```cpp
#include<bits/stdc++.h>
using namespace std;

int divideIntoTwo(vector<int> v){
    int ans = -1;
    int totalSum = 0;
    for(auto i : v){
        totalSum += i;
    }
    int prefixSum = 0;
    for(int i=0; i<v.size(); i++){
        prefixSum += v[i];
        // if(totalSum-prefixSum = prefixSum)
        if(totalSum == 2*prefixSum){
            ans = i;
            return ans;
        }
    }
    return ans;
}

int LargestSum(vector<int> v){
    int maxi = INT_MIN;
    for(int i=0; i<v.size(); i++){
        int prefixSum = 0;
        for(int j=i; j<v.size(); j++){
            prefixSum += v[j];
            maxi = max(maxi,prefixSum);
        }
    }
    return maxi;
}

int LargestSumUsingKadanes(vector<int> v){
    int maxi = INT_MIN, prefixSum=0;
    for(int i=0; i<v.size(); i++){
        prefixSum += v[i];
        if(prefixSum<0){
            prefixSum = 0;
        }
        maxi = max(maxi,prefixSum);
    }
    return maxi;
}


vector<int> greaterThanThree(vector<int> v){
    unordered_map<int,int> mp;
    vector<int> ans;
    int minLimit = v.size()/3 + 1;
    for(auto ele : v){
        if(mp[ele]==minLimit){
            ans.push_back(ele);
        }
        mp[ele]++;
    }
    return ans;
}

vector<int> greaterThanThree1(vector<int> v){
    int minLimit = v.size()/3 + 1;
    int cnt1 = 0,cnt2 = 0, ele1 = INT_MIN, ele2 = INT_MIN;

    for(int i=0; i<v.size(); i++){
        if(cnt1==0 and ele2!=v[i]){
            ele1 = v[i];
            cnt1++;
        }
        else if(cnt2==0 and ele1!=v[i]){
            ele2 = v[i];
            cnt2++;
        }
        else if(v[i]==ele1){
            cnt1++;
        }
        else if(v[i]==ele2){
            cnt2++;
        }
        else{
            cnt1--;
            cnt2--;
        }
    }

    cnt1 = cnt2 = 0;
    for (auto ele : v) {
        if (ele == ele1) cnt1++;
        else if (ele == ele2) cnt2++;
    }
    vector<int> res;
    if (cnt1 > minLimit) res.push_back(ele1);
    if (cnt2 > minLimit) res.push_back(ele2);
    return res;
}


int trapWater(vector<int> h){
    int len = h.size();
    vector<int> leftMax(len,0);
    vector<int> rightMax(len,0);
    for(int i=1; i<len-1; i++){
        leftMax[i] = max(h[i-1],leftMax[i-1]);
    }
    for(int i=len-2; i>=0; i--){
        rightMax[i] = max(h[i+1],rightMax[i+1]);
    }
    int waterCount = 0;
```

```cpp
    for(int i=0; i<len; i++){
        int minHeight = min(leftMax[i],rightMax[i]);
        if(minHeight>h[i]){
            waterCount += minHeight-h[i];
        }
    }
    return waterCount;
}


bool threeSum(vector<int> arr, int x){
    sort(arr.begin(), arr.end());
    for(int i=0; i<arr.size()-2; i++){
        int finalX = x-arr[i];
        int start=i+1, end=arr.size()-1;
        while (start < end) {
            int sum = arr[start] + arr[end];
            if (sum == finalX) {
                return true;
            }
            else if (sum > finalX) {
                end--;
            }
            else {
                start++;
            }
        }
    }
    return false;
}


int main(){
    //divide array in 2 subarray of equal sum
    vector<int> v = {3,4,-2,5,8,20,-10,8};
    int ans = divideIntoTwo(v);
    cout<<"Divide Array into Subarray: "<<ans<<endl;

    //largest sum subarray
    //mtd-1
    vector<int> vec = {3,4,-5,8,-12,7,6,-2};
    int res = LargestSum(vec);
    cout<<"Largest Sum Subarray: "<<res<<endl;

    //mtd-2 - kadane's algo
    vector<int> vec1 = {3,4,9,-5,8,-12,7,6,-2};
    int res1 = LargestSumUsingKadanes(vec1);
    cout<<"Largest Sum Subarray Using Kadane's: "<<res1<<endl;

    vector<int> arr = {2,2,1,2,2,1,2,1,1,1};
    //Element Appearing greater than n/3 times in the given array - Mtd-1
    vector<int> ans1 = greaterThanThree(arr); //Given : There might be multiple element
    for(auto ele : ans1){
        cout<<ele<<" ";
    }cout<<endl;

    //Element Appearing greater than n/3 times in the given array - Mtd-2
    vector<int> ans2 = greaterThanThree1(arr); //More Opitimized , Given : ans should
always return 2 ele
    for(auto ele : ans2){
        cout<<ele<<" ";
    }cout<<endl;

    //Trapping Rain Water
    vector<int> height = {4,2,0,5,2,6,2,3};
    cout<<trapWater(height)<<endl;

    //threeSum
    vector<int> arr1 = {1,4,45,6,10,8};
    int target = 13;
    cout<<threeSum(arr1,target)<<endl;

    return 0;
}


//Add 2 Number
int main(){
    string str1 = "999583";
    string str2 = "798";

    int carry = 0;
    int first = str1.size()-1, second = str2.size()-1;
    string ans = "";
    while(second>=0){
        int sum = str1[first--]-'0' + str2[second--]-'0' + carry;
        carry = sum/10;
        ans += (sum%10 + '0');
    }

    while(first>=0){
        int sum = str1[first--]-'0' + carry;
        carry = sum/10;
        ans += (sum%10 + '0');
    }
    if(carry){
        ans+="1";
    }
    reverse(ans.begin(),ans.end());
    cout<<ans;
    return 0;
}



//Longest Substring without repeatring char
int main(){
    string str = "abcdecbeadf";
    int length = INT_MIN, first = 0, second = 0;
    vector<bool> v(256,0); //to handle all ASCII characters
    while(second<str.size()){
        while(v[str[second]]){
            v[str[first++]] = 0;
        }
        v[str[second]] = 1;
        length = max(length,second-first+1);
        second++;
    }
    cout<<"Longest SubString: "<<length<<endl;
    return 0;
}


//Smallest Distinct Window
int main(){
    string str = "aabcbcdbca";
    int first = 0, second = 0, length = INT_MAX, diff = 0;
    vector<int> v(256,0);
    while(first<str.size()){
        if(v[str[first]]==0){
            diff++;
        }
        v[str[first]]++;
        first++;
    }
    first = 0;

    for(int i=0; i<256; i++){
        v[i] = 0;
    }

    while(second<str.size()){
        while(diff and second<str.size()){
            if(v[str[second]]==0){
                diff--;
            }
            v[str[second]]++;
            second++;
        }

        length = min(length,second-first);

        while(diff!=1){
            length = min(length, second-first);
            v[str[first]]--;
            if(v[str[first]]==0) diff++;
            first++;
        }
    }

    cout<<"Length: "<<length<<endl;
    return 0;
    }
```