```
vector<int> ans;
int n = nums.size();
unordered_map<int,int> mp;

for(int i=0; i<n; i++){
    mp[nums[i]]++;
}
int target = n/3;
for(int i=0; i<n; i++){
    if(mp[i] > target){
        ans.push_back(nums[i]);
    }
    ~~return ans~~
}
return ans;
```

(38,9)
(13,8)
(-4,6)
(-5,5)
(5,3)
(3,2)
(1,0)

(key, value)
(PS index)

3   Sum = ~~x~~ ~~x~~ 3

1 -1 3 2 -2 -8 1 7 10 23

maxi = ~~x~~ ~~2~~ 5

(3,4)
(3,7)

Merge sorted array

arr1 = {1,3,5,7,9}
arr2 = {0,2,6,8}

```
vector<int> arr3 ;
~~int~~ int i=0, j=0
~~for(i=0; i< arr1.size(); i++)~~
        ~~n1 -> 1st size~~
        n2 -> 2nd size
~~while(i<n1 && j<n2){~~
    ~~if(arr[j] > arr[i]){~~
    ~~ans.pu~~


while(i<n1 && j<n2){
    if(arr1[i] < arr2[j]){
        ans.push_back(arr1[i]);
        i++;
    }
    else if(arr1[i] > arr2[j]){
        ans.push_back(arr2[j]);
        j++
    }
```

x ~> 1st element of current
y ~> 2nd element of previous

~~*~~ x > y
~~[illegible]~~

~~for(i=0; i<n; [illegible]){~~
~~if~~

```
arr[]={3,1,-2,-5,2,-4};
                                  ANS→ {3,-2,1,5,2,-4}
```

~~int pnt [illegible]~~
~~for i.~~

```
    [illegible] pos ;
    [illegible] neg ;
    ~~int pnt = arr[0];~~


                                  for(i=0 → n-1){
int k=0; l=0 ;
int pos [n/2] ;
int neg [n/2] ;
~~int [illegible]~~
~~int [illegible]~~
for(int i=0 ; i<n ; i++){
    if (arr[i] >0){
        pos[k] = arr[i] ;
        k++ ;
    }
    else if (arr[i] <0){
        neg[k] = arr[i] ;
        l++ ;
    }
}
```

$^{n}P_{r}$

```
        vector <int> leaders ;
        ~~int leaders~~
        for(int i=0 ; i < n-1 ; i++){
            for(int j=i+1 ; j< n-1 ; j++){
                if(A[i] > A[j]){
                    leaders.push_back(A[i]);
                }
            }
            return leaders
        }
        ~~leaders.push_back(A[n-1]);~~
```

Date

```
arr1 = {1,3,5,7,8}
arr2 = {0,2,6,9}


vector<int> ans;
int i=0,j=0
while(i< arr1.size() & j < arr2.size()){
    if(arr1[i] < arr2[j]){
        ans.push_back(arr1[i]);
        i++;
    }
    ~~else if (arr1[i] > arr2[j]){~~
        ans.push_back(arr2[j]);
        j++;
    }
}
while(i< arr1.size
```

```
arr ——> {(1,3)(2,4)(2,6).(8,10),(8,9),(15,18),(16,17)}
    ~~vector <int> ans;~~
    Sort(arr.begin(); arr.end());
    for(int i=0; i<n; i++){
        if( ans.empty() || arr[i][0] > ans.back()[1]){
            ans.push_back(arr[i]);
        }
        else{
            ans.push_back = max(ans.back()[1], arr[i][1]);
        }
    }
    return ans;
```

```
n = nums.size();
vector<int> ans;
sort(nums.begin(), nums.end());

for(int i = 0; i<n; i++){

    if(i>0 && nums[i]==nums[i-1]){
        continue;
    }
    int j = i+1;
    int 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]});
            j++;
            k--;

            while(j<k && nums[j] == nums[j-1]) j++ [illegible];
            while(j<k && nums[k]==nums[k-1]) k--;
        }
        else if(sum>0){
            k--;
        }
        else{
            j++
        }
    }
}
} ~~return ans~~
return ans;
```

Page No.
Date

```
nums = {1, 2, 3, -3, 4, -2, 2, 1}   k=3
    n=nums.size();
    int maxLen=0, pre=0;
    unordered_map <int, int> mp;

    for(int i=0; i<n; i++){
        pre= pre + nums[i]

        if (pre==k){
            maxLen=i+1;
        }
        else if (mp.find(pre-k) != mp.find()){
            maxLen= max(maxLen, i-mp[pre-k]);
        }
        else{
            mp.find(pre) == mp.end()){
                mp[pre] = i;
        }
    }
    return maxLen;
```

```
nums = {4,3,6,2,1,1}
 n = nums.size();
 vector <int> hash (n+1 ,0);

 for(int i=0; i<n; i++){
     hash[nums[i]]++   // we stored value at
                          every index of
 }                        hash array

int missing = -1 , repeating = -1
for(int i=1; i<=n; i++){
   if(hash[i] == 0){
        missing = i;
   }
   else if(hash[i]>1){
        repeating = i;
   }
}
return {missing, repeating};
```

```
vector<int> ans;
 n = nums.size()                              x -> 1st element of current
                                              y -> 2nd element of previous
 sort(nums.begin(), nums.end());
 for(int i=0; i<n; i++){
     if(ans.empty() || arr[i][0] > ans.back()[1]){
         ans.emplace_back(arr[i]);
     }
     else{
          ans.back()[1] = max(arr[i][1], ans.back()[1]);
          ~~ans.push_back~~
     }
 } return ans;
```

```
nums = {2, -3, 0, -2, -4, -1}
n = nums.size();
[illegible]
int pre = 1, sub = 1, ans = INT_MIN;
for(int i = 0; i<n; i++){
    if (pre==0) pre =1;
    if (sub==0) suf = 1;
    pre = pre* nums[i];
    sub = sub* nums[n-i-1];

    ans = max(ans, max(pre,sub));
}
return ans;
```

```
nums[] = {1, -1, 3, 2, -2, -8, 1, 7, 10, 23}
  int maxi = INT_MIN, Sum=0
  n = nums.size();
 unordered_map<int, int> mp;
 for(int i=0; i<n; i++){
     pre = pre + nums[i];
```

Date

```
nums = {1 ,2, 3, -3, 4, -2, 2, 1}  k=3
    n=nums.size();
    int maxLen=0, pre=0;
    unordered_map <int,int> mp;

    for(int i=0; i<n; i++){
        pre= pre + nums[i]

        if (pre==k){
            maxLen=i+1;
        }
        else if (mp.find(pre-k) != mp.find()){
            maxLen = max(maxLen, i - mp[pre-k]);
        }
        else{
            mp.find(pre) == mp.end()){
                mp[pre] = i;
            }
    }
    return maxLen;
```