1 Doubts with Lakshay Bhaiya [week 4]

Swap implement → (a, b) → swap

Three methods.

1. Temp variable.
2. (+, −) Arithmetic
3. XOR.

Arithmetic :-

int a, int b

1. a = a+b
2. b = a−b
   ↓
   5−3 = 2
   b = 2
3. a = a−b
   = 5−2
   a = 3

2	a
3	b

(i) a = a+b
(ii) b = a−b
(iii) a = a−b

XOR :-
(fastest)

int a, int b
a ↓ 2, b ↓ 3

(i) a = a^b
(ii) b = a^b
(iii) a = a^b

a = a^b = 2^3
b = a^b = 2^3^3 = 2
a = a^b = 2^3^2 = 3

Rotate Array

1. Extra space. ✓
2. Single rotate · K times · ✓
3. Modulo ((i+K)%n) → O(n) ✓
4. Reversal Method → O(n)

4. Reversal Method

K = 3

I/P → 1|2|3|4|5|6|7

↓

7|6|5|4|3|2|1

5|6|7|4|3|2|1

↓

O/P → 5|6|7|1|2|3|4

Algo

1. Reverse whole Array. Rev(0, n-1)
2. Rev(0, K-1)
3. Rev(K, n-1)

Code.

K = K % nums.size ← what if K = nums.size() or K > nums.size().

```
reverse(nums.begin(), nums.end());
reverse(nums.begin(), nums.begin()+K);
reverse(nums.begin()+K, nums.end());
```

→ Reverse (STL)

1|2|3|4|5|7

↓ nums.begin() = 0 ⟶ nums.end() = n-1

Searching and Sorting - Level 2

→ find pivot Element
↓
Subquestion.

[Search in a rotated & Sorted Array]
↓
question

| 2 | 4 | 6 | 8 | 10 | 12 | 14 | 16 |

Sorted ← Monotonic →

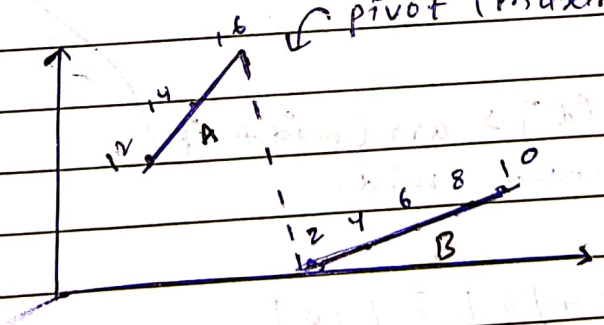
| 12 | 14 | 16 | 2 | 4 | 6 | 8 | 10 |

Rotated and Sorted Array

→ decreasing

12	14	16	2	4	6	8	10
0	1	2	3	4	5	6	7

find pivot (maximum no.)



Approach-1
→ Linear search. O(n)

Approach - 2
→ sort O(nlogn)

#3 Approach -

Binary search (olog n)

| 12 | 14 | 16 | 2 | 4 | 6 | 8 | 10 |

(A) (B)

↳ seperately handle.

| 12 | 14 | 16 | 2 | 4 | 6 | 8 | 10 |

↑ ↓

aus mid

arr[mid] < arr[mid-1]

return mid-1 (index)

ye wala condition sirf inhi do element pe lagta hai

↗ aus

| 12 | 14 | 16 | 2 | 4 | 6 | 8 | 10 |

↑

mid

arr[mid] > arr[mid+1]

return mid.

| 12 | 14 | 16 | 2 | 4 | 6 | 8 | 10 |

(A) (B)

↳ Agar me is line p hota to mai pehla 12 k upar hota.

arr[start] > arr[mid]

↳ left

e = mid-1;

else s = mid + 1 → Right me jayenge.

Hume pivot index mil gaya hai aur ab hum array me search karenge.

| 12 | 14 | 16 | 2 | 4 | 6 | 8 | 10 |

(A) (Binary search) (B) (Binary search)

target = 14

Agar tumahara target 12 se 16 ke beech me aata hai to (A) me binary search laga lo

Agar tumhara target 2 se 10 ke beech me aata hai to (B) me binary search laga lo.

