1.8) Given a matrix, row wise sorted & column wise sorted, & element value
Given a K, search if K is present or not?

| | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 0 | 2 | 3 | 7 | 13 | 14 | 15 |
| 1 | 16 | 19 | 20 | 23 | 24 | 25 |
| 2 | 30 | 32 | 33 | 34 | 36 | 37 |
| 3 | 38 | 39 | 40 | 42 | 43 | 44 |

K = 50

K = 43 → 23 (after applying BS)

$3^{rd}$ Apply BS on

K = 38 → 30

→ Solutions

1) Search ele by ele
TC: O(NM) SC: O(1)

2) Apply BS row by row
TC: N log M SC: O(1)
Apply BS col by col
TC: M(log N) SC: O(1)

3) Take optimal from TC or BS
TC: O(N+M) SC: O(1)

4) Treat row by row matrix
domain with K &
apply BS only on 1 row
TC: N + log M

5) Apply BS on base column
& get idx i & a given K,
Now along as row mat
again apply BS.
TC: [log N + log M]

TC: [log N]

→ Both $5^{th}$ & $6^{th}$ are same

$\log A + \log B = \log A B$

10 → → (1,1) → (1,2, 1,3)
→ → (1,2) → (1,3, 1,4)
11 → (1,1) → (1,2, 1,3)
→ 5 notice last row, column

$\boxed{\frac{M+N}{2}}$

$^{th}$ 8:

Assume 2D → 1D
Convert 2D → 1D TC: O(MN) SC: O(MN) → This is taking time
BS on 1D TC: $\log_2(MN)$ SC: O(1)

Hint: Without actually converting 2D → 1D, we
assume 2D matrix is 1D & apply BS

```
l = 0, h = N*M-1,
while( l < = h){
    int mid = (l+h)/2
    // mid index based on 1D value
    // convert 1D index → 2D index value
    int r = mid/N, int c = mid%N;
    if (mat[r][c] == K){
        return True;
    }
    if( mat[r][c] > K){
        high = mid-1
    }
    else{
        low = mid+1
    }
}
return False;
```

→ VIMP used in graphs

TC: $\log(N*M)$ SC: O(1)

2) Special Integer
Given an array which contains all +ve Integers,
find max K such that, there is no subarray sum of size K > B

A[7] = [3 2 6 9 4 7 8], B = 20
0 1 2 3 4 5 6
5 ≤ 20 13 ≤ 20
8 ≤ 20 11 ≤ 20
15 ≤ 20 15 ≤ 20

K = 2

K = 3
3 2 6 → 11 ≤ 20
2 6 9 → 17 ≤ 20
6 9 4 → 19 ≤ 20
9 4 7 → 20 ≤ 20
4 7 8 → 19 ≤ 20
Final ans

K = 4
3 2 6 9 → 20 ≤ 20
2 6 9 4 → 20 ≤ 20
6 9 4 7 → 26 > 20

[This subarray sum > B]
[K > 4 proved correct]

Subtask:
Given K size write a func to check
whether it can be a ans or not?

```
bool check(int arr[], int n, int k, int B){
    // 1st subarray [0, K-1],
    int sum = 0, maxsum = 0;
    for (int i = 0; i < k; i++){
        sum = arr[i]
    }
    maxsum = max(maxsum, sum);
    for (int i = k; i < n; i++){
        sum = sum + arr[i] - arr[i-k]
        maxsum = max(maxsum, sum);
    }
    // maxsum contains max subarray sum of subarrays
    of size K.
    if( maxsum <= B){
        return True;
    }
    else{ return False;}
}
```

sum = [0, K-1] + arr[K] - arr[0]
= [1, K]

TC: O(N) SC: O(1)

Simple BF:
Apply check function for all
K value from [0, N] &
get max K, which is valid

TC: [N+1] * [N] SC: O(1)
= O(N²) SC: O(1)

Optimization:
In above example K = 5 6 7 8
F F F F — F

In above example K = 2
T T

Pattern: T T T T .. F F F F F

If mid is valid
ans = mid
goto right
Better ans on right
else
high = mid-1

Pseudocode BS

```
lo = 0, hi = n,
while ( lo <= hi){
    int mid = (lo+hi)/2
    if( check(A, N, mid, B)){
        ans = mid;
        low = mid+1
    }
    else{
        high = mid-1
    }
}
return ans;
```

