Grokking the Coding Interview: Patterns for Coding Questions

27% completed

Search Course

Solution Review: Problem Challenge 2
Problem Challenge 3
Solution Review: Problem Challenge 3

Pattern: Cyclic Sort

Introduction
Cyclic Sort (easy)
Find the Missing Number (easy)
Find all Missing Numbers (easy)
Find the Duplicate Number (easy)
Find all Duplicate Numbers (easy)
Problem Challenge 1
Solution Review: Problem Challenge 1
Problem Challenge 2
Solution Review: Problem Challenge 2
Problem Challenge 3
Solution Review: Problem Challenge 3

Pattern: In-place Reversal of a LinkedList

Introduction
Reverse a LinkedList (easy)
Reverse a Sub-list (medium)
Reverse every K-element Sub-list (medium)
Problem Challenge 1
Solution Review: Problem Challenge 1
Problem Challenge 2
Solution Review: Problem Challenge 2

Pattern: Tree Breadth First Search

Introduction
Binary Tree Level Order Traversal (easy)
Reverse Level Order Traversal (easy)
Zigzag Traversal (medium)
Level Averages in a Binary Tree (easy)
Minimum Depth of a Binary Tree (easy)
Level Order Successor (easy)
Connect Level Order Siblings (medium)
Problem Challenge 1
Solution Review: Problem Challenge 1
Problem Challenge 2
Solution Review: Problem Challenge 2

Pattern: Tree Depth First Search

Find all Missing Numbers (easy)

We'll cover the following

- Problem Statement
- Try it yourself
- Solution
- Code
  - Time complexity
  - Space complexity

Problem Statement

We are given an unsorted array containing numbers taken from the range 1 to 'n'. The array can have duplicates, which means some numbers will be missing. Find all those missing numbers.

Example 1:

```
Input: [2, 3, 1, 8, 2, 3, 5, 1]
Output: 4, 6, 7
Explanation: The array should have all numbers from 1 to 8, due to duplicates 4, 6, and 7 are m
issing.
```

Example 2:

```
Input: [2, 4, 1, 2]
Output: 3
```

Example 3:

```
Input: [2, 3, 2, 1]
Output: 4
```

Try it yourself

Try solving this question here:

Java | Python3 | JS | C++

```
const find_missing_numbers = function(nums) {
  missingNumbers = [];
  // TODO: Write your code here
  return missingNumbers;
};

```

TEST SAVE RESET

Solution

This problem follows the **Cyclic Sort** pattern and shares similarities with Find the Missing Number with one difference. In this problem, there can be many duplicates whereas in 'Find the Missing Number' there were no duplicates and the range was greater than the length of the array.

However, we will follow a similar approach though as discussed in Find the Missing Number to place the numbers on their correct indices. Once we are done with the cyclic sort we will iterate the array to find all indices that are missing the correct numbers.

Code

Here is what our algorithm will look like:

Java | Python3 | C++ | JS

```
function find_missing_numbers(nums) {
  let i = 0;
  while (i < nums.length) {
    const j = nums[i] - 1;
    if (nums[i] !== nums[j]) {
      [nums[i], nums[j]] = [nums[j], nums[i]]; // swap
    } else {
      i += 1;
    }
  }
  missingNumbers = [];

  for (i = 0; i < nums.length; i++) {
    if (nums[i] !== i + 1) {
      missingNumbers.push(i + 1);
    }
```

Introduction
Binary Tree Path Sum (easy)
All Paths for a Sum (medium)
Sum of Path Numbers (medium)
Path With Given Sequence (medium)
Count Paths for a Sum (medium)
Problem Challenge 1
Solution Review: Problem Challenge 1
Problem Challenge 2
Solution Review: Problem Challenge 2

Pattern: Two Heaps

Introduction
Find the Median of a Number Stream (medium)
Sliding Window Median (hard)
Maximize Capital (hard)
Problem Challenge 1
Solution Review: Problem Challenge 1

Pattern: Subsets

Introduction
Subsets (easy)
Subsets With Duplicates (easy)
Permutations (medium)
String Permutations by changing case (medium)
Balanced Parentheses (hard)

```
    }

    return missingNumbers;
  }


  console.log(find_missing_numbers([2, 3, 1, 8, 2, 3, 5, 1]));
  console.log(find_missing_numbers([2, 4, 1, 2]));
  console.log(find_missing_numbers([2, 3, 2, 1]));
```

RUN SAVE RESET

Time complexity

The time complexity of the above algorithm is $O(n)$.

Space complexity

Ignoring the space required for the output array, the algorithm runs in constant space $O(1)$.

Interviewing soon? We've partnered with Hired so that companies apply to you, instead of the other way around. See how

✓ MARK AS COMPLETED

← Back
Find the Missing Number (easy)

Next →
Find the Duplicate Number (easy)

Report an Issue