PW SKILLS

# Binary Search

# Today's checklist

1. Basics of Binary Search
2. Time complexity Analysis
3. Lower and upper bounds
4. Questions on binary search
5. Binary search on answer questions

# What and Why? → It is very very efficient

PW SKILLS

It is used to search a given element in a sorted space (array)

It is used to reduce the search space in half after every turn

Usually we apply B.S when it is given that 'it is a sorted array'

arr = { 10, 15, 21, 34, 81, 105, 180, 500 }

target = 105

Linear Search → T.C. = O(n)

# What and Why?

Sorted array

SKILLS

0 n-1

arr = { 10, 15, 21, 34, 81, 105, 180, 500 }

m

target = 81

Comparisons

1

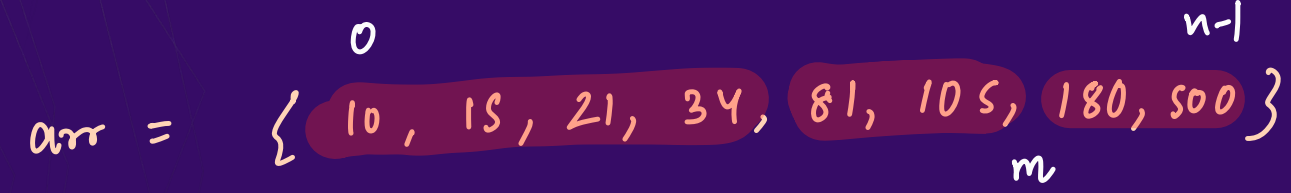
1+1+1

if(arr[mid] < target) go right

else if (arr[mid] > target) go left

else if (arr[mid] == target) done

# Binary search algorithm

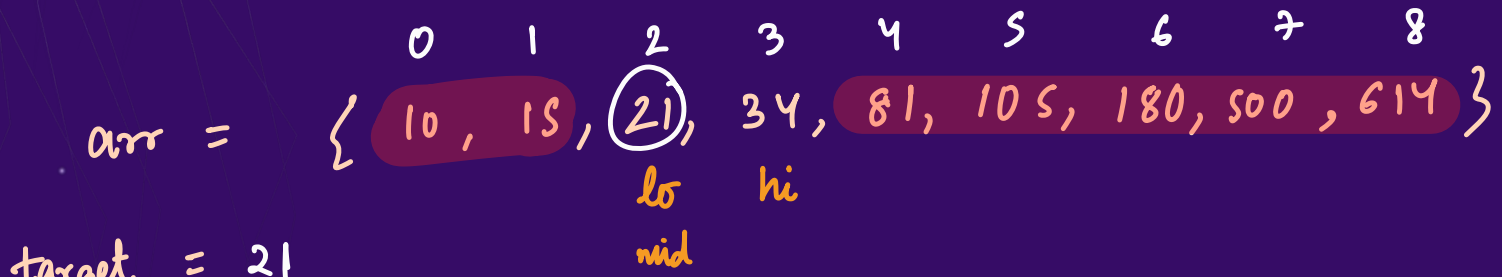



if(arr[mid] < target) go right

else if (arr[mid] > target) go left

else if (arr[mid] == target) done

# Binary search algorithm




```
flag = false;
while (lo <= hi){
    int mid = (lo+hi)/2;
    if(arr[mid] < target) lo = mid+1
    else if (arr[mid] > target) hi = mid-1;
    else if(arr [mid] == target) flag = true; break;
}
```

