Cyclic Sort.

→ When & Where to apply. ?
└→ given an array → length n.
↓
1 to n , 0 to n , 0 to n-1
Duplicate , missing No.

⇒ Ex ①.
n - length → 0 to n-1
↓
Each ele appearing Once

n = 5.

$$\text{arr} = \{ \overset{0}{4}, \overset{1}{1}, \overset{2}{2}, \overset{3}{0}, \overset{4}{3} \}$$

└→ sort → O(n) T. C.

swap(arr[i], arr[arr[i]])
idx = arr[i].

$$\to \{ \overset{0}{0}, \overset{1}{1}, \overset{2}{2}, \overset{3}{3}, \overset{4}{4} \}$$
i idx

⇒ Ex ②. n - length → 1 to n.
arr[i] arr = { 5 , 6 , 1 , 2 , 3 , 4 }.
i
↓
arr[arr[i] - 1]

⇒ In , each swap , atleast one element gets at right place

[# Time Complexity → O(n)]

LeetCode Qno. 268. ⇒ Missing Number.

n = 7 size 0 to 7 numbers are in array, missing Need to find.

Size ⇒ n+1

$$\text{arr} = \{ \overset{0}{1}, \overset{1}{6}, \overset{2}{4}, \overset{3}{7}, \overset{4}{0}, \overset{5}{5}, \overset{6}{2} \}$$

$$b = \{ \overset{0}{T}, \overset{1}{T}, \overset{2}{T}, \overset{3}{F}, \overset{4}{T}, \overset{5}{T}, \overset{6}{T}, \overset{7}{T} \} \Rightarrow \text{M-1}$$

M-1.

```
⇒ Code :- psvm{
    int[] nums = {9,6,4,2,3,5,7,0,1};
    int n = nums.length;
    boolean[] b = new boolean[n+1];
    for(int ele : nums){
        b[ele] = true;
    }
    for(int i = 0; i<=n ; i++){
        if(b[i]==false) Sout(i);
    }
}.
        for(i=0; i<n; i++){
            if(nums[i]!=i) Sout(i);
        }
    }.
```

M-2

arr = { 1, 6, 4, 7, 0, 5, 2 }

$$\to \{ \overset{0}{0}, \overset{1}{1}, \overset{2}{2}, \overset{3}{7}, \overset{4}{4}, \overset{5}{5}, \overset{6}{6} \}$$
i

```
⇒ Code :- psvm{
    int[] nums = {9,6,4,2,3,5,7,0,1};
    int n = nums.length;
    int i = 0;
    while(i<n){
        if(nums[i]==i || nums[i]==n)
            i++;
        else{
            swap(i,nums[i],nums);
        }
    }.
```

LeetCode Q No. 287 → Find Duplicate

```
          0   1   2   3   4
nums = { 1 , 3 , 4 , 2 , 2 }          1 to n
                                      Find duplicate.
[use the 0th index to check or swap]
```

```
=> Code :- psvm {
    int[] nums = {1, 3, 4, 2, 2};
    while (true) {
        int ele = nums[0];
        if (nums[ele] == ele) {
            Sout(ele);
            break;
        }
        swap(0, ele, nums);
    }
}
```

LeetCode Q No. 448

```
        0   1   2   3   4   5   6   7
arr = { 4 , 3 , 2 , 7 , 8 , 2 , 3 , 7 }

CycleSort
        0   1   2   3   4   5   6   7
      { 3 , 2 , 3 , 4 , 7 , 2 , 7 , 8 }
        i

ans = { 1, 5, 6 } //
```

```
=> Code :- psvm {
    int[] nums = {4, 3, 2, 7, 8, 2, 3, 7};
    int n = nums.length;
    int i = 0;
    while (i < n) {
        int ele = num[i];
        if (nums[i] == i+1 || nums[i] ==
                nums[ele-1]) i++;
        else {
            swap(i, ele-1, nums);
        }
    }

    ArrayList<Integer> ans = new ArrayList<>();
    for (i = 0; i < n; i++) {
        if (nums[i] != i+1) ans.add(i+1);
    }
    Sout(ans);
}
```

LeetCode Q No. 41

=> First Missing Positive
 # Using Cycle sort.

```
        0   1   2   3
arr = { 1 , 8 , 7 , 4 }
            i
  └──> [Ans = 2]

        0   1   2   3
arr = { 7 , 8 , 9 , 10 }
        i
  └──> [Ans = 1]
```

i++ when ?   where
① arr[i] <= 0.   idx = arr[i]
② arr[i] = i+1.
③ arr[i] > n.
④ arr[i] == arr[idx-1].

```
=> Code :- psvm
    int[] nums = {3, 4, -1, 1};
    int n = nums.length;
    int i = 0;
    while (i < n) {
        int ele = nums[i];  // ele will be placed at ele-1 idx.
        if (ele <= 0 || ele == i+1 || ele > n || ele == nums[ele-1]) i++;
        else swap(i, ele-1, nums);
    }
    for (i = 0; i < n; i++) {
        if (nums[i] != i+1) Sout(i+1);
    }
}
```

IM-2 $\longrightarrow$ (41)

using HashSet.

=> Code :- psvm {

```
int[] nums = { 3, 4, -1, 1};
int n = nums.length;
HashSet<Integer> set = new HashSet<>();
for(int ele : nums){
      set.add(ele);
}
for(int num = 1 ; num <=n+1 ; num ++){
      if( ! set.contains (num)) {
            sout (num);
            break;
      }
}
}
```

M T W T F S S
Page No.:
Date:
YOUVA

Home Work Done

① Leetcode Q No 645 ✓

② Leetcode Q No 442. ✓

— × — × — × — × — × —