```
// Day 4 2Sum

import java.util.*;
class sum2{
	public static void main(String[] args) {

		int arr[] ={2,7,6,7,9};
		int n=arr.length;
		int target = 9;
		ans obj = new ans(arr,n,target);
		System.out.println(obj.sum_ans());
	}
}

class ans{
	int a[]=new int[10];
	int n;
	int tar;
	ans(int arr[], int n,int tar){
		a=arr;
		this.n=n;
		this.tar=tar;
```

```
     }

     public int[] sum_ans(){
    HashMap<Integer,Integer> map= new HashMap<>();
          for(int i=0;i<n;i++){
               map.put(a[i],i);
          }

          for(int i=0;i<n;i++){
               int num = a[i];
               int rem = tar - num;
               if(map.containsKey(rem)){
                    int index = map.get(i);
                    return new int[] {num , index};
               }
          }

          return new int[]{};

     }

}
```

```
/// O(Nlogn)  + O(1)
import java.util.*;
class longestSubarrayWithZero{
        static int longSum(int arr[], int n)
        {
                int sum=0;
                int maxi=0;
                HashMap<Integer,Integer> mpp = new HashMap<>();
                for(int i=0;i<n;i++){
                        sum+=arr[i];

                        if (sum==0) {
                                maxi= i+1;
                        }
                        if(mpp.get(sum)!=null){
                                maxi=Math.max(maxi,i-mpp.get(sum));
                        }
                        else
                        {
                                mpp.put(sum,i);
                        }
                }
                return maxi;
        }
   public static void main(String[] args) {
        int arr[] ={1,-1,3,2,-2,-8,1,7,10,23};
        int n=arr.length;
        System.out.println(longSum(arr,n));
   }
}
```

```
// Day 4 Longest SubString for non repeating

import java.util.*;
class LongestSubstringNonRepeat{
        static int LongestSubstringNonRepeat_fun(String str){
                int n = str.length();
                int left=0,right=0,len=0;
                HashMap<Character,Integer> mpp = new
HashMap<Character,Integer>();
                while(right<n){
                    if(mpp.containsKey(str.charAt(right)))
                        left=Math.max(mpp.get(str.charAt(right))+1,
left);
                    mpp.put(str.charAt(right),right);
                    len = Math.max(len,right - left +1);
                    right++;
                }
                return len;
        }
        public static void main(String[] args) {
          String str = "abcabcdeacd";
          int len = LongestSubstringNonRepeat_fun(str);
          System.out.println(len);
        }
}


//maxsubarray with xor for K
import java.util.*;
class maxsubArrayXORasK{

        static int maxsubArrayXORasK_fun(int a[], int k){
```

```
		Map<Integer,Integer> freq = new
	HashMap<Integer,Integer>();
		int xor =0, cnt=0;
		for (int i=0;i<a.length ;i++ ) {
			xor = xor^a[i];

			if(freq.get(xor^k)!=null){
				cnt+=freq.get(xor^k);
			}
			if(xor==k){
				cnt+=1;
			}

			if(freq.get(xor)!=null){
				freq.put(xor,freq.get(xor)+1);
			}
			else
			{
				freq.put(xor,1);
			}
		}
		System.out.println(freq);
```

```
        return cnt;

    }

    public static void main(String[] args) {

        int arr[] ={2,4,4,6,2,5,7};
        int k =6;
        int cnt= maxsubArrayXORasK_fun(arr,k);
        System.out.println(cnt);

    }
}

// subarraymaxconsequtive Element

import java.util.*;
class subarraymaxconsequtiveElement{
    static int maxConseqElement_fun(int arr[]){
        Set<Integer> nums = new HashSet<Integer>();
        for(int num : arr){
            nums.add(num);
        }
        int longstreak=0;
        for(int num: arr){
            if(!nums.contains(num-1)){
                int currnum = num;
                int currcnt = 1;
                while(nums.contains(currnum+1)){
                    currcnt+=1;
                    currnum+=1;
                }
                longstreak=Math.max(longstreak, currcnt);
            }

        }
