WEEKELY CODING CHALLENGES
B. Tech 3rd Year CSE
Season: Winter 2023-2024

Kapil Singh

2100290100082

WEEK – 1

1. Subarray Sum Equals K (560)

```
lass Solution {
public:
    int subarraySum(vector<int>& nums, int k) {
        unordered_map<int,int> mp;
        mp[0]=1;
        int sum=0,ans=0;
        for(auto i : nums){
            sum+=i;
            if(mp[sum-k]>0)  ans+=mp[sum-k];
            mp[sum]++;
        }
        return ans;
    }
};
```

2. Majority Element II (229)

```
class Solution {
public:
    vector<int> majorityElement(vector<int>& nums) {
        vector<int> result;
        int candidate1 = 0, candidate2 = 1, count1 = 0, count2 = 0;

        for (int num : nums) {
            if (num == candidate1) count1++;
            else if (num == candidate2) count2++;
            else if (count1 == 0) candidate1 = num, count1 = 1;
```

```
            else if (count2 == 0) candidate2 = num, count2 = 1;
            else count1--, count2--;
        }

        count1 = count2 = 0;

        for (int num : nums) {
            if (num == candidate1) count1++;
            if (num == candidate2) count2++;
        }

        if (count1 > nums.size() / 3) result.push_back(candidate1);
        if (count2 > nums.size() / 3) result.push_back(candidate2);

        return result;
    }
};
```

3. 3Sum (15)

```
class Solution {
public:
    vector<vector<int>> threeSum(vector<int>& nums) {
        int n=nums.size();
        vector<vector<int>> ans;
        sort(nums.begin(),nums.end());
        // return {nums};
        n=nums.size();
        for(int i=0;i<n-2;++i){
            if(nums[i]>0) break;
            if(i>0 && nums[i]==nums[i-1]) continue;
            int j=i+1,k=n-1;
            while(j<k){
                int sum=nums[i]+nums[j]+nums[k];
                if(sum==0) {
                    ans.push_back({nums[i],nums[j],nums[k]});
                    int t=nums[k];
                    while(k>j && nums[k]==t) k--;
                }
                else if(sum>0) k--;
                else  j++;
            }
        }
        return ans;
    }
};
```

4. Merge Intervals (56)

```
class Solution {
public:
    static bool cmp(vector<int> &a,vector<int> &b){
        if(a[0]!=b[0]) return a[0]<b[0];
        return a[1]<b[1];
    }
    vector<vector<int>> merge(vector<vector<int>>& inter) {
        sort(inter.begin(),inter.end(),cmp);
        vector<vector<int>> res;
        int l=inter.size();
        res.emplace_back(inter[0]);
        int j=0;
        for(int i=1;i<l;i++){
            if(res[j][1]>=inter[i][0]){
                if(res[j][1]>=inter[i][1])  ;
                else
                res[j][1]=inter[i][1];
            }
            else{
                res.emplace_back(inter[i]);
                j++;
            }
        }
        return res;
    }
};
```

5. Reverse Pairs (493)

```
class Solution {
public:
    //merge two sorted logic:
  void merge(vector<int> &nums, int l, int m, int r){
      int count = 0;

      //copy into right & left array
      vector<int>left(nums.begin()+l,nums.begin()+m+1);
      vector<int>right(nums.begin()+m+1,nums.begin()+r+1);

      int i = 0, j = 0, n1 = left.size(),n2 = right.size();

      while(i < n1 && j < n2){
          if (left[i] <= right[j])
              nums[l++] = left[i++];
          else{
            nums[l++] = right[j++];
          }
```

```
        }
        while (i < n1)
            nums[l++] = left[i++];
        while (j < n2)
            nums[l++] = right[j++];
    }
    //count pair
    int countPairs(vector<int>&nums,int l, int m, int r){
        int count = 0;
        for(int i = l, j = m+1; i <= m; i++){
            while(j <= r && nums[i] > 2LL*nums[j]) j++;
            count += j-(m+1);
        }
        return count;
    }
    int mergeSort(vector<int> &nums, int l, int r){
        int count = 0;
        if (l == r) return count;

        int m = l + (r-l)/2;
        //left part sort
        count += mergeSort(nums,l,m);
        //right part sort
        count += mergeSort(nums,m+1,r);
        //count pairs
        count += countPairs(nums,l,m,r);
        //merge these sorted array
        merge(nums,l,m,r);
        return count;
    }
      int reversePairs(vector<int>& nums) {
          return mergeSort(nums,0,nums.size()-1);
      }
};
```

58. Maximum Product Subarray (152)

```
class Solution {
public:
    int maxProduct(vector<int>& nums) {
        int maxi = INT_MIN;
        int prod=1;

        for(int i=0;i<nums.size();i++)
        {
          prod*=nums[i];
          maxi=max(prod,maxi);
          if(prod==0)
           prod=1;
        }
        prod=1;
```

```
        for(int i=nums.size()-1;i>=0;i--)
        {
          prod*=nums[i];

          maxi=max(prod,maxi);
          if(prod==0)
           prod=1;
        }
        return maxi;
    }
};
```