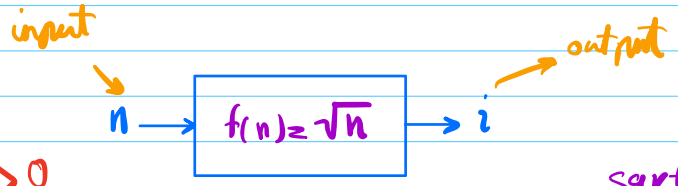
⓪ Intro

topics: ① - Painters
Partition problem

② - Aggressive
Cows!

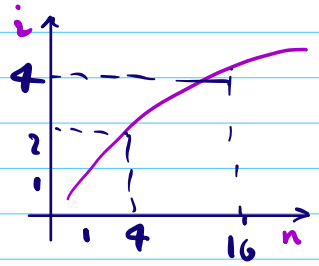
① $i = \sqrt{n}$ →




$n > 0$

$i > 0$

| i | i*i | n |
|---|---|---|
| 1 | 1 | |
| 2 | 4 | |
| 3 | 9 | |
| 4 | 16 | |
| 5 | 25 | 26 |
| 6 | 36 | |
| 7 | 49 | |
| ⋮ | ⋮ | |
| 25 | | |
| 26 | | |




sqrt(26)=?

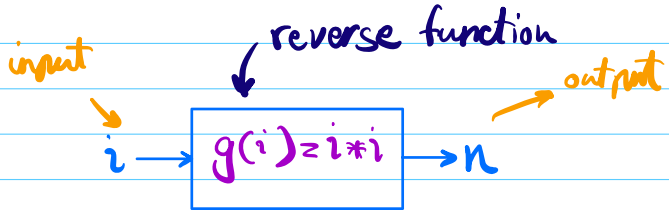
| n | i |
|---|---|
| 1 | 1 |
| 4 | 2 |
| 25 | 5 |
| 26 | 5.099... |
| 36 | 6 |

√26

1 26

⌊sqrt(26)⌋ =5

② $n = i^2$ →




- What is the reverse Function?
- Can we apply this method to any algorithmic problem?
  - Condition
  - practicality and feasibility

P1.1 Painters Partition problem

order of boards are fixed.

wait for examples so question becomes clear

Given n boards with length of each board's

(a) A painter takes t unit of time to paint 1 unit of length.
ex. t = 2h

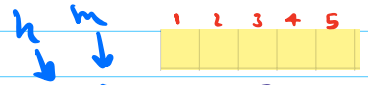
(b) A board can only be painted by exactly 1 painter.

(c) A painter can only paint boards placed next to each other (i.e. cont. segment)

Goal: find min number of painters required to paint all boards in X unit of time. ret -1 if not possible.

ex
10 day
10 hours

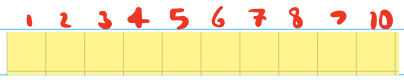
1 board len 5




h m

2x5 = 10h

1 board len 10



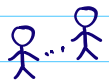

2x10 = 20h

ex1 t = 2h (hours)
X = 15h ans = -1

t = 2h
X = 20h P = 2 ← ans

20hours

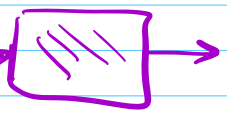
min number of painters



time ← X

min Painter

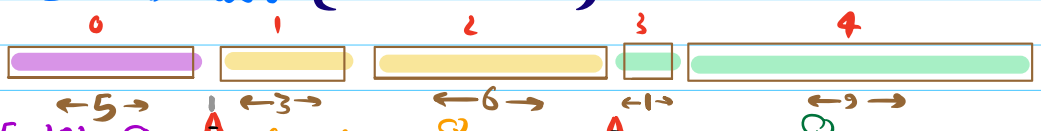
P →



→ min time

ex
#boards ← n=5 boards: {5, 3, 6, 1, 9} ← a
(0 1 2 3 4)

input array ← a




2 hours for each painter to paint 1 unit of board ← t=2

2×5=10h

(3+6)×2 = 18h

(1+9)×2 = 20h

max(10, 18, 20) = 20h

check: X = 15h → ans = -1

20h

2(5+3+6+1+9) = 48

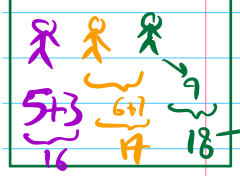
max( 5×2 (10), (3+6+1+9)×2 (38) ) = 38

max( (5+3)2 (16), (6+1+9)×2 (32) ) = 32

max( (5+3+6)2 (18), (1+9)×2 (20) ) = 28

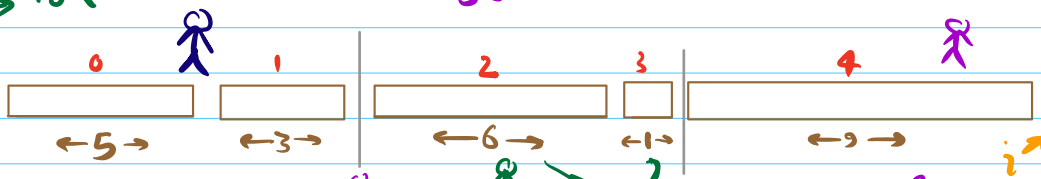
max( (5+3+6+1)2 (30), 9×2 (18) ) = 30

X = 20




18 < 20 ✓

idea?




