

**Solution:**

class Solution:

    def threeSum(self, nums: List[int]) -> List[List[int]]:

        rs = []

        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):

                sum = a + nums[l]+nums[r]

                if sum > 0:

                    r -= 1

                elif sum < 0:

                    l += 1

                else:

                    rs.append([a, nums[l], nums[r]])

                    l += 1

                    while nums[l] == nums[l - 1] & l < r:

                        l +=1

        return rs

demo = Solution()

demo.threeSum([-1,0,1,2,-1,-4])

**Output:**

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

**Time Complexity:** O(n log n)+O(n2) = O(n2) - as here many repeated work is doing

**Space Complexity:** O(1) or O(n) – based of sort method works