Find All Numbers Disappeared in an Array

# Question

Given an array of integers where 1 ≤ a[i] ≤ n (n = size of array), some elements appear twice and others appear once.

Find all the elements of [1, n] inclusive that do not appear in this array.

Could you do it without extra space and in O(n) runtime? You may assume the returned list does not count as extra space.

**Example:**

Input:

[4,3,2,7,8,2,3,1]

Output:

[5,6]

# Pseudo Code

## V1.0

Allocate the Memory in Heap for the Dynamic Array

Declare and Initialize k = 0;

Run the First For Loop from i = 1 to size

Run the Second For Loop from j = 0 to size

If arr[j] is equal to i

Break out from the loop

If j is equal to (numsSize - 1)

ptr[k] = i

K++

\*returnSize = k

Return ptr

# Source Code

## V1.0 (Language C)

**\*Solution not accepted due to more runtime O(n^2) Complexity, but the approach is correct\***

1. /\*\*
2. \* Note: The returned array must be malloced, assuming caller calls free().
3. \*/
4. int\* findDisappearedNumbers(int\* nums, int numsSize, int\* returnSize){
6. if(numsSize == 0) {
7. \*returnSize = numsSize;
8. return (int\*)malloc(numsSize \* sizeof(int));
9. }
11. int missingNumsCounter = 0;
12. int\* mallocedNums = (int\*)malloc(numsSize \* sizeof(int));
14. for(int i=1 ; i<=numsSize ; i++) {
15. for(int j=0 ; j<numsSize ; j++) {
16. if(nums[j] == i) {
17. break;
18. }
20. if(j == (numsSize - 1)) {
21. mallocedNums[missingNumsCounter] = i;
22. missingNumsCounter++;
23. }
24. }
25. }
26. \*returnSize = missingNumsCounter ;
27. return mallocedNums;
28. }

## v2.0 (Language C)

1. /\*\*
2. \* Note: The returned array must be malloced, assuming caller calls free().
3. \*/
4. int\* findDisappearedNumbers(int\* nums, int numsSize, int\* returnSize){
6. if(numsSize == 0) {
7. \*returnSize = numsSize;
8. return (int\*)malloc(numsSize \* sizeof(int));
9. }
11. int\* mallocedNums = (int\*)malloc(numsSize \* sizeof(int));
12. int missingNumsCounter = 0;
14. for(int i=0 ; i<numsSize ; i++) {
15. if (nums[abs(nums[i]) - 1] > 0) {
16. nums[abs(nums[i]) - 1] \*= -1;
17. }
18. }
20. for(int i=0 ; i<numsSize ; i++) {
21. if(nums[i] > 0) {
22. mallocedNums[missingNumsCounter] = i + 1;
23. missingNumsCounter++;
24. }
25. else {
26. nums[i] \*= -1;
27. }
28. }
30. \*returnSize = missingNumsCounter;
31. return mallocedNums;
32. }