Day-5 06-01-2021

Level: Easy

Q-1 Remove duplicates from array.

Input: [1,1,2]
output: [1,2]

// Check nums[i] != nums[i-1]
// then nums[k++] = nums[i]
// return K.

```
int removeDuplicates(vector<int>& nums)
 {
  int n = nums.size();
  if (n==0)
    return 0;
  int k=1;
  for(int i=1;i<n;i++)
    {
      if (nums[i] != nums[i-1])
      {
        nums[k++] = nums[i];
      }
  } return k;
}
```

Level: Medium

Q-2: find peak element
input: [1,2,3,1]
Output: 2 //return index

// Check n==1; return 0;
// Check i==0 && nums[i] > nums[i+1] → return i

// check i==n-1 && num[i] > nums[i-1] → return i

if (i!=0 and i!=n-1 and nums[i] > num[i-1] and
num[i] > num[i+1]
return i
// return -1;

Extra Que } Do later // {

4Sum using recursion }

Recursi

Level-Medium

Q-3 find first and last position of element in Sorted array.

```
// using binary search.          // find firstPosition
// find mid.                     // find secondPosition

vector<int>searchRange(vector<int>&nums, int target)
{
    vector<int> res;
    int firstIndex = firstPosition(nums, target);
    int lastIndex = lastPosition(nums, target);
    res.push_back(firstIndex);
    res.push_back(lastIndex);
    return res;
}

int firstPosition(vector<int>& nums, int target)
{
    int mid;
    int left=0;
    int right = nums.size()-1;
    int position=-1;

    while(left<=right)
    {
        mid = (left+right)/2;
        if(nums[mid]==target)
        {
            position = mid;
            right = mid-1;
        }
```

```
else if (nums[mid] < target)
{
    left = mid+1;
}
else {
    right = mid-1;
}
}
    return position;
}
int lastPosition(vector<int>& nums, int target)
{
    int mid;
    int left;
    int right = nums.size()-1;
    int position = -1;

    while (left <= right)
{
    mid = (left + right) / 2;
    if (nums[mid] == target)
{
    if position = mid;
        left = mid +1;
}
    else if (nums [mid] < target)
{
    left = mid +1;
}
    else
    { right = mid -1;
    }
}  return position;
}
```