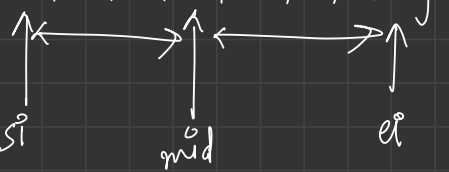
Infinite Sorted Array

arr[] = { 1, 2, 3, 4 . . . . . ∞ }

Key = 10

Binary Search

arr[] = { 0 1 2 3 4 5 6 7
1, 2, 3, 4, 5, 6, 7, 8 }

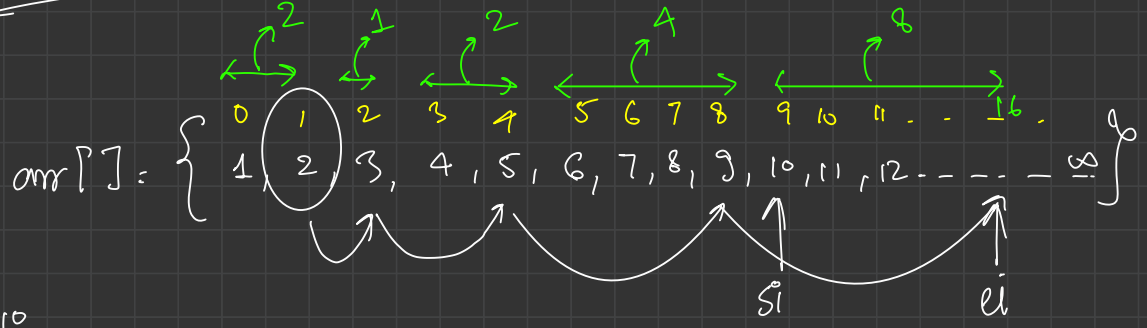


Key = 2

TC: O(logN)

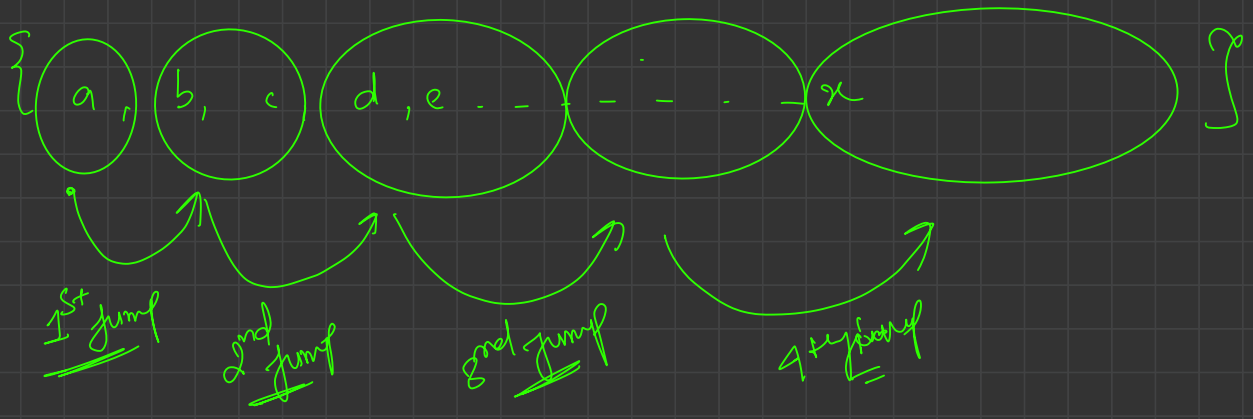
→ Size of the Array

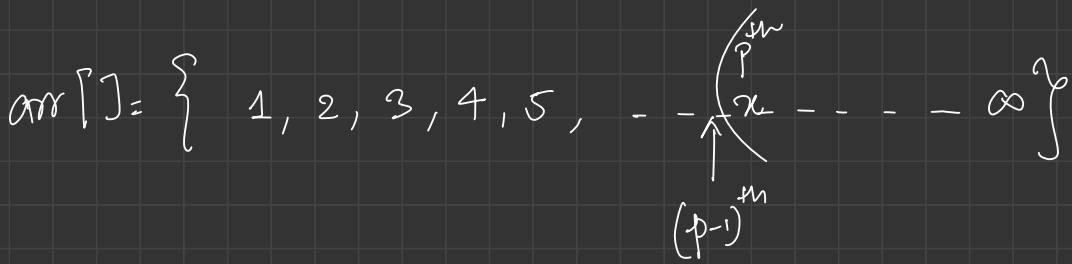
Search In Infinite Sorted Array



Key = 10

$$\begin{cases} si = ei + 1; \\ ei = 2 \times ei \end{cases}$$


{
a, b, c d, e x
}
1st jump
2nd jump
3rd jump
4th jump



si ⟶ p
ei ⟶ 2(p-1)

