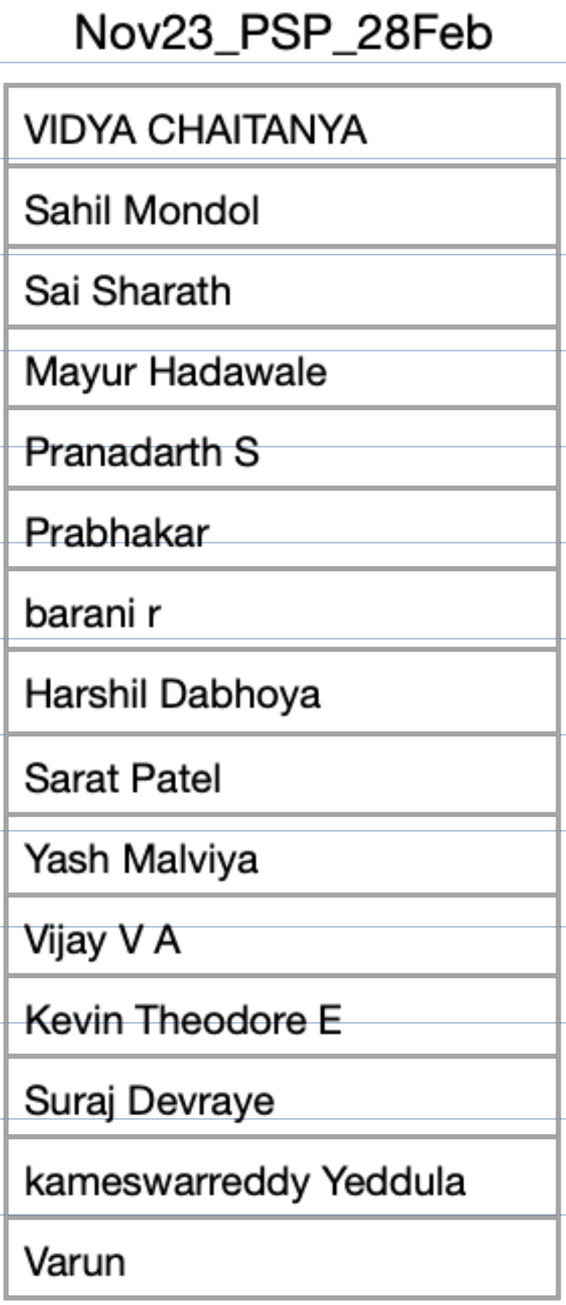
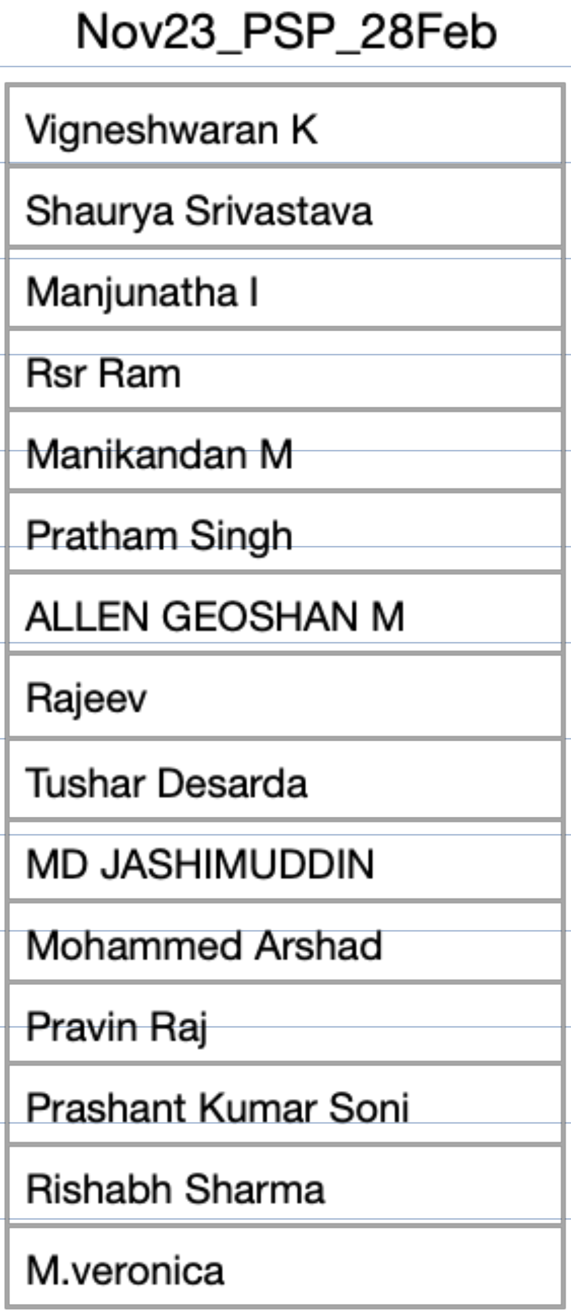
Batch PSP 57.78 → 62