# Ques:

100000 → 50000 → 25000 → 12500 → 6250 → 3125

1 ← 3 ← 6 ← 12 ← 25 ← 50 ← 100 ← 200 ← 400 ← 800 ← 1600

## Q1 : Binary Search

↓

Reduction in search space by half

Consider that we have 100000 elements present in a sorted array

Linear Search

Tno = 100000

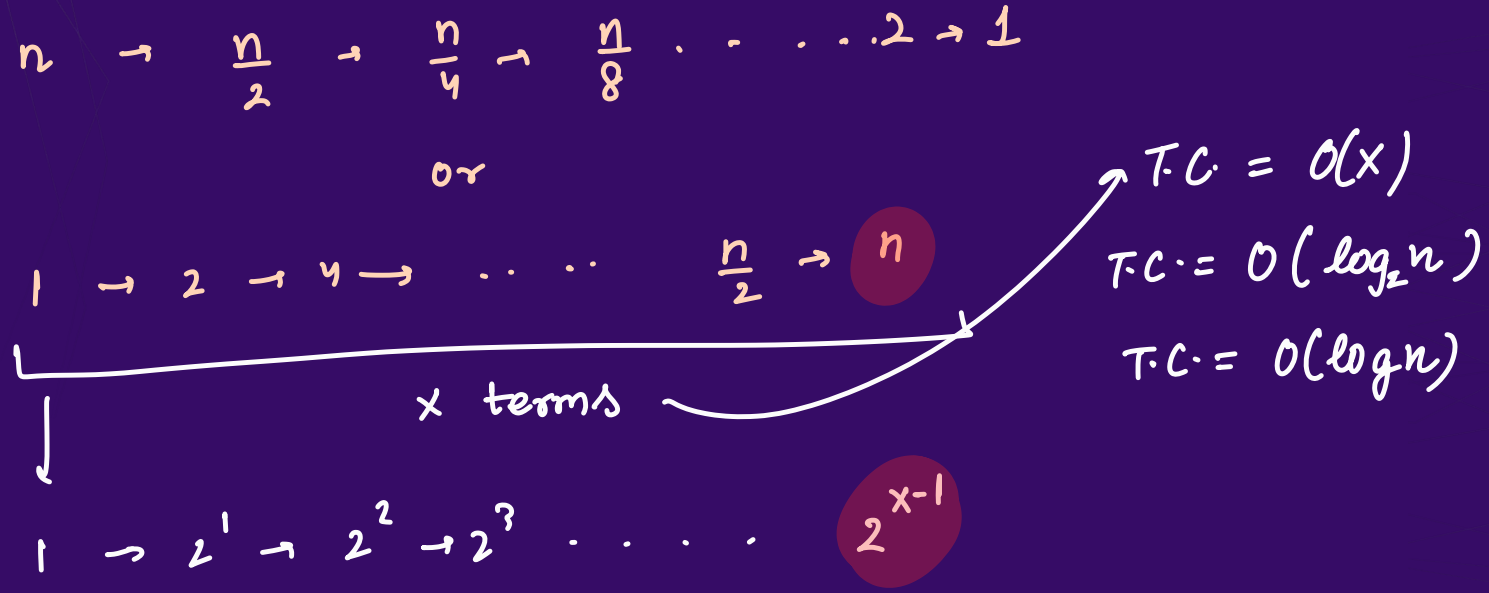
Binary Search

Tno ≈ 17

T.C. = O(log n)

[Leetcode 704]

# Time complexity analysis




$2^{x-1} = n$

$\log_2 n = x - 1$ or $x = 1 + \log_2 n$

# Time complexity analysis

PW SKILLS

$$mid = \frac{lo + hi}{2}$$

lo & hi are 'int'

$$-2^{31} <= lo, hi <= 2^{31} - 1$$

What if $(lo + hi) > 2^{31} - 1$ (Error: Integer Overflow Error)

→ But if $lo = 1$ & $hi = 2^{31} - 1$ then $mid = 2^{30} <= 2^{31} - 1$

$$mid = \frac{lo + hi}{2} = \frac{(hi - lo) + (lo + lo)}{2} = \frac{hi - lo}{2} + lo$$

# *Lower bound → index

**Q2 : Given a sorted integer array and an integer 'x', find the lower bound of x.** smallest idx such that arr[idx] >= x

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| arr = { | 10, | 20, | 30, | 30, | 40, | 50, | 60, | 70 } |

x = 40, lb = 4

x = 30, lb = 2

x = 35, lb = 4

x = 25, lb = 2

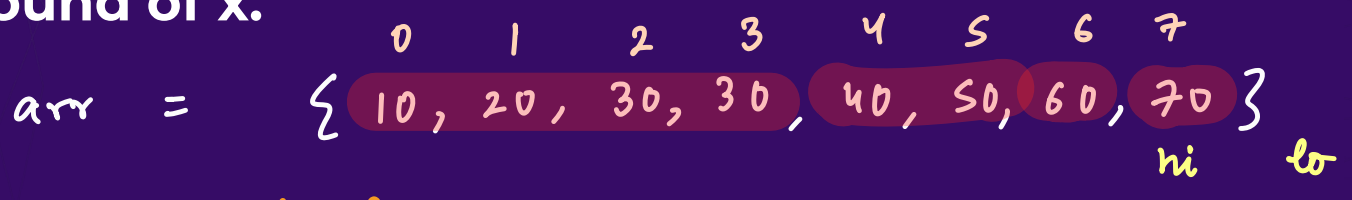
x = 80, lb = 8

x = 5, lb = 0

# *Lower bound

arr[idx] >= x
idx → minimise

**Q2 : Given a sorted integer array and an integer 'x', find the lower bound of x.**




x = 80

```
int lb = 8;
while(lo<=hi){
    mid = (lo+hi)/2;
    if(arr[mid]>=x){
        lb = min(lb, mid);
        hi = mid-1;
    }
    else lo = mid+1;
}
```

lb = 8

T.C. = O(log n)

