Can you place cows at atleast 20 distance

NO

Can you place cows at atleast 5 distance

Yes

3 4 5 _ _ _ _ _ _ _ 20 21 22

T T T T T T T F F F F F F F

Search

Target : distance b/w 2 cows

l : Min distance b/w 2 stalls

h : ar[n-1] - ar[0]

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

l	h	m	
3	41	22	×
3	21	12	✓
13	21	17	×
13	16	14	×
13	13	13	×
13	12	STOP!!!	

cows = 4

ans = 12

Code

```
int moo(int dist[], int N, int cows){
    l = min adjacent distance
    h = dist[n-1] - dist[0]
    while(l <= h){
        m = (l+h)/2
        if(check(m, dist, N, cows)){
            ans = m        l = m+1
        }
        else    h = m-1
    }
    return ans
}
```

TC:

SC: O(1)

$n \times \log(dist[n-1] - dist[0] - min\_dist + 1)$

```
bool check(int m, int dist[], int N, int cows){
    last_cow = dist[0]
  count = 1
  for(i=1; i<n; i++){
    if(dist[i]-last_cow >= m){
      c++, last_cow = dist[i]
      if(c==cows) return true
    }
  }
  return false
}
```

TC: O(n)

SC: O(1)

<done>

Monotonicity