5×2=10

3×2=6h

12

2

(8)

6

9×2=18

14

X = 20

20 − 10 = 10

X = 15

```
int NumPainters(a, X, t){
  ans=1  rTime = X        O(n) & O(n)
                          O(1) & O(1)
  for(i=0; i<n; i++){
    if(a[i]*t > X) ret -1
    if(a[i]*t <= rTime) rTime -= a[i]*t
    else{ans++; rTime=X-a[i]*t;}
  }
  ret ans
```

| X | P |
|---|---|
| 1 | 100 |
| 2 | 50 |
| 3 | 10 |
| 4 | 7 |
| 5 | 4 |

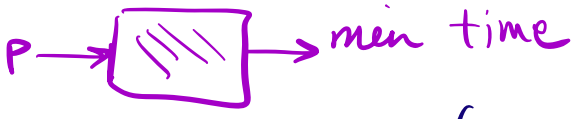
i ↓

5, 3, 6, 1, 9

rTime 20, 10, 8, ~~6~~ 2

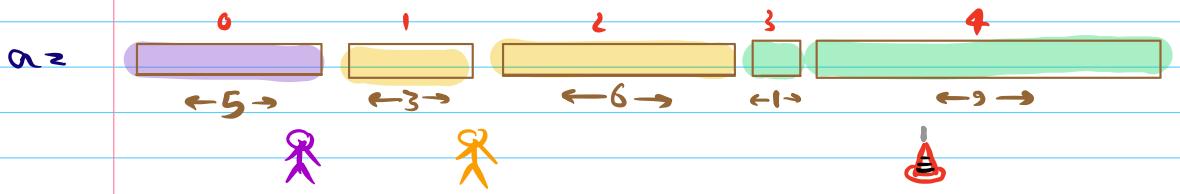
ans 1 2 3

5




P1.2 Find min time to paint all boards if P painters are available.

5




P=2

max( 5×2 (10), (3+6+1+9)×2 (38) ) =
max( (5+3)2 (16), (6+1+9)×2 (32) ) =
max( (5+3+6)2 (18), (1+9)×2 (20) ) =
max( (5+3+6+1)2 (30), 9×2 (18) ) =

1 X 1

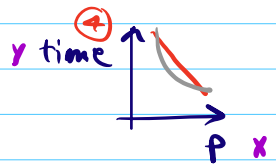
TC: O(1X1xn)

for X = 1 ... $\sum_{i=0}^{n-1} a_i$ * t

p2 ← NumPainters(a, X, t) → O(n)

p2_prev > P
p2 ≤ P

① P ↑ overall time ↓ P ↓ ② overall time




P is reverse proportional to time  $P \propto \frac{1}{time}$

P is no. of Painter

time

while

TC: O(n)

① Search space

$l = \max_{i=0}^{n-1}(a_i) \times t$ $\quad r = \sum_{i=0}^{n-1} a[i] \times t$

② check mid is answer (mid = (l+r)/2)

how? O(n)

if( NumPainters(a, mid, t) <= P &&

P2 NumPainters(a, mid-1, t) > P )

ret mid

③ Cut portion of search space by updating l or r

if( P2 > P )

l = mid+1

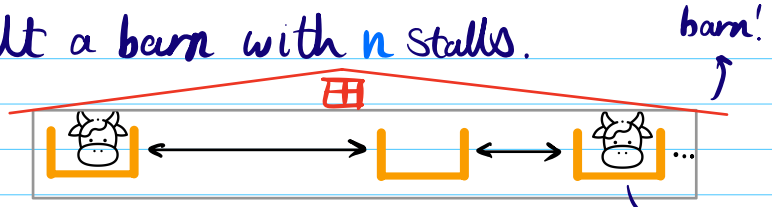
else r = mid-1

TC: O(n log n)

SC: O(1)

```
int binSearch(          ){
    n = a.len
    l = ; r = ①
    while(l <= r){
        mid = (l+r)/2 // floor
        if(  ②  ) ret mid
        if(  ③  ) l = mid+1
        else r = mid-1          // a[mid] > target
    }
    ret -1 // not found
}
```

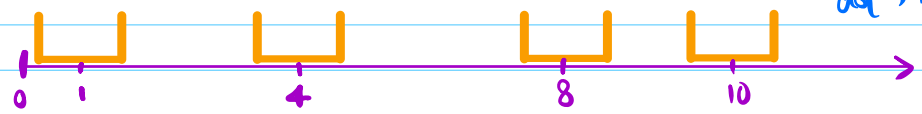
P2 Farmer has built a barn with n stalls.



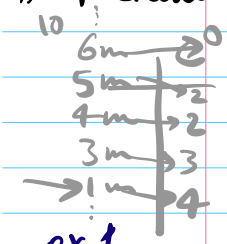


# of cows

① a[i]: location of ith stall in sorted oreder.

