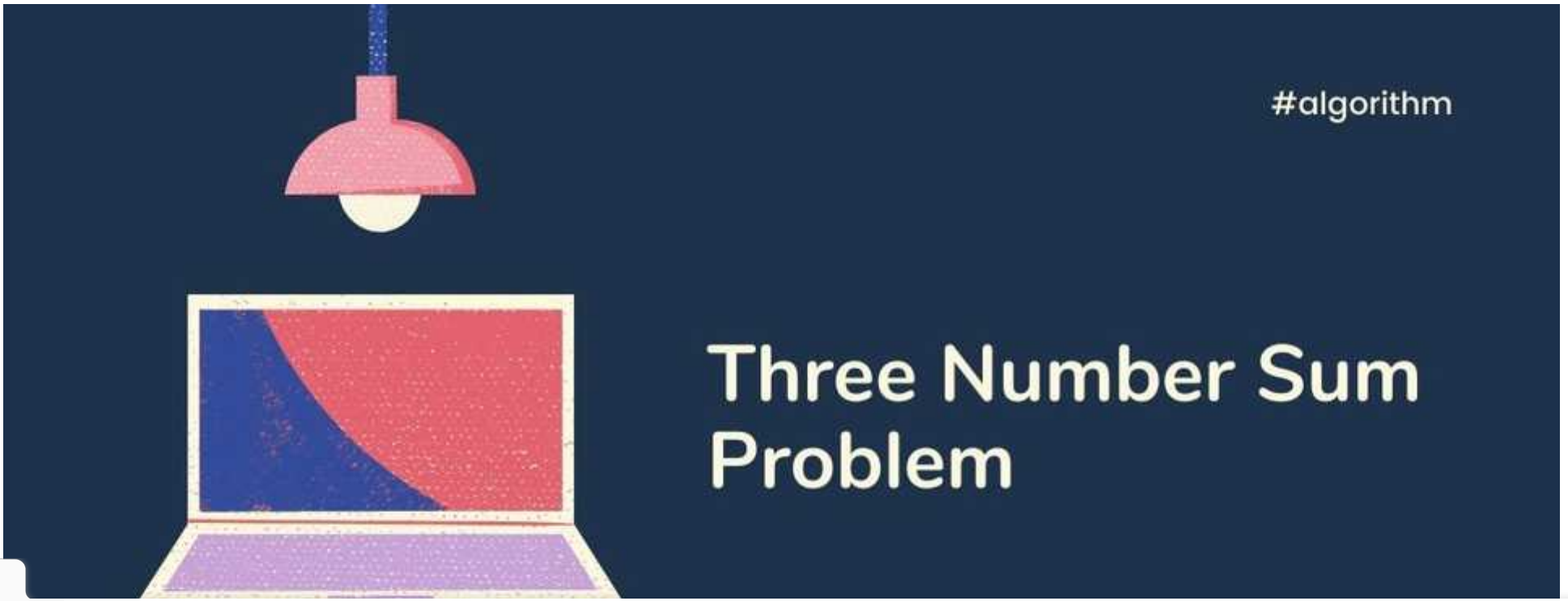
CALLICODER

Three Number Sum Problem (Find all triplets with the given sum)

by RAJEEV SINGH · ALGORITHMS · NOVEMBER 20, 2019 · 1 MINS READ



Three Number Sum Problem Statement

Given an array of integers, find all triplets in the array that sum up to a given target value.

In other words, given an array `arr` and a target value `target`, return all triplets a, b, c such that `a + b + c = target`.

Example:

```
Input array: [7, 12, 3, 1, 2, -6, 5, -8, 6]
Target sum: 0

Output: [[2, -8, 6], [3, 5, -8], [1, -6, 5]]
```

Three Number Sum Problem solution in Java

METHOD 1. Naive approach: Use three for loops

The naive approach is to just use three nested for loops and check if the sum of any three elements in the array is equal to the given target.

Time complexity: O(n^3)

```
import java.util.Scanner;
import java.util.List;
import java.util.ArrayList;
import java.util.Arrays;

class ThreeSum {

  // Time complexity: O(n^3)
  private static List<Integer[]> findThreeSum_BruteForce(int[] nums, int target) {
    List<Integer[]> result = new ArrayList<>();
    for (int i = 0; i < nums.length; i++) {
      for (int j = i + 1; j < nums.length; j++) {
        for (int k = j + 1; k < nums.length; k++) {
          if (nums[i] + nums[j] + nums[k] == target) {
```

```
    }
    return result;
  }

  public static void main(String[] args) {
    Scanner keyboard = new Scanner(System.in);

    int n = keyboard.nextInt();
    int[] nums = new int[n];

    for (int i = 0; i < n; i++) {
      nums[i] = keyboard.nextInt();
    }
    int target = keyboard.nextInt();

    keyboard.close();

    List<Integer[]> result = findThreeSum_Sorting(nums, target);

    for(Integer[] triplets: result) {
      for(int num: triplets) {
        System.out.print(num + " ");
      }
```

```
    }
  }
```

METHOD 2. Use Sorting along with the two-pointer approach

Another approach is to first sort the array, then -

- Iterate through each element of the array and for every iteration,
  - Fix the first element (`nums[i]`)
  - Try to find the other two elements whose sum along with `nums[i]` gives `target`. This boils down to the two sum problem.

Time complexity: O(n^2)

```java
import java.util.ArrayList;
import java.util.Arrays;

class ThreeSum {

  // Time complexity: O(n^2)
  private static List<Integer[]> findThreeSum_Sorting(int[] nums, int target) {
    List<Integer[]> result = new ArrayList<>();
    Arrays.sort(nums);
    for (int i = 0; i < nums.length; i++) {
      int left = i + 1;
      int right = nums.length - 1;
      while (left < right) {
        if (nums[i] + nums[left] + nums[right] == target) {
          result.add(new Integer[] { nums[i], nums[left], nums[right] });
          left++;
          right--;
        } else if (nums[i] + nums[left] + nums[right] < target) {
          left++;
        } else {
          right--;
        }
      }
```

```
    }
  }
```

METHOD 3. Use a Map/Set

Finally, you can also solve the problem using a Map/Set. You just need to iterate through the array, fix the first element, and then try to find the other two elements using the approach similar to the two sum problem.

I'm using a Set in the following solution instead of a Map as used in the two-sum problem because in the two-sum problem, we had to keep track of the index of the elements as well. But In this problem, we just care about the element and not its index.

Time complexity: O(n^2)

```java
import java.util.HashSet;

class ThreeSum {

  // Time complexity: O(n^2)
  private static List<Integer[]> findThreeSum(int[] nums, int target) {
    List<Integer[]> result = new ArrayList<>();
    for (int i = 0; i < nums.length; i++) {
      int currentTarget = target - nums[i];
      Set<Integer> existingNums = new HashSet<>();
      for (int j = i + 1; j < nums.length; j++) {
        if (existingNums.contains(currentTarget - nums[j])) {
          result.add(new Integer[] { nums[i], nums[j], currentTarget - nums[j] });
        } else {
          existingNums.add(nums[j]);
        }
      }
    }
    return result;
  }
}
```

Share on social media