# *Upper bound

**Q3 : Given a sorted integer array and an integer 'x', find the upper bound of x.** ub is the minimum idx such that arr[idx] > x

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| arr = { | 10, | 20, | 30, | 30, | 40, | 50, | 60, | 70 } |

x = 30
lb = 2
ub = 4

x = 35
lb = 4
ub = 4

x = 5
lb = 0
ub = 0

x = 80
lb = 8
ub = 8

Same code for lower bound except arr[mid] >= x the statement is arr[mid] > x

# Ques:

lower bound ?

upper bound ?

## Q4 : Find First and Last Position of Element in Sorted Array

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| arr = | 10 | 10 | 20 | 20 | 20 | 20 | 20 | 30 | 30 | 30 | 40 | 40 |

x = 20

fp = 2 = lb

lp = 6 = ub - 1

M-I : 1) Check if ele is present or not

2) Find lb

3) Find ub

[Leetcode 34]

# Ques:

PW SKILLS

## Q4 : Find First and Last Position of Element in Sorted Array

M-2:

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| arr = { | 10, | 10, | 20, | 20, | 20, | 20, | 20, | 30, | 30, | 30, | 40, | 40 } |
| | lo | | mid | | hi | | | | | | | |

target = 20

```
if(arr[mid] == target){
    if(arr[mid] == arr[mid-1])  hi = mid-1;
    else {fp = mid; break;}
}
else if (arr[mid] < target)  lo = mid+1;
else if (arr[mid] > target)  hi = mid-1;
```

[Leetcode 34]

# Ques:

PW SKILLS

## Q4 : Find First and Last Position of Element in Sorted Array

```
                 0   1   2   3   4   5   6   7   8   9   10  11
arr = { 10, 10, 20, 20, 20, 20, 20, 30, 30, 30, 40, 40 }
                                 mid lo              hi
```

target = 20

```
if(arr[mid] == target){
    if(arr[mid] == arr[mid+1]) lo = mid+1;
    else { ans = mid; break; }
}
else if (arr[mid] < target) lo = mid+1;
else if (arr[mid] > target) hi = mid-1;
```

[Leetcode 34]

# Ques:

PW SKILLS

## Q5 : Peak index in a Mountain Array

arr = {10, 20, 30, 50, 40, 20, 4}
peak

peak = 3

if ( arr[i] > arr[i-1] && arr[i] > arr[i+1]) → peak = i

Binary Search → array must be sorted ?

Not Everytime !

[Leetcode 852]

# Ques:

PW SKILLS

## Q5 : Peak index in a Mountain Array

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|---|
| arr = { | 10, | 20, | 30, | 40, | 50, | 60, | 70, | 20, | 10 } |
| | | | | | | lo | mid | hi | |

```
if (arr[mid] > arr[mid-1] && arr[mid] > arr[mid+1]) return mid;
else if (arr[m] > arr[m-1] && arr[m] < arr[m+1]) lo = mid + 1;
else if (arr[m] < arr[m-1] && arr[m] > arr[m+1] hi = mid - 1;
```

[Leetcode 852]

# Ques:

PW SKILLS

## Q6 : Sqrt(x)

for any no. x , $0 =< Sqrt(x) <= x$ hint

x = 16


```
if(m^2 == x) return m;
else if (m^2 > x) hi = m-1;
else if (m^2 < x) lo = m+1;
```

[Leetcode 69]

# Ques:

## Q6 : Sqrt(x)

$hi = 2^{25}$ $lo = 0$

$mid = 2^{24}$

$mid * mid = 2^{24} \times 2^{24} = 2^{48}$ > Integer.MAX_VALUE

SKILLS

$x = 14$


m

$m^2 == x$ or $m == x/m$ | long

```
if(m^2 == x) return m;
else if (m^2 > x) hi = m-1;
else if (m^2 < x) lo = m+1;
```

