15. 3Sum

Difficulty	medium
☑	☑
Finished	@July 10, 2023
Problem	array
Previously asked company	Facebook Goldman Sachs
website	leetcode

Question:

Given an integer array nums, return all the triplets `[nums[i], nums[j], nums[k]]` such that `i != j`, `i != k`, and `j != k`, and `nums[i] + nums[j] + nums[k] == 0`.

Notice that the solution set must not contain duplicate triplets.

Example 1:

```
Input: nums = [-1,0,1,2,-1,-4]
Output: [[-1,-1,2],[-1,0,1]]
Explanation:
nums[0] + nums[1] + nums[2] = (-1) + 0 + 1 = 0.
nums[1] + nums[2] + nums[4] = 0 + 1 + (-1) = 0.
nums[0] + nums[3] + nums[4] = (-1) + 2 + (-1) = 0.
The distinct triplets are [-1,0,1] and [-1,-1,2].
Notice that the order of the output and the order of the triplets does not matter.
```

Example 2:

```
Input: nums = [0,1,1]
Output: []
Explanation: The only possible triplet does not sum up to 0.
```

Example 3:

```
Input: nums = [0,0,0]
Output: [[0,0,0]]
Explanation: The only possible triplet sums up to 0.
```

Optimal Solution:

Time complexity: O(n^2)

Space complexity: O(1) or O(n) based on the sorting technique that is used

```
class Solution(object):
    def threeSum(self, nums):
        res = []
        nums.sort()
```

```
        for i, a in enumerate(nums):
            if i > 0 and a == nums[i-1]:
                continue

            l, r = i+1, len(nums)-1
            while l < r:
                threeSum = a + nums[l] + nums[r]
                if threeSum < 0:
                    l += 1
                elif threeSum > 0:
                    r -= 1
                else:
                    res.append([a,nums[l],nums[r]])
                    l += 1
                    while nums[l] == nums[l - 1] and l < r:
                        l += 1

        return res
```