DSA SERIES

- Learn Coding

Topic to be Covered today

Binary Search Mixed Problems

LETS START TODAY’S LECTURE

2616. Minimize the Maximum Difference of Pairs

```
class Solution {
public:
    bool isValid(vector<int>& nums, int mid, int p) {
        int i = 0;
        int pairs = 0;
        int n = nums.size();

        while (i < n - 1) {
            if (nums[i + 1] - nums[i] <= mid) {
                pairs++;
                i += 2;
            } else {
                i++;
            }
        }

        return pairs >= p;
    }
    int minimizeMax(vector<int>& nums, int p) {
        int n = nums.size();
        sort(begin(nums), end(nums));
        int left = 0;
        int right = nums[n - 1] - nums[0];

        int result = INT_MAX;

        while (left <= right) {
            int mid = left + (right - left) / 2;

            if (isValid(nums, mid, p)) {
                result = mid;
                right = mid - 1;
            } else {
                left = mid + 1;
            }
        }

        return result;
    }
};
```

153. Find Minimum in Rotated Sorted Array

```
class Solution {
public:

    int findMin(vector<int>& nums) {
     int n = nums.size();
     int left = 0;
     int right = n-1;

     while(left<right){
        int mid = left + (right-left)/2;

        if(nums[mid]>nums[right]){
            left = mid +1;
        } else{
            right = mid;
        }
     }
     return nums[left];
    }
};
```

33. Search in Rotated Sorted Array

```
class Solution {
public:
    int FindPivot(vector<int>& nums, int l, int r) {
        while (l < r) {
            int mid = l + (r - l) / 2;
            if (nums[mid] > nums[r]) {
                l = mid + 1;
            } else {
                r = mid;
            }
        }

        return r;
    }

    int binarySearch(vector<int>& nums, int left, int right, int target) {
        while (left <= right) {
            int mid = left + (right - left) / 2;

            if (nums[mid] == target) {
                return mid;
        } else if (nums[mid] > target) {
            right = mid - 1;
        } else {
            left = mid + 1;
        }
    }
    return -1;
}

int search(vector<int>& nums, int target) {
    int n = nums.size();

    int pivot = FindPivot(nums, 0, n - 1);

    if (nums[pivot] == target) {
        return pivot;
    }

    // Find the element in the right part and the left part respectively

    int idx = -1;
```

```
        idx = binarySearch(nums, pivot + 1, n - 1, target);

        if (idx != -1) {
            return idx;
        }

        idx = binarySearch(nums, 0, pivot - 1, target);

        return idx;
    }
};
```

81. Search in Rotated Sorted Array II

```
class Solution {
public:
int FindPivot(vector<int> &nums, int l ,int r){
    while(l<r){
        while(l<r && nums[l]==nums[l+1])
        l++;

        while(r>l && nums[r]==nums[r-1])
        r--;

        int mid = l + (r-l)/2;

        if(nums[mid]<=nums[r]){
            r=mid;
        } else{
            l=mid+1;
        }

    }

    return r;
;
}
bool binarySearch(vector<int>& nums, int left, int right, int target) {
        while (left <= right) {
            int mid = left + (right - left) / 2;

            if (nums[mid] == target) {
                return true;
            } else if (nums[mid] > target) {
                right = mid - 1;
            } else {
                left = mid + 1;
            }
        }
        return false;
    }
bool search(vector<int>& nums, int target) {
        int n = nums.size();
        int pivot = FindPivot(nums,0,n-1);

        if(binarySearch(nums,0,pivot-1,target)){
            return true;
        }
        return binarySearch(nums,pivot,n-1,target);

    }
}
```

162. Find Peak Element

```
class Solution {
public:
    int findPeakElement(vector<int>& nums) {
        int s = 0,  e = nums.size() - 1;

        while (s < e) {
            int mid = s + (e - s) / 2;

            if (nums[mid] > nums[mid + 1]) {
                e = mid;
            }
            else{
                s = mid+1;
            }
        }

        return s;
    }
};
```

Learn coding

THANK YOU