Binary Search Iteration
→ log (N)

TC: log(n) * n
SC: O(1)

Concept:

| | BS Iteration | TC |
|---|---|---|
| : N/2 | $\log_2^N = 1$ | → $\log_2^N$ |
| : N/3 | $\log_3^N = 2$ | → $2\log_3^N$ |
| : N/4 | $\log_4^N = 3$ | → $3\log_4^N$ |
| : N/N | $\log_N^N = (N-1)$ | → (N-1)(1) |

$\log_2^N$ $2\log_3^N$ $\boxed{\frac{4}{3} = 3}$
$\log_3^{N^2}$
$\log_2^{N^2} = \frac{1}{\log_2 4}\log_2 N^2 = \log_2^{N^{3/4}}$

$\boxed{\log_2^N < \log_2^{N^{3/4}}}$

TC increasing

Given A, B, C, find $A^{th}$ magical Number
↳ If it can be divisible by
either B or C

B = 2, C = 3, A = 8 ans = 12 → [1, 2 3]
↳ is is $8^{th}$ multiple of 2
actual $8^{th}$ magical < 16

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

⇒ BF: [1-∞] & get $A^{th}$ magical number
TC: $A^{th}$ min(B, C)

B = 10 — $10^{th}$ of multiple : 1000
C = 15 — $10^{th}$ of multiple : 1500
A = 300

int retmcount(int X, int B, int C){
int cnt = 0;
int LCM = (B*C)/gcd(B, C)
cnt = (X/B) + (X/C) - (X/LCM)
return cnt
}

Hint:
B = 2, C = 3, A = 10
mid = 20 : 20 ≥ 10
30 31 32 ...
20 20 20
mid = 10 : 7 < 10
8 9 10
<10 <10 <10

Final BS using Hint 2
min(B,C)
int lo = 0, hi = min(B, C) * A;
int LCM = (B*C)/gcd(B, C)
while( lo <= hi){
int mid = (lo+hi)/2
int c = mid/B + mid/C - mid/LCM
if( c > A){
// goto left
high = mid-1
}
else if( c < A){
// goto right →
low = mid+1
}
else{
ans = mid
high = mid-1
}
}
return ans;

B = 5, C = 7 A = 3
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
0 0 0 0 1 1 2 2 2 3 3 3 3 4 5

Hint for optimization
Given a, calculate how many
magical numbers are there
from [1-n]. B, C

a = 100, B = 2, C = 3
gcd(2,3) = 1
LCM(2,3) = 6

[100/2 + 100/3 - 100/6]

// Generalize
X, B, C
[X/B] + [X/C] - [X/LCM(B,C)]

a = 100 B = 10, C = 15
[100/10] + [100/15] - [100/LCM(10, 15)]
⇒ LCM[10, 15] ≥ 30
All 30 multiples are
counted twice

→ Recursive Binary Search

```
bool BinarySR(int arr[], int lo, int hi, int k){
    // Assumption: check if an element is present
    in range [lo-hi] return T/F
    if( lo > hi) { return False;}
    int mid = (lo+hi)/2
    if( arr[mid] == k) return True;
    if( arr[mid] > k) {
        return BinarySR(arr, lo, mid-1, k) → (N/2)
    }
    else{
        return BinarySR(arr, mid+1, hi, k)
    }
}
```

$T(N) = T(N/2) + 1$
Recursive Relation of BS
$T(N) = \log_2^N$
Solve using Master's Theorem

O(1)

Family of Strings: → Boy and girl story

$S_0$ = 0 : 1, $2^1-1$ → [final year]
$S_1$ = 001 : 3, $2^2-1$
$S_2$ = 0011100 : 7, $2^3-1$
$S_3$ = 0011100110011001 : 15, $2^4-1$

$S_i = S_{i-1} + 0 + (S_{i-1})^c$ → complement

Given N & K find $K^{th}$ character in $S_N$
0 based index

BF: Construct $S_N$ & get $K^{th}$ character

1 ≤ N ≤ 50
$S_N$ : (length of string)
: $2^{N+1}-1$
: $2^{51}-1$
We cannot store these
many characters

K = 4 ans = 0 mid
0 14 → 7 > 4 goto left
0 6 3 < 4 goto right ans = mid ans = 1
4 6 5 > 4 goto left
4 → 4 4 > 4 [return ans] 1

S3
S2
S1
S0
0
1
0
1
0
1