Nov23_PSP_28Feb
VIDYA CHAITANYA
Sahil Mondol
Sai Sharath
Mayur Hadawale
Pranadarth S
Prabhakar
barani r
Harshil Dabhoya
Sarat Patel
Yash Malviya
Vijay V A
Kevin Theodore E
Suraj Devraye
kameswarreddy Yeddula
Varun



Nov23_PSP_28Feb
Vigneshwaran K
Shaurya Srivastava
Manjunatha I
Rsr Ram
Manikandan M
Pratham Singh
ALLEN GEOSHAN M
Rajeev
Tushar Desarda
MD JASHIMUDDIN
Mohammed Arshad
Pravin Raj
Prashant Kumar Soni
Rishabh Sharma
M.veronica

Agenda

searching in sorted rotated array

floor (sqrt (a))

Ath magic number

Median of two sorted array

Basics

Binary Search

Search space

Range in which we are performing our search

WHERE

Target

Key of Search

WHAT

Conditions

1. For answer
2. To eliminate left half
3. To eliminate right half

HOW

Question

Finding an element in a rotated sorted array

Rotated sorted array

arr =

1	2	3	4	5	6	7	8

Rotated k times

①	8	1	2	3	4	5	6	7
②	7	8	1	2	3	4	5	6
③	6	7	8	1	2	3	4	5

Brute force

Do a linear Search using single for loop

T.C = O(n)   S.C = O(1)

Optimized Approach

arr =	1	2	3	4	5	6	7	8

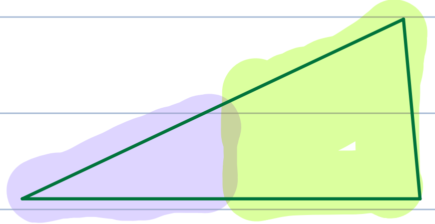
①	8	1	2	3	4	5	6	7
②	7	8	1	2	3	4	5	6
③	6	7	8	1	2	3	4	5

Observation 1

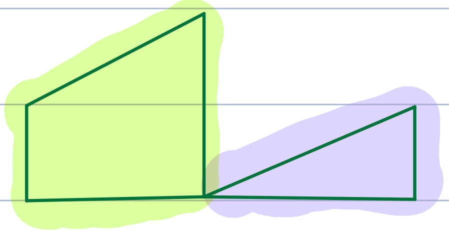
in a rotated array we have 2 individual sorted subarray

Observation 2

To identify the section of array we can use A[0] element



```
if (k >= A[0])
        // 1st part
else
        // 2nd part .
```



Can you find the element if I give the no. of times array is rotated ?

