**DAY-13**

**Majority element:**

class Solution {

public int majorityElement(int[] nums) {

if(nums.length == 1){

return nums[0];

}else{

int count =0;

int candidate = 0;

for(int num:nums){

if(count ==0){

candidate = num;

}

count+=(num == candidate)?1:-1;

}

return candidate;

}

}

}

initialize two variables count and candidate as 0 initially

-> traverse through the array

-> if count is zero then the candidate = 0 is replaced with num and increase the count if num and candidate are equal else decrement count

-> finally return the candidate.

From the given array assume that starting index of arr is my majority element

* From that target elememnt
* If we got same value increase the count if we got different values
* If counter becomes zero it menas till now we havnt seen majority element again select the new target element . The Very next element is your target element at the end of foorloop if counter is more than zero it menas that their s a majority element
* Taking that value(target) count how mnay times that waew reapted that array
* Counter is repeted n/2.

Step1:Create a set

Run a loop :using insert push elements of array in set

Iterate set values

We have to find whether the element is starting or not by decreasing the value and checking that value is their or not int the set.

If we got starting value then start comparing

**Longest Consecutive Elements:**

-> create a set

-> run a for loop take the elements from the array and push into set using insert() method

-> start iterating set() values

-> we have to find weather the element is starting or not by decreasing a value and checking that value is there or not in the set

-> if you got starting value then start comparing

code:

import java.util.\*;

public class Main {

public static void main(String[] args) {

int[] arr = {1, 2, 101, 3, 102, 4, 103};

int longest = findLongestConsecutive(arr);

System.out.println(longest);

}

public static int findLongestConsecutive(int[] arr) {

Arrays.sort(arr);

int longest = 1;

int current = 1;

for (int i = 1; i < arr.length; i++) {

if (arr[i] == arr[i - 1]) {

continue;

} else if (arr[i] == arr[i - 1] + 1) {

current++;

} else {

longest = Math.max(longest, current);

current = 1;

}

}

longest = Math.max(longest, current);

return longest;

}

}

LeetCode Problems:

* 128
* 169
* 1432
* 239
* 203
* 55
* 3