Number of elements = ei − si + 1
= 2(p-1) − p + 1
= 2p − 2 − p + 1 = (p − 1) elements

TC: of BS O(log P) { index of target }

ei	2^0	2^1	2^2	2^3	...	2^{K-1}
	1	2	4	8	...	$2^{(p-1)}$
jump	1	2	3	4	.....	K

$$2^{K-1} = 2^{(p-1)}$$

$$\frac{2^{K-1}}{2} = (p-1)$$

$$2^{K-2} = (p-1)$$

$$\log_2 2^{(K-2)} = \log(p-1)$$

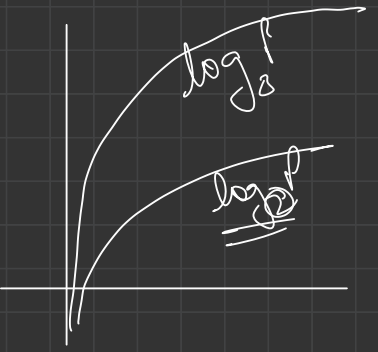
$$(K-2) = \log_2(p-1)$$

$$K \approx \log_2 p$$

TC: $O(K)$

: $O(\log_2 p)$

Total TC:



TC. to find Range + TC. to find Key

$\log_2 I + \log_2 I$

$2\log_2 I = O(\log_2 I)$

I → index of key in infinite sorted array

```
int find (int[] arr, int target)
{
    if (target < arr[0])
        return -1;

    int si = 0;
    int ei = 1;
    while (arr[ei] < target)
    {
        si = ei + 1;
        ei = 2 * ei;
    }

    int idx = BinarySearch(arr, si, ei, target);
    System.out.println(idx);
}
```