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

① Identify which part of array the key is located

② . Apply Binary search on that subarray by modifying start & end index

Give me solution in 1 BS

10	20	30	1	2	3	4	5	6	7	8	9
0	1	2	3	4	5	6	7	8	9	10	11

part 1 part 2

Overall Binary search

search space : The entire array

target : Key , value asked by user

conditions :

① Identify which part of array is my user input part of

a) condition for ans a[mid] == k return mid

b) eliminate left

Dry Run

0	1	2	3	4	5	6	7	8	9	10	11
10	20	30	1	2	3	4	5	6	7	8	9

Search for 20

part 1 part 2

target in First Half = True

S	E	MID	a[mid]	mid on which part	same part	condition
0	11	5	3	2nd SA	✗	e= m-1
0	4	2	30	1st SA	✓	e = m-1
0	1	0	10	1st SA	✓	s = m+1
1	1	1	20			

pseudo code

```
int search array (int [] arr, int t, int n){

    int s = 0 , e = n-1;
    // check if target is in part 1
    boolean target in FP = true;
    if (t < a[0])
        target in FP = false;

    // perform Binary Search
     while (s <= e) {
         mid = s + (e-s)/2;
         if (arr[mid] == t)  return mid;
         // check if mid & target are in
         // same subarray
         if (target in FP) {
             if ( arr [mid] >= a[0]) {
                 // Basic BS
                 if (arr[mid] > t)  e = mid -1
                 else  s = mid + 1;
             }
```

```
                else  e = mid - 1;
            }
            else {
                if (arr [mid] >= a[0])
                    s = mid + 1;
                else {
                    if (arr [mid] > target) {
                        e = mid - 1;
                    }
                    else  s = mid + 1;
                }
            }
        }
    }
```

T.C = O(log n)

S.C = O(1)

Question

Given an integer n , find floor (sqrt (n)) .

example

n	floor ($\sqrt{n}$)
9	3
12	3
17	4

Brute force

(n=50)

i	i*i	ans
1	1	1
2	4	2
3	9	3
⋮	⋮	⋮
7	49	7
8	64	return 7

T.C = $O(\sqrt{n})$

Binary Search

Search space 1 to n

Because my sqrt (n) value will always be between 1 to N

target : sqrt (n)

condition

for ans :

visualize

if (mid * mid == n)

only works for perfect square

if (mid * mid <= N) → potential ans

s = mid + 1

if (mid * mid > N) e = mid - 1

Dry Run N = 50

ans = ~~1~~ ~~6~~ 7

s	e	mid	mid * mid	where next?
1	50	25	625	625 > 50 L
1	24	12	144	144 > 50 L
1	11	6	36	36 <= 50 R
7	11	9	81	81 > 50 L
7	8	7	49	49 <= 50 R
8	8	8	64	64 > 50 L

8 7

pseudo code

```
int getsqrt ( int n) {
    int s=1 ; e = n; ans = 1;
    while (s <=e) {
        mid = s + (e-s) /2 ;
        if (mid * mid <= n)
            ans = mid ; s = mid +1 ;
        else
            e = mid -1;
    }
    return ans;
}
```

Break

10:25 pm

T.C = O(log n)   S.C = O(1)

LCM (Lowest Common Multiple)

lcm (a, b) = smallest number divisible by both a and b

property

lcm (a, b) * gcd (a, b) = a * b

Question

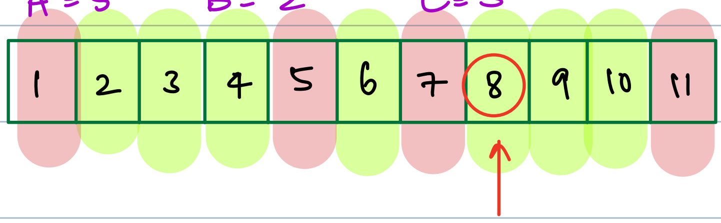
Given A, B and C. Find Ath magic number

Magic number is defined as

Any number that is divisible by B or C or both

example

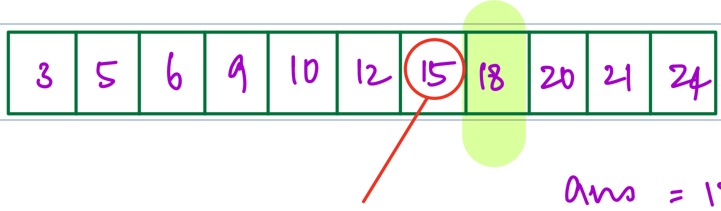
A = 5 B = 2 C = 3



1	2	3	4	5	6	7	8	9	10	11

5th magic no.

