

Grokking the Coding Interview: Patterns for Coding Questions

11% completed

- Pair with Target Sum (easy)
- Remove Duplicates (easy)
- Squaring a Sorted Array (easy)
- Triplet Sum to Zero (medium)
- Triplet Sum Close to Target (medium)
- Triplets with Smaller Sum (medium)
- Subarrays with Product Less than a Target (medium)
- Dutch National Flag Problem (medium)
- 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: Fast & Slow pointers

- Introduction
- LinkedList Cycle (easy)
- Start of LinkedList Cycle (medium)
- Happy Number (medium)
- Middle of the LinkedList (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: Merge Intervals

- Introduction
- Merge Intervals (medium)
- Insert Interval (medium)
- Intervals Intersection (medium)
- Conflicting Appointments (medium)
- 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: Cyclic Sort

- Introduction
- Cyclic Sort (easy)
- Find the Missing Number (easy)

Solution Review: Problem Challenge 1

We'll cover the following

- Quadruple Sum to Target (medium)
- Solution
 - Code
 - Time complexity
 - Space complexity

Quadruple Sum to Target (medium)

Given an array of unsorted numbers and a target number, find all **unique quadruplets** in it, whose **sum is equal to the target number**.

Example 1:

```
Input: [4, 1, 2, -1, 1, -3], target=1
Output: [-3, -1, 1, 4], [-3, 1, 1, 2]
Explanation: Both the quadruplets add up to the target.
```

Example 2:

```
Input: [2, 0, -1, 1, -2, 2], target=2
Output: [-2, 0, 2, 2], [-1, 0, 1, 2]
Explanation: Both the quadruplets add up to the target.
```

Solution

This problem follows the **Two Pointers** pattern and shares similarities with [Triplet Sum to Zero](#).

We can follow a similar approach to iterate through the array, taking one number at a time. At every step during the iteration, we will search for the quadruplets similar to [Triplet Sum to Zero](#) whose sum is equal to the given target.

Code

Here is what our algorithm will look like:

JavaPython3C++JS

```
1 function search_quadruplets(arr, target) {
2   arr.sort((a, b) => a - b)
3   const quadruplets = [];
4   for (let i = 0; i < arr.length - 3; i++) {
5     // skip same element to avoid duplicate quadruplets
6     if (i > 0 && arr[i] === arr[i - 1]) {
7       continue;
8     }
9     for (let j = i + 1; j < arr.length - 2; j++) {
10      // skip same element to avoid duplicate quadruplets
11      if (j > i + 1 && arr[j] === arr[j - 1]) {
12        continue;
13      }
14      search_pairs(arr, target, i, j, quadruplets);
15    }
16  }
17  return quadruplets;
18 }

19
20
21 function search_pairs(arr, targetSum, first, second, quadruplets) {
22   let left = second + 1;
23   right = arr.length - 1;
24   while (left < right) {
25     sum = arr[first] + arr[second] + arr[left] + arr[right];
26     if (sum === targetSum) { // found the quadruplet
27       quadruplets.push([arr[first], arr[second], arr[left], arr[right]]);
28       left++;
29     }
30   }
31 }
```

RUN

SAVE

RESET

Close

Output3.319s

[[[-3, -1, 1, 4], [-3, 1, 1, 2]]]

[[[-2, 0, 2, 2], [-1, 0, 1, 2]]]

Time complexity

Sorting the array will take $O(N * \log N)$. Overall `searchQuadruplets()` will take $O(N * \log N + N^3)$, which is asymptotically equivalent to $O(N^3)$.

Refer a Friend

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

+

Create

Pattern: In-place Reversal of a LinkedList

Space complexity

The space complexity of the above algorithm will be $O(N)$ which is required for sorting.

← Back


Problem Challenge 1

✓

 MARK AS COMPLETED

Next →

Problem Challenge 2

 Report an Issue