Todays Content

1. SQRT ✓
2. Painters Partition.

1Q: Given the find SQRT(N)

Find greatest i such that i*i <= N

Sqrt(25) : return 5

Sqrt(37) : 5*5 <= 37, 6*6 <= 37, 7*7 <= 37 return 6.

N = 30

i	i*i <= N	ans =
1	1*1 <= 30	ans = 1 ✓
2	2*2 <= 30	ans = 2 ✓
3	3*3 <= 30	ans = 3 ✓
4	4*4 <= 30	ans = 4 ✓
5	5*5 <= 30	ans = 5 ✓
6	6*6 <= 30	return ans = 5

```
int sqrt(int N){ TC: O(√N)  SC: O(1)
    int i=1, ans=0;          → i^2 <= N ⇒ i <= √N
    while( i*i <= N ){
        ans = i;   # update ans
        i++;       # look for better as
    }
    return ans;
}
```

Idea:

1. Target: Greatest i, with i*i <= N.

2. Search Space: Space where we search for target, it can be imaginary

   S1 : [1 .. √N] → we don't know √N   l h

   S2 : [1 .. N/2] , if N=1, search space [1..0] *

   S3 : [1 .. N], if N=1, search spa [1..1] ✓

   Note: In your search space it should contain target for every ele.

3. Discard

   N=30   Search Space [1..N] # Imaginary: No need to store

l	h	m		l	m	h	
1	30	15	m*m <= 30	{ 1	15	16 17 .. 30 }	h = m-1;
1	14	7	m*m <= 30	{ 1	7	8 9 ... 14 }	h = m-1;
1	6	3	m*m <= 30	{ 1	3	6 }	ans = 3, l = m+1
4	6	5	m*m <= 30	{ 4	5	6 }	ans = 5, l = m+1
6	6	6	m*m <= 30	{ 6	6	6 }	h = m-1;
6	5	return ans = 5.					

Constraints:

$N = 1 \quad\underline{\hspace{3em}}\quad 10^{18}$

$1 <= N <= 10^{18}$ → sqrt = $[1 \quad\underline{\hspace{3em}}\quad 10^9]$ # If we take 1..10^9, product of m*m will be in long range

```
int Sqrt(int N){
    int l = 1, h = ~~N~~ 10^9, min(N, 10^9) , ans = 0;        Note: Talk about datatype

    while( l <= h){
        long m = (l+h)/2;
        if( m*m <= N){
            ans = m;
            l = m+1;
        }
        else{ # m*m > N goto left
            h = m-1;
        }
    }
} return ans;
```

Note: #Search Space Size [1..N] : # N Elements

TC: $O(\log_2 N)$ SC: $O(1)$

Q1:

We have to paint N boards of length {C_0 C_1 C_2 .... C_{N-1}}

There are W painters available and each of them takes 1 unit of time to paint 1 unit of board

Calculate & return the minimum time requried to paint all boards

Notes:

1. Two painters cannot share a board to paint, A board cannot be painted partially by 1 painter & partially by another.
2. A painter can only paint contiguous boards
3. All painters start work at same time.

$l = 10$ $h = 71$

Ex:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	
N=15	3	5	1	7	8	2	5	3	10	1	4	7	5	4	6	Time Take
W=3	$W_1=34$								$W_2=15$			$W_3=22$				34.
	$W_1=31$							$W_2=18$				$W_3=22$				31.
	$W_1=24$					$W_2=25$						$W_3=22$				25

Ex2:	0	1	2	3	4	5		
ar[6] =	1	1	1	1	1	101		Time Taken
W=2	$W_1=5$					$W_2=101$	:	101

Idea: Average Time will not work, as seen in abve example

l = smallest val pos

h = greatest val pos.