~~Edge~~ Edge case.

[1, 3] → Isme code fatega.

↓

mid arr[mid] < arr[mid-1]

↙

that's why we take cond^n invalid index.

if (mid < n-1)
if (mid > 0) } on pivot element function.

Code

```
// finding pivot Element.
int pivotIndex(vector<int>& nums) {
    int n = nums.size();
    int s = 0;
    int e = n-1;
    while (s<=e) {
     if (s==e) return s;
    // corner case if the size of array is 1.
     int mid = (s+e)/2;
     if (mid<n-1 && nums[mid] >nums[mid+1]
            return mid;
    else if (mid>0 && nums[mid] < nums[mid-1]
            return mid;
    else if (nums[mid] <nums [s] )
            e = mid-1;
    else s = mid+1;
    }
return -1;

// Binary search for finding target.

int binarySearch(vector<int>&nums, int s, int e, int target)
{
    while (s<=e)
    {
        int mid = s+(e-s)/2;
        if (nums[mid] = target) return mid;
        else if (target >nums[mid]  s = mid+1;
        else   e = mid-1;
    }
return -1;
```

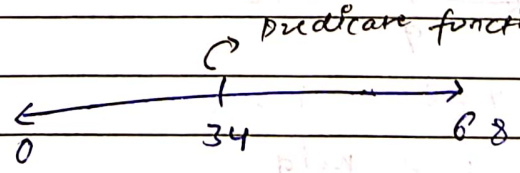
```
// searching in sorted Array.
int search (vector<int>& nums, int target)
{
    int n = nums.size()-1;
    int pivotRes = pivotIndex(nums);
    if (target >= nums[0] && target <= nums[pivotRes])
    {
        return binarySearch(nums, 0, pivotRes, target);
    }
    else
    {
        return binarySearch(nums, pivotRes+1, n, target);
    }
return -1;
}
```

Q. SQRT(x) (without using STL function)

i/p → x

o/p → SQRT(x) → $\sqrt{x}$ ans.

eg: x = 68 o/p $\sqrt{68}$ → 8 (nearly)



steps

① search space

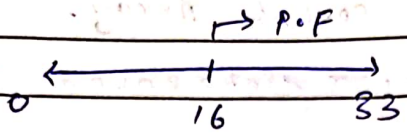
② predicate function

s = 0
e = 68
} mid = 34

↳ 34 x 34 = 68 → false

34 x 34 > 68 → left me jaega.

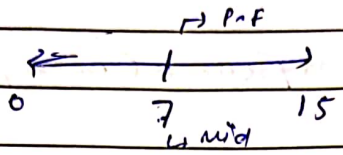
e = 34 - 1 = 33



mid = 16

16 × 16 = 256 > 68

e = 16 - 1 = 15

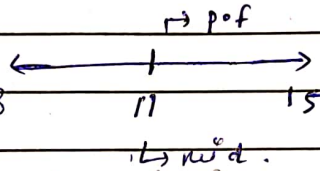


7 × 7 = 49 < 68

ans = 7 ↳ store kr lenge ans

right me jayenge.

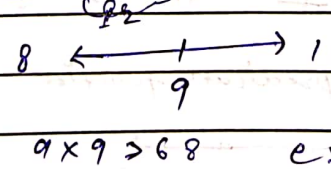
s = mid + 1 = 7 + 1 = 8



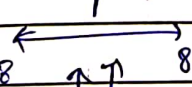
11 × 11 = 121 > 68

e = mid - 1

e = 15 - 1 = 10



9 × 9 > 68 e = 8



start → end.

8 × 8 < 68

ans = 8

classmate
Date
Page 210

Steps

1. Search space
2. Predicate function → True / false
3. Store ans.

Code

```
int mySqrt(int x){
    int s = 0;
    int e = x;
    int ans = -1;
    while(s<=e){
        long long int mid = s+(e-s)/2;
        Long Long int sqrt = mid * mid;
        if(x==sqrt) return mid;
        else if(sqrt>x) e = mid-1;
        else{
            ans = mid;
            s = mid+1;
        }
    }
    return ans;
}
```

Time Complexity = O(logn)
Space complexity = O(1)