[Leetcode 69]

# Ques: T.C. = O(log n)

PW SKILLS

## Q7 : Search in Rotated Sorted Array [Very important]

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| arr = | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| | 3 | 4 | 5 | 6 | 7 | 0 | 1 | 2 |

Method - 1 : 1) Find the pivot idx

↓

O(log n)

2) Binary Search in

(0, p) & (p+1, n-1)

[Leetcode 33]

# Ques: pivot = largest

PW SKILLS

## Q7 : Search in Rotated Sorted Array

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| arr = | 3 | 4 | 5 | 6 | 7 | 0 | 1 | 2 |
| | | | | | lo | m | | hi |

```
int p;
if (arr[m] > arr[m-1] && arr[m] > arr[m+1] ) p = mid ; break;
else if (arr[m] < arr[m-1] && arr[m] < arr[m+1] ) p = mid - 1 ; break;
else if (arr[m] > arr[m-1] && arr[m] < arr[m+1] ){
    if(arr[m] > arr[n-1] ) lo = mid + 1 ;
    else  hi = mid - 1;
}
```

[Leetcode 33]

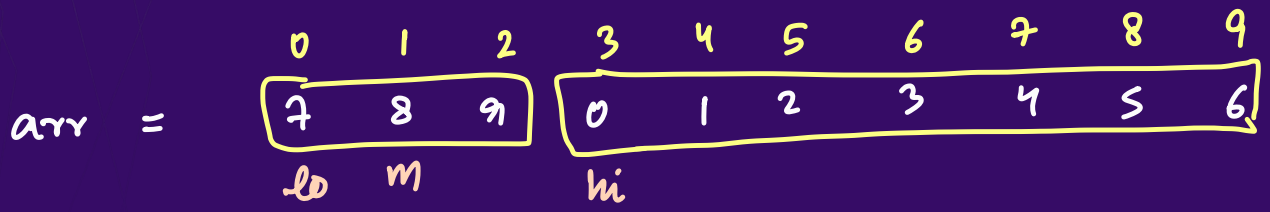
# Ques:

{0 1 2 3 4 5 6 7 8 9}

SKILLS

## Q7 : Search in Rotated Sorted Array

target = 8




```
else if (arr[mid] <= arr[hi]){  //right
    if ( arr[mid]<= target && target <= arr[hi]) lo =mid+1;
    else  hi= mid - 1;

}
else if( arr[mid] > arr[hi]){// you are in left sorted array  lo-mid
    if (arr[lo ] <= target && target <= arr[mid ])  hi=mid-1
    else  lo = mid + 1;
}
```

[Leetcode 33]

# Ques:

## Q7 : Search in Rotated Sorted Array

```
int n = arr.length;
int lo = 0, hi = n-1;
while(lo<=hi){
    int mid = lo + (hi-lo)/2;
    if(arr[mid]==target) return mid;
    else if(arr[mid]<=arr[hi]){ // i am in right sorted array (mid to high everything is sorted)
        if(target>arr[mid] && target<=arr[hi]) lo = mid + 1;
        else hi = mid - 1;
    }
    else{ // i am in left sorted array (lo to mid everything is sorted)
        if(target>=arr[lo] && target<arr[mid]) hi = mid - 1;
        else lo = mid + 1;
    }
}
return -1;
```

# Ques:

PW SKILLS

## Q8 : Find K Closest Elements

arr = { 1, 2, 3, 4, 5 }  i, j, lb  x = 3  K=2

arr = { 1, 3, 5, 8, 9, 10 }  i, j, lb  x = 6  K=3

lb = 3

arr = { 1, 2, 3, 4, 5 }  x = -1  K = 4

[Leetcode 658]

# Ques:

PW SKILLS

## Q8 : Find K Closest Elements

| | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| arr = { | 1, | 2, | 3, | 4, | 5 } |

$x = 7$, $k = 3$

↓

$lb = 5$

For lb → arr[lb] >= x

↓ minimise

[Leetcode 658]

# Ques:

## Q8 : Find K Closest Elements

```
List<Integer> ans = new ArrayList<>();
int n = arr.length;
if(x<arr[0]){
    for(int i=0;i<k;i++){
        ans.add(arr[i]);
    }
    return ans;
}
if(x>arr[n-1]){
    for(int i=n-1;i>=n-k;i--){
        ans.add(arr[i]);
    }
    Collections.sort(ans);
    return ans;
}
```