        return longstreak;
    }
    public static void main(String[] args) {
        int arr[]= {101,102,2,3,4,1,6,103};
        int longStreak = maxConseqElement_fun(arr);
        System.out.println(longStreak);
            }
}
```

```
// Day 4 2Sum

import java.util.*;
class sum2{
	public static void main(String[] args) {

		int arr[] ={2,7,11,15};
		int n=arr.length;
		int target = 9;
		ans obj = new ans(n,target);
		int anss[] =Arrays.copyOf(obj.sum_ans(arr),2);
		for (int i :anss ) {
			System.out.print(i+" ");
		}

	}
}

class ans{
	int n;
	int tar;
```

```
   ans(int n,int tar){
           this.n=n;
           this.tar=tar;
   }

   int[] sum_ans(int a[]){

HashMap<Integer,Integer> map = new HashMap<>();
//List<Integer> ll = new ArrayList<>();
     for(int i=0 ;i< n ;i++){
           map.put(a[i],i);

    }
    for (int i =0;i<n ;i++ ) {
           int num= a[i];
           int rem = tar - a[i];
           if (map.containsKey(rem)){
                      int idx = map.get(rem);
                      if(idx == i) continue;
                      return new int[]{i,idx};
           }
```

```
          }
          return new int[]{};
         }
}




import java.util.*;

class sum3 {
        static List<List<Integer>> res = new ArrayList<>();
        static List<List<Integer>> threesum(int num[]){
                Arrays.sort(num);
                int lo,hi,sum;
                for (int i=0 ;i<num.length-2 ;i++ ) {
                        if (i== 0 || (i>0 && num[i] != num [i-1])) {
                                lo=i+1;
                                hi=num.length-1;
                                sum =0-num[i];
                                while(lo<=hi){
                                        if(num[lo]+ num[hi] == sum){
```

```
        res.add(Arrays.asList(num[i],num[lo],num[hi]));
                                        while( lo < hi && num[lo] ==
num[lo+1]) lo++;
                                        while( hi > lo && num[hi] == num[hi-
1]) hi--;
                                        lo++;
                                        hi--;
                                    }
                                    else if(num[lo] + num [hi] < sum ) lo ++;
                                    else hi--;
                                }
                            }
         }
        return res;
        }
        public static void main(String[] args) {
                int a[] = {-1,0,1,2,-1,-4};
                 System.out.println(threesum(a));
        }
}
```

// Sum 4 linear Complexity

```
import java.util.*;

class sum4{

        static ArrayList<List<Integer>> res = new
ArrayList<List<Integer>>();
        static ArrayList<List<Integer>> four_sum(int a[],int tar){
                if(a.length == 0|| a == null) return res;

        Arrays.sort(a);
        int n= a.length;

        for(int i=0 ;i<n;i++ ){
                for ( int j=i+1; j<n;j++ ) {
                        int tar2 = tar -a[j] - a[i];
                        int left = j+1;
                        int right= n-1;

                        while(left < right){
                                int two_sum = a[left] + a[right];
                                if (two_sum < tar2) left++;
                                else if(two_sum > tar2) right--;
                else{
                    List<Integer> quad = new ArrayList<>();
                    quad.add(a[i]);
                    quad.add(a[j]);
                    quad.add(a[left]);
                    quad.add(a[right]);
                    res.add(quad);
                    while(left < right && a[left] == quad.get(2))
++left;
                    while(left < right && a[right] ==
quad.get(3)) --right;
                }

            }

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

        }
        while(i+1 < n  && a[i+1] == a[i]) ++i;
    }
    return res;

    }
    public static void main(String[] args) {
        int arr[]={1,0,-1,0,-2,2};
        int tar = 0;
        System.out.println(four_sum(arr,0));
    }
}
```