LeetCode 532

https://leetcode.com/problems/k-diff-pairs-in-an-array/description/

Yifeng Zeng

Description

532. K-diff Pairs in an Array

Given an array of integers and an integer k, you need to find the number of unique k-diff pairs in the array. Here a k-diff pair is defined as an integer pair (i, j), where i and j are both numbers in the array and their absolute difference is k.

Example 1:

Input: [3, 1, 4, 1, 5], k = 2

Output: 2

Explanation: There are two 2-diff pairs in the array, (1, 3) and (3, 5).
Although we have two 1s in the input, we should only return the number of unique pairs.

Example 2:

Input:[1, 2, 3, 4, 5], k = 1

Output: 4

Explanation: There are four 1-diff pairs in the array, (1, 2), (2, 3), (3, 4) and (4, 5).

Example 3:

Input: [1, 3, 1, 5, 4], k = 0

Output: 1

Explanation: There is one 0-diff pair in the array, (1, 1).

Note:

- The pairs (i, j) and (j, i) count as the same pair.
- The length of the array won't exceed 10,000.
- All the integers in the given input belong to the range: [-1e7, 1e7].

Idea Report

We are looking for absolute difference, so if k < 0, then we return 0. For each number n in nums, we are actually looking to see if there is a n + k, or a n - k so that their absolute difference is k. And to find another number in an array using just O(1) time, we think of a hash map. The key is the number itself, and the value is the frequency it appears. If k == 0, we only need to see if any number n's frequency is larger than 1, and if the frequency is larger than 1, we increase our counter, and return it after checking all the keys in the map. If k != 0, for any number n, we see if n + k is in the input, and we see if n - k is in the input, we increase our counter if they are in the input. At the end, we need to return half of the counter because when count the numbers, we have duplications. For example if input is [2,4,5] and k = 2. When we check 2, we see n + k = 2 + 2 = 4 is in the map, and wehn we check 4, we see n - k = 4 - 2 = 2 is in the map, but this (2,4) pair and (4,2) pair are consider the same, so finally our count is double the size of the actual result, so we return count / 2 if k != 0.

Code

```java
class Solution {
    public int findPairs(int[] nums, int k) {
        if (nums == null || nums.length == 0 || k < 0) {
            return 0;
        }

        Map<Integer, Integer> map = new HashMap<>();
        for (int n : nums) {
            map.put(n, map.getOrDefault(n, 0) + 1);
        }

        int count = 0;
        for (int n : map.keySet()) {
            if (k == 0 && map.get(n) > 1) {
                count++;
            } else if (k != 0) {
                if (map.containsKey(n + k)) {
                    count++;
                }
                if (map.containsKey(n - k)) {
                    count++;
                }
            }
        }

        return k == 0 ? count : count / 2;
    }
}
```

Summary

- Use HashMap when we need to find a number in a collection in O(1) time.