30 Days Leetcode challenge - Day 13

import java.util.HashMap;

public class leetday13

{

class Solution1 {

public int subarraySum(int[] arr, int k) {

int prefixSum = 0;

int count = 0;

int n = arr.length;

HashMap<Integer,Integer> map = new HashMap<>();

map.put(0,1);

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

{

prefixSum = prefixSum + arr[i];

int remove = prefixSum - k ;

if(map.containsKey(remove))

{

count += map.get(remove);

}

map.put(prefixSum, map.getOrDefault(prefixSum, 0) + 1);

}

return count;

}

}

class Solution2 {

public int search(int[] arr, int target)

{

int low = 0;

int high = arr.length - 1 ;

int ans = binarySearch(arr, target, low, high);

if (ans==-1)

{

return -1;

}

else{

return ans;

}

}

public static int binarySearch(int arr[], int tar, int low , int high)

{

if(low>high) return -1;

int mid = low+(high-low)/2;

if(arr[mid]==tar) return mid;

if(arr[mid]<tar) return binarySearch(arr, tar, mid+1, high);

return binarySearch(arr, tar, low, mid-1);

}

}

}