Base Cases

```
int lb = n;
int lo = 0, hi = n-1;
while(lo<=hi){
    int mid = lo + (hi-lo)/2;
    if(arr[mid]>=x){
        lb = mid;
        hi = mid - 1;
    }
    else lo = mid + 1;
}
```

Binary Search to find lb of x

2 pointer

```
int j = lb, i = lb - 1;
while(k>0 && i>=0 && j<n){
    int di = Math.abs(x-arr[i]);
    int dj = Math.abs(x-arr[j]);
    if(di<=dj){
        ans.add(arr[i]);
        i--;
    }
    else{
        ans.add(arr[j]);
        j++;
    }
    k--;
}
while(i<0 && k>0){
    ans.add(arr[j]);
    j++;
    k--;
}
while(j==n && k>0){
    ans.add(arr[i]);
    i--;
    k--;
}
Collections.sort(ans);
return ans;
```

# Ques:

PW SKILLS

## Q9 : Find the smallest Divisor given a Threshold

sd = 5

arr = {5, 1, 9, 2}

t = 6
max sum of array

d = 1 Sum = 17 ✗
d = 2 Sum = 3 + 1 + 5 + 1 = 10 ✗
d = 3 Sum = 2 + 1 + 3 + 1 = 7 ✗
d = 4 Sum = 2 + 1 + 3 + 1 = 7 ✗

d = 5 Sum = 1 + 1 + 2 + 1 = 5 < 6 ✓

d = 9 Sum = $\frac{5}{9} + \frac{1}{9} + \frac{9}{9} + \frac{2}{9}$

1 + 1 + 1 + 1 = 4 < 6 (Acceptable)

[Leetcode 1283]

# Ques:

## Q9 : Find the smallest Divisor given a Threshold

```
int d;
for(d=1;d<=mx;d++){
    int sum = 0;
    for(int i=0;i<n;i++){
        if(arr[i]%d==0) sum += arr[i]/d;
        else sum += arr[i]/d + 1;
    }
    if(sum<=t) return d;
}
return d;
```

$arr[i] <= 10^6$

$mx <= 10^6$

$\log_2 mx <= \log_2 10^6$

$< 24$

$6 \log_2 10$

$3-4$

$T.C. = O(n * mx)$

largest element of array

[Leetcode 1283]

# Ques:

PW SKILLS

## Q9 : Find the smallest Divisor given a Threshold

arr = { 44, 22, 33, 11, 1 }  t = 6

lo hi

d = 1 to 44 ✓ not linearly

lo = 1 ~~23~~ ~~28~~ 31

hi = ~~44~~ 32

mid = ~~22~~ ~~33~~ ~~27~~ ~~30~~ 31

d = 22 sum = 2 + 1 + 2 + 1 + 1 = 7 ✗

d = 33 sum = 2 + 1 + 1 + 1 + 1 = 6 <= 6 ✓

d = 27 sum = 2 + 1 + 2 + 1 + 1 = 7 ✗

d = 30 Sum = 2 + 1 + 2 + 1 + 1 = 7 ✗

[Leetcode 1283]

# Ques:

PW SKILLS

## Q9 : Find the smallest Divisor given a Threshold

```
int d = 1;
int lo = 1, hi = mx;
while(lo<=hi){ → log(mx)
    int mid = lo + (hi-lo)/2;
    if(isLess(mid,arr,t)){ O(n)
        d = mid;
        hi = mid - 1;
    }
    else lo = mid + 1;
}
return d;
```

```
public boolean isLess(int mid, int[] arr, int t){
    int sum = 0;
    for(int i=0;i<arr.length;i++){
        if(arr[i]%mid==0) sum += arr[i]/mid;
        else sum += arr[i]/mid + 1;
    }
    if(sum<=t) return true;
    return false;
}
```

$T.C. = O(n * \log mx)$

[Leetcode 1283]

# Ques: hi = sum , lo = mx

PW SKILLS

## Q10 : Capacity to ship packages within D days

arr = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 } d = 5

1 2 3 | 4 | 5 | 6 | 7

C = 8 → days = 8 ✗ NEVER

C = 11 → days = 6 ✗

C = 55 → days = 1 ✓

↓

Ship Capacity

[Leetcode 1011]

