Array #16

Leetcode #448

Find All Numbers Disappeared In An Array

https://leetcode.com/problems/find-all-numbers-disappeared-in-an-array/description/

Given an array of integers nums, where nums[i] is in the range [1, n], return an array of all integers in the range [1, n] that do not appear in nums

Example 1:
Input: nums = [4, 3, 2, 7, 8, 2, 3, 1]
Output: [5, 6]

8 → 1..8

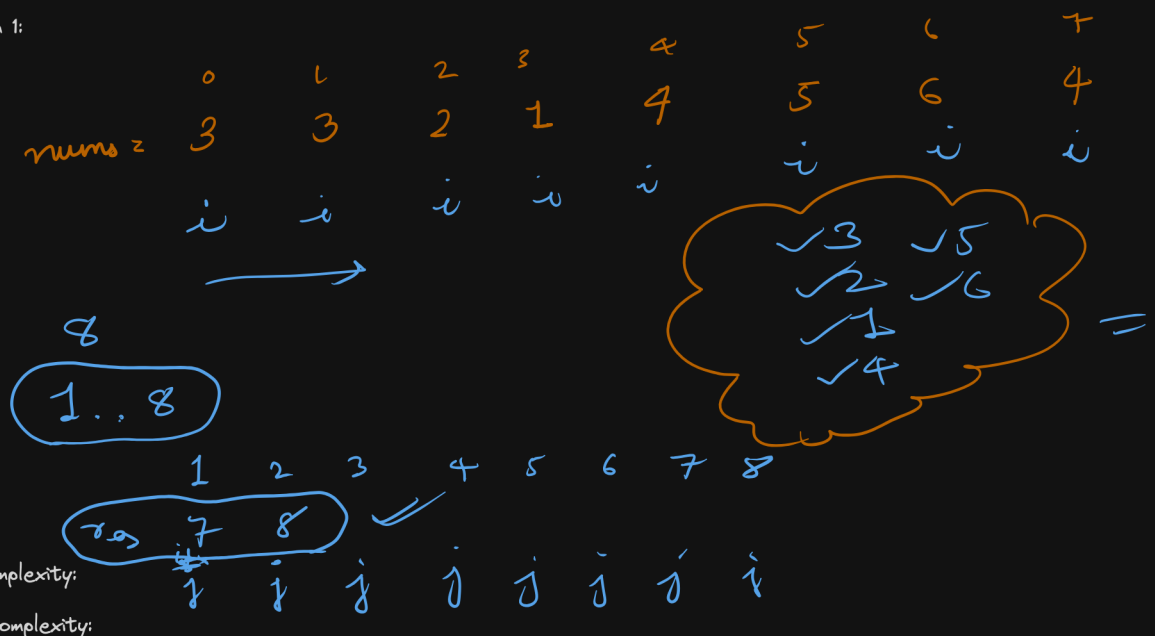
Example 2:
Input: nums = [1, 1]
Output: [2]

2 → 1..2

Constraints:
n == nums.length
1 <= nums[i] <= n
1 <= n <= 10^5

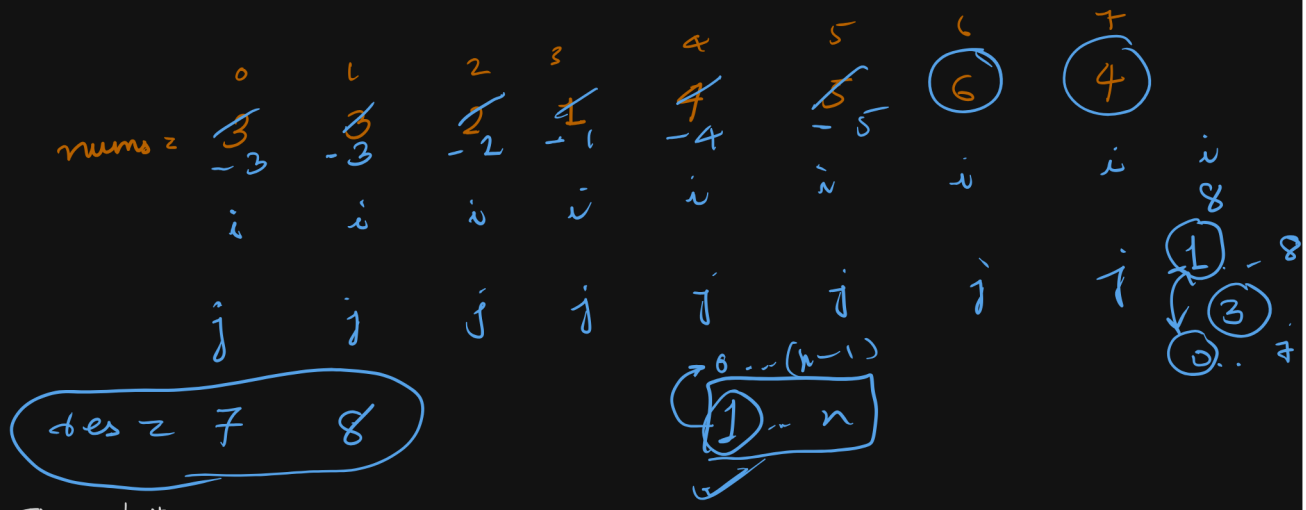
Companies:
Meta, Amazon, Google

Approach 1:



Time complexity:
O(N)
Space Complexity:
O(N)

Approach 2:



Time complexity:
O(N)
Space Complexity:
O(1)

```
class Solution {
    public List<Integer> findDisappearedNumbers(int[] nums) {
        Set<Integer> set = new HashSet<>();

        for (int num: nums) {
            set.add(num);
        }

        List<Integer> res = new LinkedList<>();

        for (int i = 1; i <= nums.length; i++) {
            if (!set.contains(i)) {
                res.add(i);
            }
        }

        return res;
    }
}
```

```
class Solution {
    public List<Integer> findDisappearedNumbers(int[] nums) {
        for (int i = 0; i < nums.length; i++) {
            int currNum = Math.abs(nums[i]);

            int idx = currNum - 1;

            if (nums[idx] < 0) {
                continue;
            }

            nums[idx] = -1 * nums[idx];
        }

        List<Integer> res = new LinkedList<>();
        // 1   1
        // -1  1
        for (int i = 0; i < nums.length; i++) {
            if (nums[i] > 0) {
```

```
                    res.add(i + 1);
                }
            }

            return res;
        }
    }
```