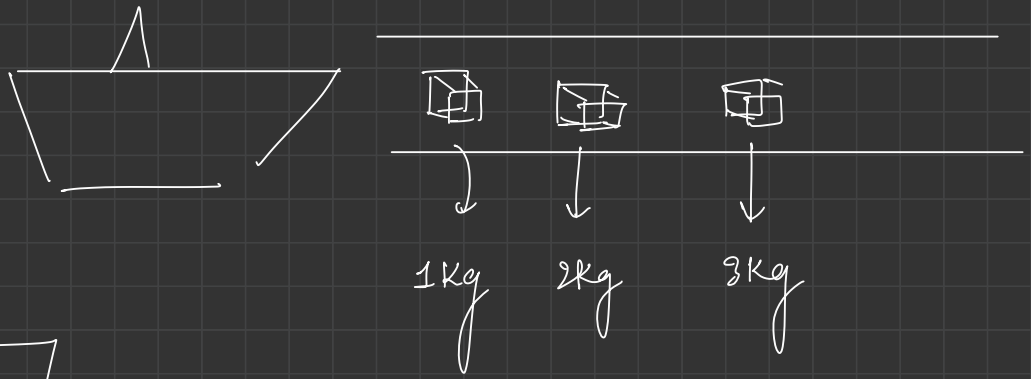
TC: $O((\log_2 i))$

SC: $O(1)$

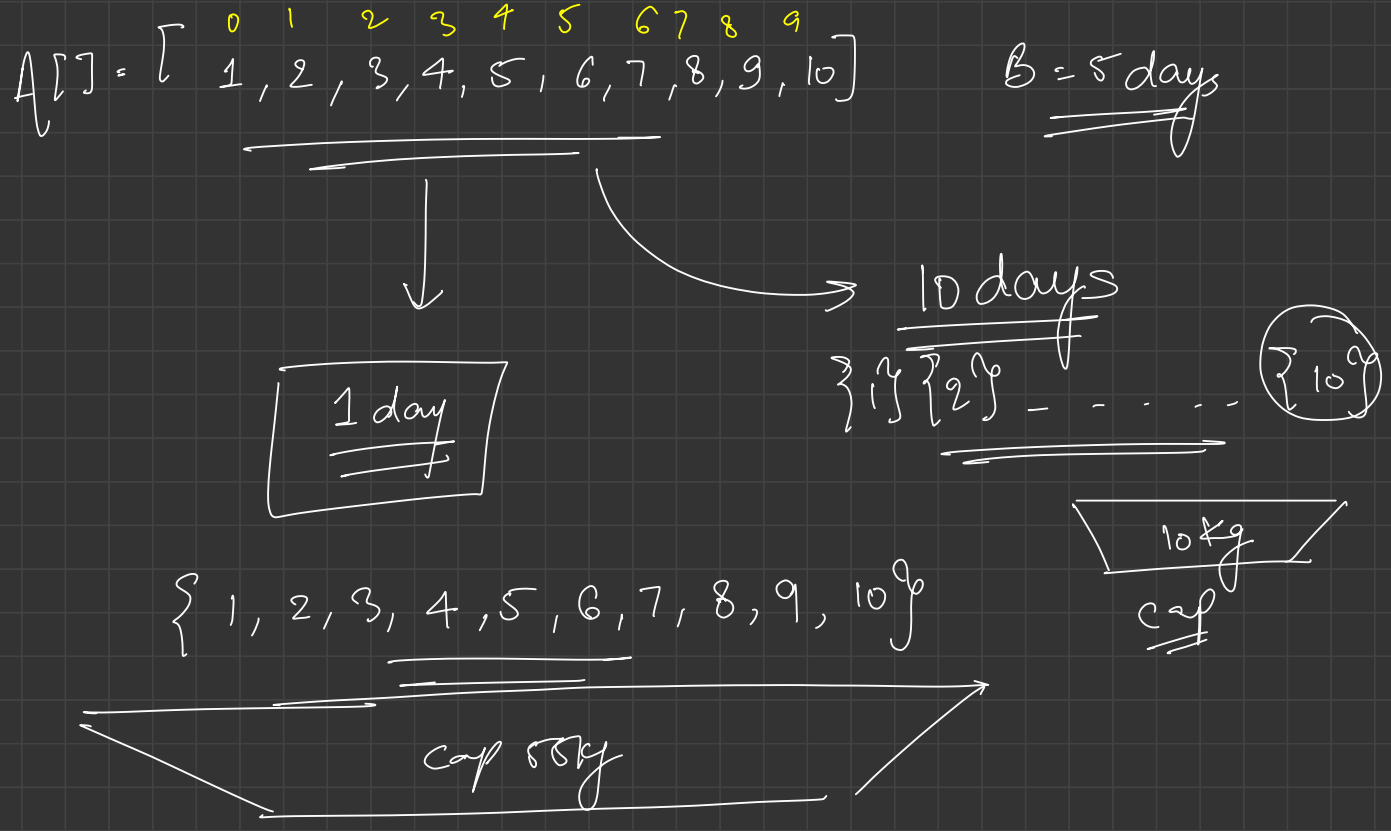
→ K jumps → $\log i$

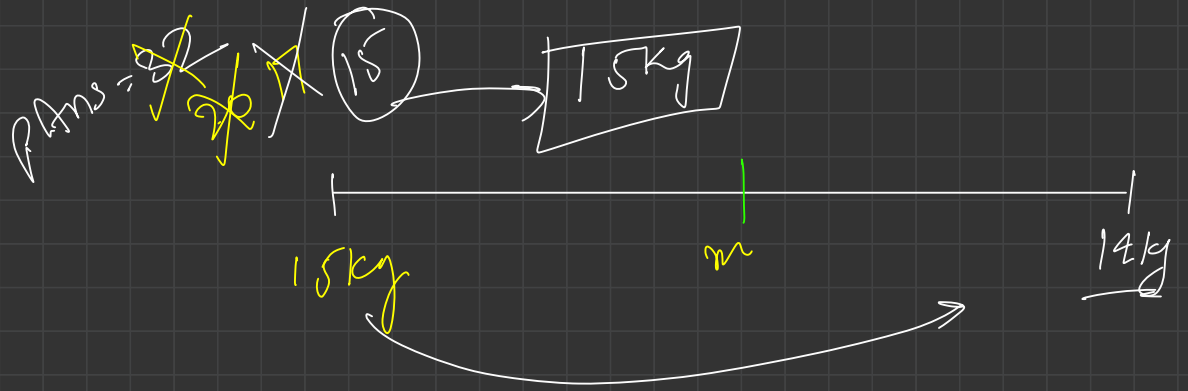
TC: $O(\log i)$

Capacity to ship Packages within D days



2 days


0 1 2 3 4 5 6 7 8 9
A[] = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
B = 5 days
1 day
10 days
{1} {2} ..... {10}
{ 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 }
cap 55kg
10kg
cap



