4 Sum

You're solving the classic 4 Sum problem, which is a natural extension of 2 Sum and 3 Sum. The key idea is to:

- Fix the first two numbers in nested loops
- Use two pointers to find the remaining two numbers
- Avoid duplicates carefully at every level.

Strategy: Sort + Two Pointers + Duplicate Skipping:

1. Sort nums to help skip duplicates and use two pointers efficiently.
2. Loop with index i from 0 to n-4: Skip duplicates for i.
3. Loop with index j from i+1 to n-3: Skip duplicates for j.
4. Use two pointers left = j+1, right = n-1.
   - Calculate the sum.
   - If it matches the target → store it and move both pointers, skipping duplicates.
   - If less than target → move left++
   - If greater than target → move right--

Time Complexity:

- Outer loops: $O(n^2)$
- Two pointers inside: $O(n)$
- Total: $O(n^3)$
- Efficient for $n \leq 200$

Edge Cases:

- Return only unique quadruplets.
- Handle input like [2,2,2,2,2] carefully.
- Don't access out-of-bounds in the loops.