A = 8 , B = 3 , C = 5 , LCM (3, 5) = 15



3	5	6	9	10	12	15	18	20	21	24

ans = 18

only counted once

Binary search

Search space :

my search space is between

[1, min(B, C) * A]

targetting : A^{th} magic number

CALCULATING MAGIC NUMBER

$A = 8$ $B = 3$ $C = 5$ assume

mid = 25

multiples of 3 in mid + multiples of 5 in mid - multiples of LCM

$$\text{position} = \frac{25}{3} + \frac{25}{5} - \frac{25}{15}$$

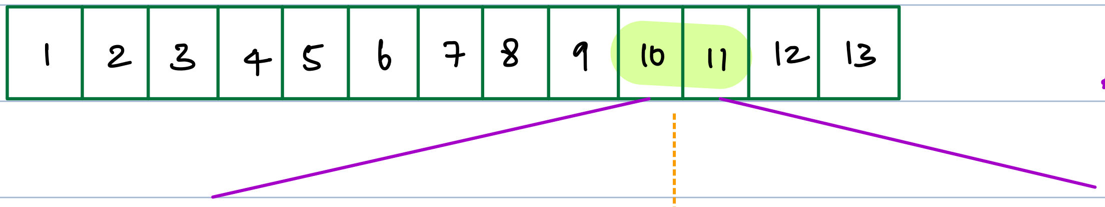
$$= 8 + 5 - 1$$

$$= 12$$

Visualize

$B = 3 \quad C = 5$

$A = 5$



$$\text{position} = \frac{10}{3} + \frac{10}{5} - \frac{10}{15}$$

$$= 3 + 2 - 0 = 5$$

$$\text{position} = \frac{11}{3} + \frac{11}{5} - \frac{11}{15}$$

$$= 3 + 2 - 0 = 5$$

many numbers can have the same position

However, we want the smallest among them

Binary search condition

$s = 1 \qquad e = \min(B, C) * A$

$LCM = B * C \,/\, GCD(B, C)$

$$pos = \frac{mid}{B} + \frac{mid}{C} - \frac{mid}{LCM}$$

```
if (pos < A) {
    // move right
    s = mid + 1
}

else {
    ans = mid;
    e = mid - 1;
}
```

A = 8 , B = 3, C = 5 LCM = 15 ans = max(B, C)

~~5~~ 18

s	e	mid	no. of mul	where to go
1	24	12	$12/3 + 12/5 - 12/15$	6th move Right
13	24	18	$18/3 + 18/5 - 18/15$	8th move left
13	17	15	$15/3 + 15/5 - 15/15$	7th move right
16	17	16	$16/3 + 16/5 - 16/15$	7th move right
17	17	17	$17/3 + 17/5 - 17/15$	7th move right
18	17			

T.C = $O(\lg(\min(B, C) * A))$

S.C = $O(1)$

Question

Given two sorted array, find the median of merged array .

median : Given sorted array , median is middle element

A =

1	3	4	7	10	12

B =

2	3	6	15

Merged array

C =

1	2	3	3	4	6	7	10	12	15
0	1	2	3	4	5	6	7	8	9

$$\text{median} = \frac{4+6}{2} = 5$$

Brute force

Merge two sorted arrays O(n+m)

Find the middle elements

calculate median

T.C = O(n+m)   S.C = O(n+m)

Binary Search

We are calculating median (middle part)

