18. 4sum

18. 4sum

ksum的递归方案, 重点是

```
if(i > start && nums[i] == nums[i - 1]) continue;
```

可以优化很多重复的条件.

```
public List<List<Integer>> fourSum(int[] nums, int target) {
        Arrays.sort(nums);
        return kSum(nums, 0, 4, target);
    }
    private List<List<Integer>> kSum (int[] nums, int start, int k, int target) {
        int len = nums.length;
        List<List<Integer>> res = new ArrayList<List<Integer>>();
        if(k == 2) { //two pointers from left and right
            int left = start, right = len - 1;
            while(left < right) {
                int sum = nums[left] + nums[right];
                if(sum == target) {
                    List<Integer> path = new ArrayList<Integer>();
                    path.add(nums[left]);
                    path.add(nums[right]);
                    res.add(path);
                    while(left < right && nums[left] == nums[left + 1]) left++;
                    while(left < right && nums[right] == nums[right - 1]) right--;
                    left++;
                    right--;
                } else if (sum < target) { //move left
                    left++;
                } else { //move right
                    right--;
                }
            }
        } else {
            for(int i = start; i < len - (k - 1); i++) {
                if(i > start && nums[i] == nums[i - 1]) continue;
                List<List<Integer>> temp = kSum(nums, i + 1, k - 1, target -
    nums[i]);
                for(List<Integer> t : temp) {
                   t.add(0, nums[i]);
                }
                res.addAll(temp);
            }
        }
        return res;
    }
```