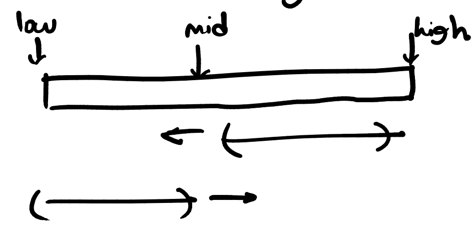
Recap: Binary search



Idea 1 linear search $O(n)$ $O(1)$

Idea 2 Binary search $O(\log n)$ $O(n)$

Not just used to find an element in the array.

It is also used to find an optimal value in a fixed range.

Q. n chefs a_i time to prepare a dish $a[] = \{a_1, a_2, \ldots, a_n\}$

d dishes. minimum time required to make these dishes.

7 mins 3 2 1 d=6 3 chefs: [2,3,5] (10)

O/p: 6 → 6 mins i/p: arr = [2,3,5] d= 6 10 mins. num = 5 3 2

range (1 , d* min time)

Trick: time to make d dishes → mid value

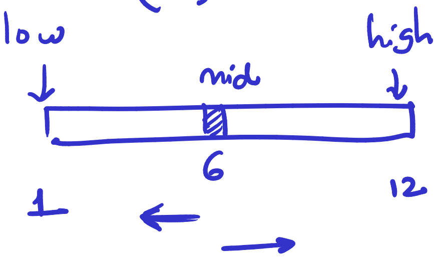
(1, 12)

going left ⇒ reduce mid ⇒ high = mid-1

In 'mid' minutes, I am making more than 'd' dishes.

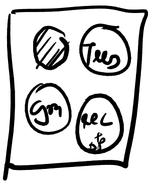
going right ⇒ increase mid ⇒ low = mid

I am making less than 'd' dishes.



```
low = 1 , high = d* min (arr) , ans = 0
while (low <= high):
    mid = (low + high)/2
    if (num_dishes (arr, mid) >= d):
        ans = mid
        high = mid -1
    else
        low = mid
return ans ;
```

```
num_dishes (arr, x):   time
    for a in arr:
        dishes += x/a
    return dishes
```



Summary:

① If you are performing BS on array i.e. some element of array is of your interest, mid represents the position of that number. Do normal BS.

② I am actually looking for some optimal value in the range:

a) The output that I am optimizing is my mid.

b) Now you can figure out min & max value of mid. This gives you low & high.

c) Figure out the rules when to go left & when to go right.

Otherwise, ans variable could be used as well.

③ Your mid at the end of loop, will be ans.

A → look for x → look for y → look for z → BS.

Mcq.

✓ ~~a~~

← (b)

~~c~~

(d)

multiple tactics :

A. question

① Constraint : gives me max Tc I can have

$n = 10^6$ $O(n^2)$ ×

× $O(n^2)$ { for for

Template

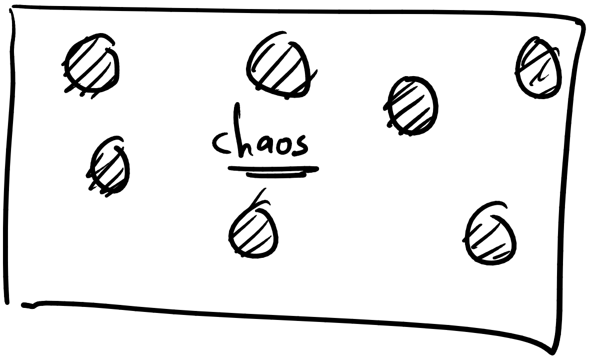
- sorting $O(n \log n)$ $n <= 10^5$
- searching   n very large
- partition $O(n)$ $n <= 10^8$
- .
- .
- .
- .

→ DP → combinatorial problems

+ Recursion   optimization problems

max, min, longest, smallest, shortest, etc.

Templates ?

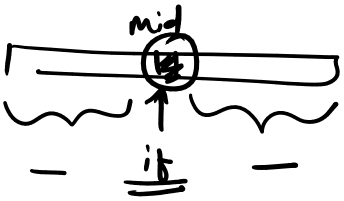


set of problems

```
{ if  mid == key
      return mid }
else if ________
      high = mid-1
else
      low = mid+1
```

✓

high = mid



low = mid