C =	1	2	3	3	4	6	7	10	12	15
	0	1	2	3	4	5	6	7	8	9

part 1 part 2

Observations

Both parts have some elements from array A and some elements from Array B

Observation

Elements in part 1 will always be lesser than elements in part 2

TRIAL AND ERROR

A =	1	3	4	7	10	12
B =	2	3	6	15		

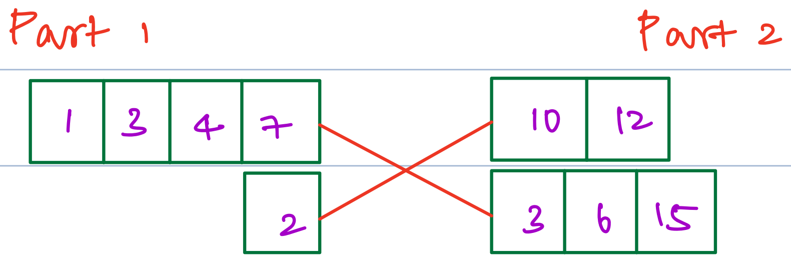
Try 1

let me pick 4 elements from A in part 1

→ 1 element from B in part 1

→ 2 element from A in part 2

→ 3 element from B in part 2



2 < 10 However 7 > 3

↑ Red flag

we should select less elements from A in part 1

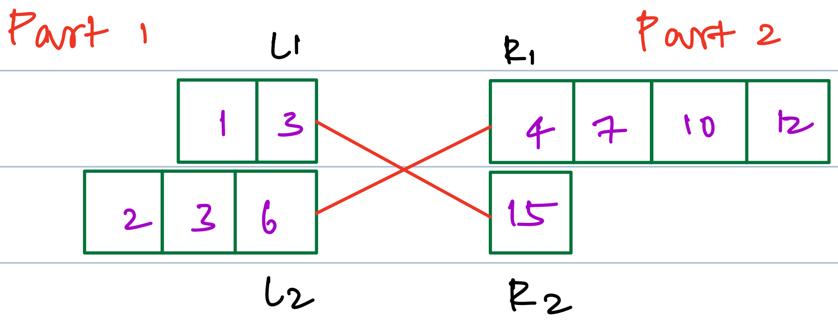
Try 2

let me pick 2 elements from A in part 1

→ 3 element from B in part 1

→ 4 element from A in part 2

→ 1 element from B in part 2



$L_1 < R_2$ $(3 < 15)$ $L_2 < R_1$ $(6 < 4)$

Select more from Array A

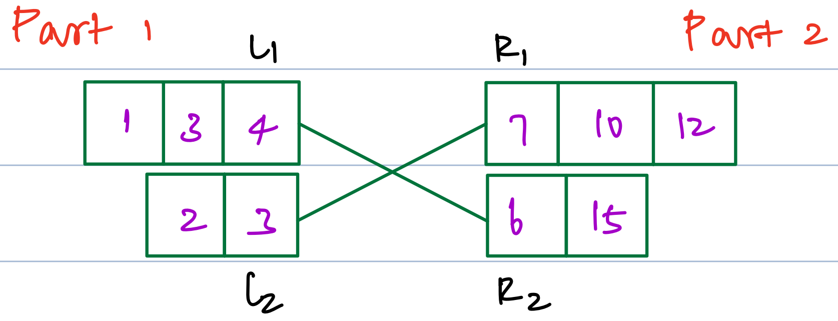
Try 3

let me pick 3 elements from A in part 1

→ 2 element from B in part 1

→ 3 element from A in part 2

→ 2 element from B in part 2



$4 < 6$ && $3 < 7$

$L_1 < R_2$ $L_2 < R_1$

$$\text{median} = \frac{\max(L_1, L_2) + \min(R_1, R_2)}{2}$$

Binary Search

Search space : Both array A & B

target : median of merged array

conditions :

To be continued in next class