# Ques: lo = mx , hi = sum

PW SKILLS

## Q10 : Capacity to ship packages within D days

arr = { 3, 2, 2, 4, 1, 4 }  days = 3

3 2 | 2 | 4 1 | 4

minC = 6

lo = 4 ~~5~~ 6
hi = ~~16~~ 9 5
mid = ~~10~~ 8 4 5

C = 10 → d = 2 ✓
C = 6 → d = 3 ✓
C = 4 → d = 5 ✗
C = 5 → d = 4 ✗

[Leetcode 1011]

# Ques:

PW SKILLS

## Q10 : Capacity to ship packages within D days

arr = { 3, 2, 2, 4, 1, 4 }  days = 3

3 2 2 | 4 1 | 4

D = 1 ✗ 3

C = 8

load = 0 ✗ ✗ ✗ 4 ✓
4

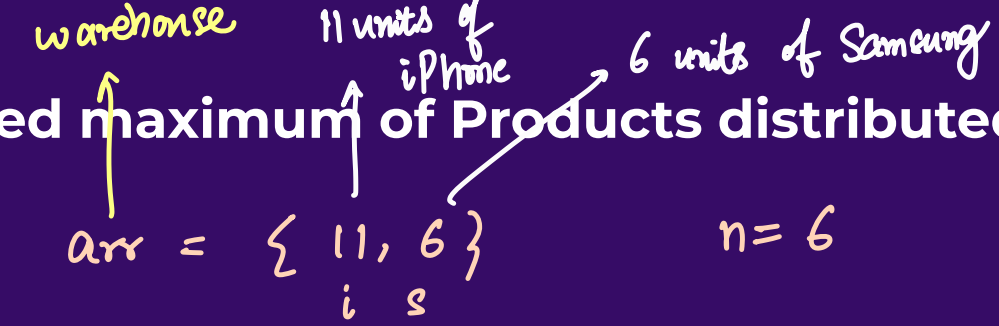
```
if(load + arr[i] <= C) load += arr[i]
else{
    load = arr[i];
    D++;
}
```
3

[Leetcode 1011]

# Ques:

PW SKILLS

**Q11 : Minimized maximum of Products distributed to any store**




[Leetcode 2064]

# Ques:

PW SKILLS

**Q11 : Minimized maximum of Products distributed to any store**

arr = {15, 10, 10}  n=7
(a, b, c)

Sa Sa Sa Sb Sb Sc Sc

lo = 1 5

hi = 15 7 8 4

mid = 8 4 6 5

maxAns = 8 6 5

[Leetcode 2064]

