**Day-3**

**Problem1**- [**Minimum Moves to Equal Array Elements II**](https://leetcode.com/problems/minimum-moves-to-equal-array-elements-ii/)

Solution-

class Solution {

public:

    int minMoves2(vector<int>& nums) {

        sort(nums.begin(),nums.end());

        int n=nums.size();

        int mid=(n/2);

        cout<<mid;

       int ans=0;

        for(int i=0;i<n;i++)

        {

            ans+=abs(nums[i]-nums[mid]);

        }

        return ans;

    }

};

**Problem2**- **3 Sum**

Solution-

class Solution {

public:

    vector<vector<int>> threeSum(vector<int>& a) {int n=a.size();

        vector<vector<int>>ans;

        int sum=0;

        sort(a.begin(),a.end());

        for(int i=0;i<n;i++ )

        {

          if(i>0 && a[i]==a[i-1] )continue;

          int j=i+1;

          int k=n-1;

          while(j<k)

          {

            sum=a[i]+a[j]+a[k];

            if(sum>0)

            {

              k--;

            }

            else if(sum<0)

            {

              j++;

            }

            else

            {

               vector<int>temp={a[i],a[j],a[k]};

               ans.push\_back(temp);

                j++;

               k--;

               while(j<k && a[j]==a[j-1]) j++;

               while(j<k && a[k]==a[k+1]) k--;

            }

          }

        }

        return ans;

    }

};

**Problem 3**- [**442. Find All Duplicates in an Array**](https://leetcode.com/problems/find-all-duplicates-in-an-array/)

class Solution {

public:

    vector<int> findDuplicates(vector<int>& nums) {

        vector<int>lookup(100000,0);

        int n = nums.size();

        for(int i=0;i<n;i++){

            lookup[nums[i]]++;

        }

        vector<int> ans;

        for(int i=1 ; i <=n ; i++){

            if(lookup[i]==2){

                ans.push\_back(i);

            }

        }

        return ans;

    }

};

**Problem 4**- [**Subarray Sums Divisible by K**](https://leetcode.com/problems/subarray-sums-divisible-by-k/)

class Solution {

public:

    vector<vector<int>> threeSum(vector<int>& a) {int n=a.size();

        vector<vector<int>>ans;

        int sum=0;

        sort(a.begin(),a.end());

        for(int i=0;i<n;i++ )

        {

          if(i>0 && a[i]==a[i-1] )continue;

          int j=i+1;

          int k=n-1;

          while(j<k)

          {

            sum=a[i]+a[j]+a[k];

            if(sum>0)

            {

              k--;

            }

            else if(sum<0)

            {

              j++;

            }

            else

            {

               vector<int>temp={a[i],a[j],a[k]};

               ans.push\_back(temp);

                j++;

               k--;

               while(j<k && a[j]==a[j-1]) j++;

               while(j<k && a[k]==a[k+1]) k--;

            }

          }

        }

        return ans;

    }

};