Search:

Target: Min time taken to pain all boards

SearchSpace: Should be based on time:

{l . . . . . h}

#Smallest time to paint all boards l: Max of ar[]

#greatest time to paint all boards h: Sum of ar[]

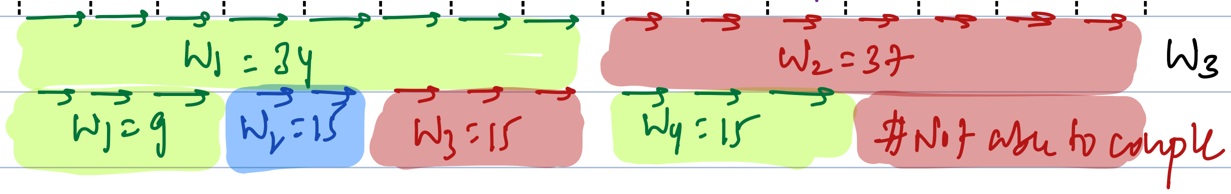
So that for every input ans in search space

Discard ?

N = 15

W = 4

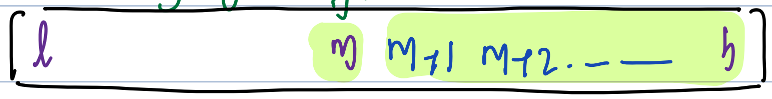
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
3	5	1	7	8	2	5	3	10	1	4	7	5	4	6



l h m

10 71 40 : Check if we can paint all boards <= 40 mins ✓

ans = 40; goto left



If we can do task in m time:

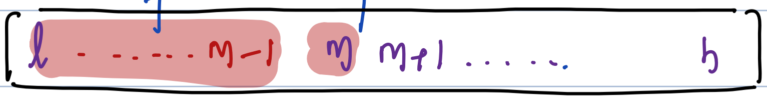
We can do task in m+1, m+2...

update m & goto better ans m left.

#Say m = 15.

m

15: Check if we can paint all boards <= 15 mins



If we cannot do task in m time:

We cannot do task in m-1, m-2. .

goto right

Check function Idea:

W=4	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	
N = 15	3	5	1	7	8	2	5	3	10	1	4	7	5	4	6	
M = 40; S=0	3	8	9	16	24	26	31	34	~~44~~ 10	11	15	22	27	31	37	..
p=1 ⟶								p++	2							p=2 <= 4

W=4	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	
N = 15	3	5	1	7	8	2	5	3	10	1	4	7	5	4	6	
M = 24 ; S = 0	3	8	9	16	24	~~26~~ 2	7	10	20	21	~~25~~ 4	11	16	20	~~26~~ 6	
p = 1 ⟶					p++	2 ⟶				p++	3 ⟶			p++	4	p = 4 <= 4

W = 4	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	
N = 15	3	5	1	7	8	2	5	3	10	1	4	7	5	4	6	
M = 16 S = 0	3	8	9	16	~~24~~ 8	10	15	~~18~~ 3	13	14	~~18~~ 4	11	16	~~20~~ 4	10	
p = 1 ⟶				p++	2 ⟶ p++			3 ⟶ p++			4 ⟶ p++			5		p = 5 > 4

l	h	m: l+h/2	Can we finish tasks in m time	update
10	71	40:	Can we finish tasks in 40m	ans = 40; h = m-1;
10	39	24:	Can we finish tasks in 24m	ans = 24; h = m-1;
10	23	16:	Can we finish tasks in 16m	l = m+1;

```
int minTime( vector<int> b, int N, int W){
    int l = max of b, h = sum of b, ans = 0;
    while( l<=h){
        int m = (l+h)/2;
        if( check(b, m, W)){
            ans = m;
        } h = m-1;
        else{
            l = m+1;
        }
    }
    return ans;
}

bool check(vector<int> &b, int m, int W){
    int p=1, s=0,
    for( int i=0; i<b.size(); i++){
        s = s+b[i];
        if( s>m){ # ith task allocate new persn
            p++;
        } s = b[i];
    }
    return p <= W; # p: persn used & W= worken allocated
}
```