# Ques:

PW SKILLS

**Q11 : Minimized maximum of Products distributed to any store**

arr = { 15, 12, 10 }  n = 3

stores = 10

maxQ = 4

4 4 4 3 | 4 4 4 | 4 4 2

stores += arr[i]/maxQ + 1

↓

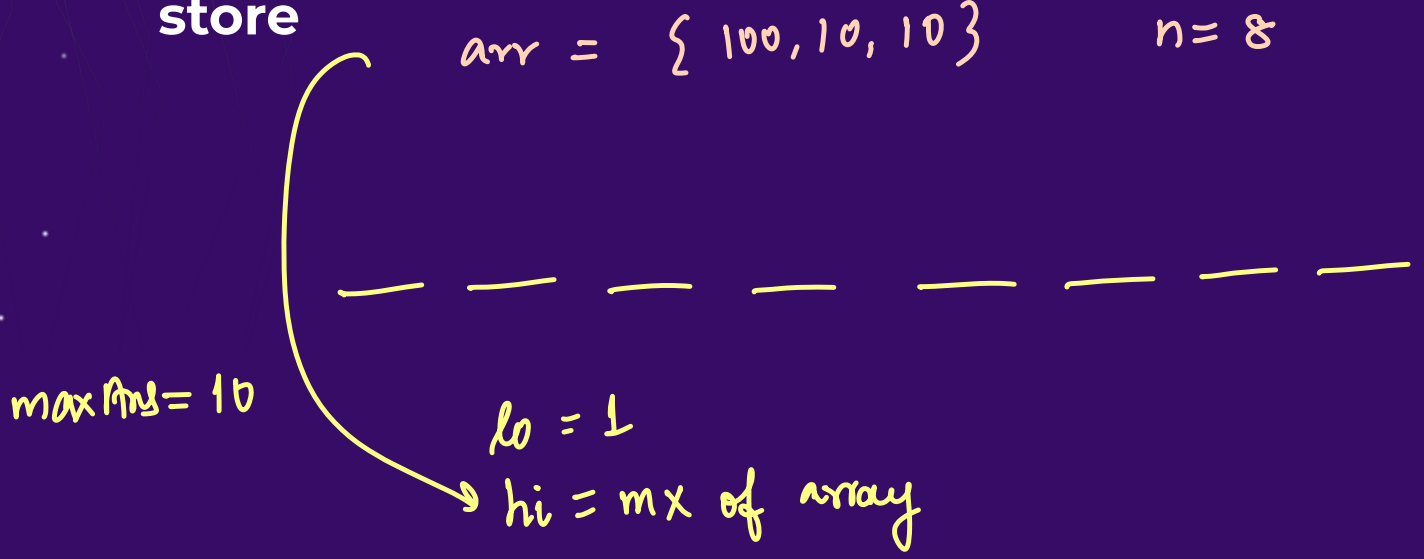
arr[i] % maxQ != 0

[Leetcode 2064]

# Ques:

PW SKILLS

## Q11 : Minimized maximum of Products distributed to any store




[Leetcode 2064]

# Ques:

Aggresive Cows, Koko Eating Bananas, Min. Time to complete Trips

SKILLS

oducts distributed to any

```java
public int minimizedMaximum(int n, int[] arr) {
    int m = arr.length, mx = Integer.MIN_VALUE;
    for(int i=0;i<m;i++){
        mx = Math.max(mx,arr[i]);
    }
    int lo = 1, hi = mx;
    int ans = 0;
    while(lo<=hi){
        int mid = lo + (hi-lo)/2;
        if(isPossible(mid,n,arr)){
            ans = mid;
            hi = mid - 1;
        }
        else lo = mid + 1;
    }
    return ans;
}
```

```java
public boolean isPossible(int maxQ, int n, int[] arr){
    int stores = 0;
    for(int i=0;i<arr.length;i++){
        if(arr[i]%maxQ == 0) stores += arr[i]/maxQ;
        else stores += arr[i]/maxQ + 1;
    }
    if(stores>n) return false;
    return true;
}
```

[Leetcode 2064]

# Ques:

O(log n)

## Q12 : Kth Missing Positive Number (Binary Search + Maths)

| | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| arr = | 2, | 3, | 4, | 7, | 11 |
| | 1 | 1 | 1 | 3 | 6 |

K = 5

1 2 3 4 5 6 7 8 9 10 11

Easy ka

[Leetcode 1539]

# Ques:

B-S. ✓ Math

PW SKILLS

## Q12 : Kth Missing Positive Number

| | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| arr = | 2, | 3, | 4, | 7, | 11 |
| | | | | hi | lo |

$K = 5$

missed = arr[mid] − (mid + 1)

if (missed < K) lo = mid + 1

else hi = mid − 1

$K^{th}$ missing number = arr[hi] + extra

extra = K − (missed no.s till hi)

= K − (arr[hi] − (hi + 1))

maths

Observations : The $K^{th}$ missing no. is b/w arr[hi] & arr[lo] [Leetcode 1539]

# Ques:

PW SKILLS

## Q12 : Kth Missing Positive Number

$K^{th}$ missing no = arr[hi] + extra

= arr[hi] + K − (arr[hi] − (hi+1))

= ~~arr[hi]~~ + K − ~~arr[hi]~~ + (hi+1)

= K + (hi+1)

or

K + lo

[Leetcode 1539]

**Ques:** arr = {2, 3, 4, 7, 11} (0 1 2 3 4; hi lo) k = 5

PW SKILLS

## Q12 : Kth Missing Positive Number

$K^{th}$ missing no. is b/w arr[hi] & arr[lo]

→ $K^{th}$ missing no = arr[lo] − x

x = missing no.s till arr[lo] − K − 1

x = arr[lo] − (lo + 1) − K − 1

$K^{th}$ → arr[lo] − (arr[lo] − (lo + 1) − (K − 1))

arr[lo] − arr[lo] + lo + 1 + (K − 1)

[Leetcode 1539]

# Ques:

PW SKILLS

## Q12 : Kth Missing Positive Number

```
int lo = 0, hi = arr.length - 1;
while(lo<=hi){
    int mid = lo + (hi-lo)/2;
    int missed = arr[mid] - (mid+1);
    if(missed<k) lo = mid + 1;
    else hi = mid - 1;
}
return k + lo;
```

[Leetcode 1539]

# THANK YOU