cap 32kg

A[] = [0 1 2 3 4 5 6 7 8 9
1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

day 1 → 1 + 2 + 3 + 4 + 5 + 6 + 7

day 2 → 8 + 9 + 10

days = 2

cap : 20kg

0	1	2	3	4	5	6	7	8	9
1	2	3	4	5	6	7	8	9	10

day 1 = 1 + 2 + 3 + 4 + 5

day 2 = 6 + 7

day 3 = 8 + 9

day 4 = 10

days = 4

cap 14 kg

A[] = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9 : 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

day 1 → 1 + 2 + 3 + 4

day 2 → 5 + 6

day 3 → 7

day 4 → 8

day 5 → 9

6 days

day = 10 kg

cap = 17kg

A[] = [0: 1, 1: 2, 2: 3, 3: 4, 4: 5, 5: 6, 6: 7, 7: 8, 8: 9, 9: 10]

Day 1 → 1 + 2 + 3 + 4 + 5

Day 2 → 6 + 7

Day 3 → 8 + 9

Day 4 → 10

Day 4

cap = 15 kg

A[] = [0 1 2 3 4 5 6 7 8 9
1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

day1 → 1+2+3+4+5

day2 → 6+7

day3 → 8

day4 → 9

day5 → 10

day 2 5

Minimum Limit of Balls in a Bag

arr[] = {2, 4, 8, 2} maxOpt = 4

→ (2,2) (1,3) ✓

{2, 1, 3, 8, 2} 3

→ (1,7) (2,6) (3,5) (4,4) ✓

{2, 1, 3, 4, 4, 2} 2

→ (1,3) ✓ (2,2)

{2, 1, 3, 1, 3, 4, 2} 1

→ (1,3)

{ 2, 1, 3, 1, 3, 1, 3, 2 }

ans = 3

arr[] = { 2, 4, 8, 2 }

4, 4

maxopt = 4

{ 2, 4, 4, 4, 2 } (3)

{ 2, 2, 2, 4, 4, 2 } (2)

{ 2, 2, 2, 2, 4, 2 } (1)

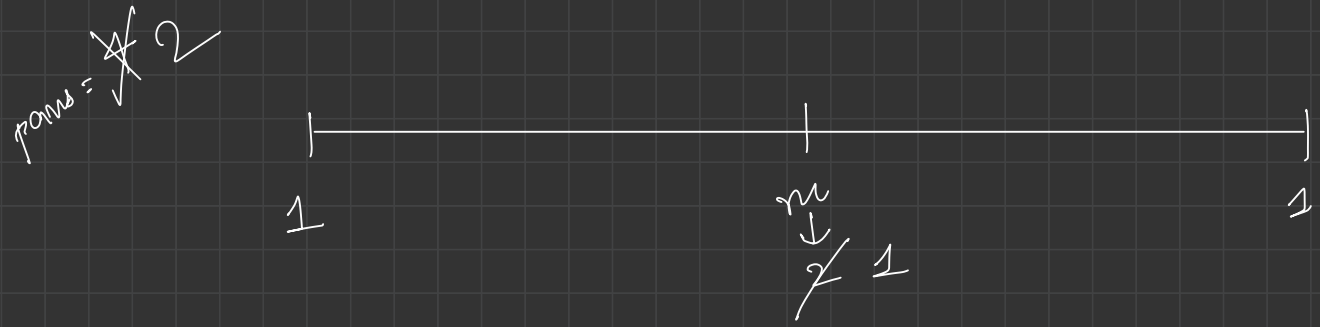
{ 2, 2, 2, 2, 2, 2, 2, 2 } (0)