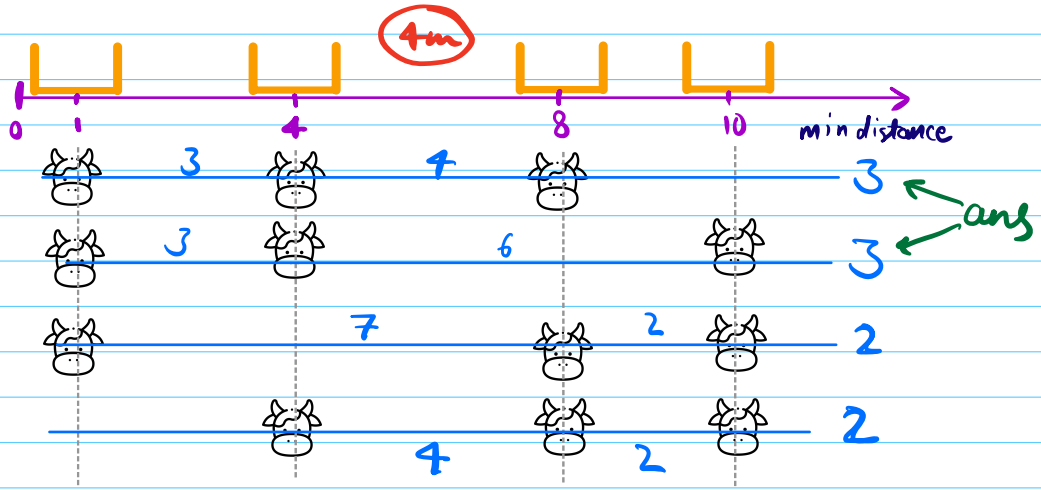



c <= n
i.e. # of cows less than # of stalls.




ex 1

③ Cows are aggressive towards each other so farmer wants to maximize the min distance between any pair of cows. Find max possible min distance.

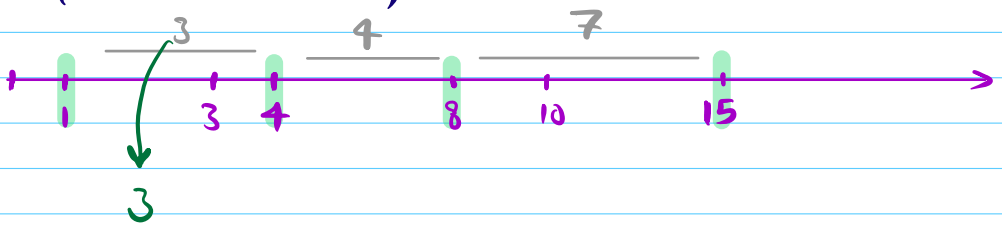



c=3

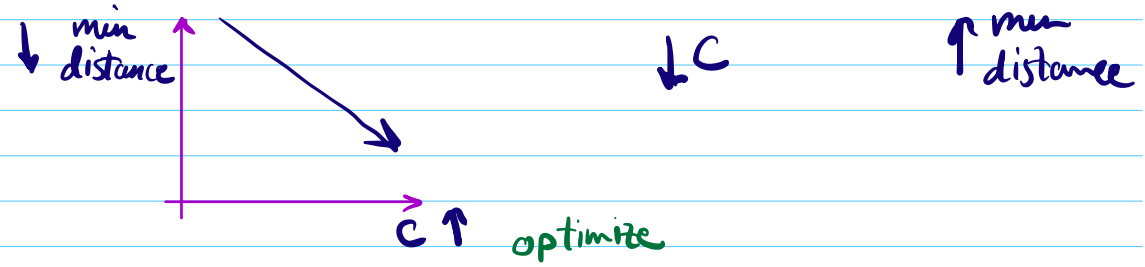
a:{1,4,8,10}

ex2 a:{1,3,4,8,10,15}

c=4




Solution




optimize

① Search Space $l = 1$ → $r = a[n-1] - a[0]$

while ←

② $(l+r)/2 \rightarrow$ mid → mid

for a given distance check if

mid ↓ if( can Place c cows with dist mid &&
cannot Place c cows with dist (mid+1) )

③ decide cut left or right

```
if(# NumberOfCows (dist=mid)<C)
        r = mid-1
else
        l = mid+1
```

TC: O(n) Algo ⊛ ret # of cows given max min distance

```
int NumberOfCows(int proposed_min_distance)
    c=1   last=a[0]
    for(i=0; i<n; i++){
        if(a[i]-last >= proposed_min_distance)
            c++
            last=a[i]
        }
    }
    ret c;
```

```
int binSearch(int a[], target){
    n = a.len
    l = 0 ; r = n-1
    while(l <= r){
        mid = (l+r)/2 // floor
        if(a[mid] == target) ret mid
        if(a[mid] < target) l = mid+1
        else r = mid-1                    // a[mid] > target
    }
    ret -1 // not found
}
```