Question: https://leetcode.com/problems/first-missing-positive/

My first idea was to store array elements in a hashmap and then iterate from 1 to n and check if at any point the iterator point value is missing in hashmap or not if missing then that is the smallest missing positive number.

Now this approach has a O(n) time complexity as well a O(n) space complexity. We can make it better by reducing the time complexity to O(1).

So what we do is first we set the value of elements less than equal to 0 or greater then n to n+1. Thereby we can skip elements wit values n+1 at net step.

Now we check for elements without the value of n+1, and check if pos!=n and nums[pos]>0 then nums[pos]\*=-1; pos is actually element value-1.

Now itterate the array and check for the first element with +ve value then return the position+1 as result.

Code:  
class Solution {

public int firstMissingPositive(int[] nums) {

int n = nums.length;

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

if(nums[i]<=0||nums[i]>n){

nums[i]=n+1;

}

}

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

int pos = Math.abs(nums[i]);

if(pos>n){

continue;

}

pos--;

if(pos!=n && nums[pos]>0){nums[pos]\*=-1;}

}

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

if(nums[i]>0)

return i+1;

}

return n+1;

}

}

Github Link :<https://lnkd.in/ecwtJeaz>