2 = ans

arr[] = { 2, 4, 8, 2 } maxopt = 4

opt = 0 penalty = 8

opt = ∞ penalty = 1



arr[] = { 2, 4, 8, 2 }

{ 2, 4, 4, 4, 2 }

penalty = 4

4

opt = 1

arr[] = { 2, 4, 8, 2 }

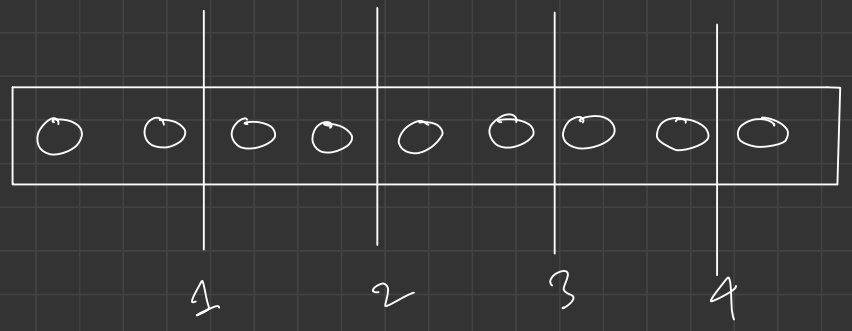
2 2

(4, 4)

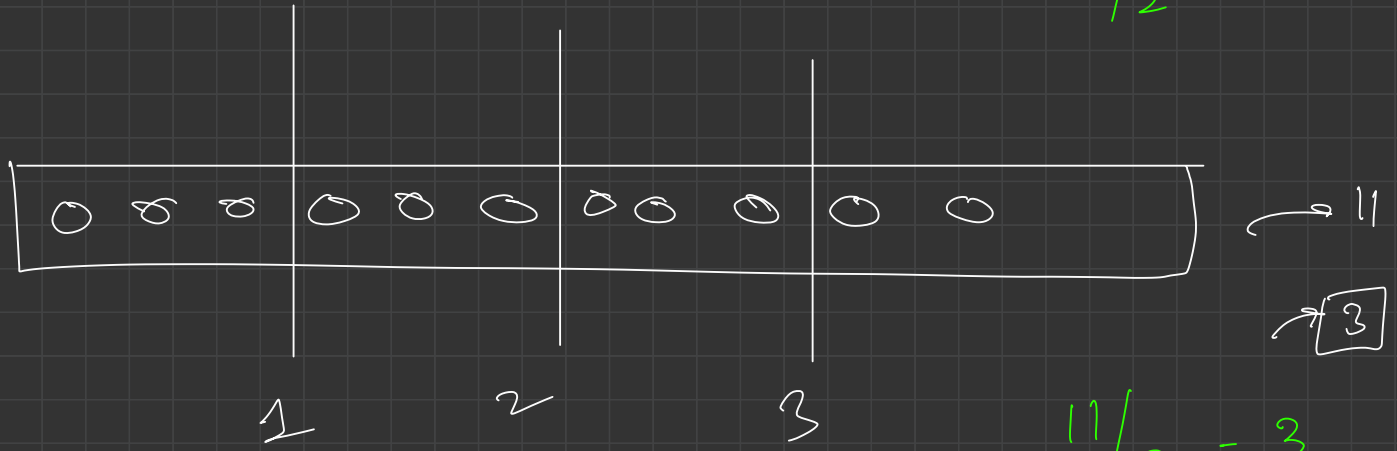
(2, 2) (2, 2)

penality = 2

opt = 1 + 3



$9/2 = 4$



$11/3 = 3$

$x \rightarrow$ Balls

penalty $= p$

$\Big\} \rightarrow \boxed{opt = x/p}$

$15/4 = 3$

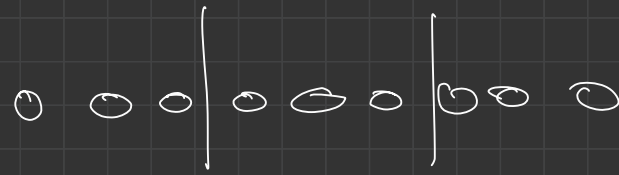
15

- (4, 11)
- (4, 4, 7)
- (4, 4, 4, 3)

$$\text{min}^{m} \text{ no. of opt} = \frac{\text{Balls in a Bag}}{\text{max}^{m} \text{ one can hold}}$$

$x = 8 \quad 2$

$8/2 = 4$



$x = 9$

$p = 3$

$9/3 = \boxed{3}$

if $x \% p == 0$

$ans = (x/p) - 1$

else

$ans = (x/p)$

PS Possible

Binary Trees

End!

Stacks
Queues
Hashing

Binary Search

Friday

Sat Sund Mon

Tu Wed