■ Four Sum Problem – Notes

Find all unique quadruplets in an array that sum up to a given target value.

Algorithm: 1. Sort the array to simplify duplicate handling. 2. Fix the first two numbers using two nested loops (i and j). - Skip duplicates for i and j. 3. Use two pointers (k and l) to find the remaining two numbers. - Initialize k = j + 1, l = n - 1. - Calculate total = arr[i] + arr[j] + arr[k] + arr[l]. - If total < target → move k forward. - If total > target → move l backward. - If total == target → store quadruplet, then skip duplicates for k and l. 4. Return the list of unique quadruplets.

Complexity Analysis: - Time Complexity: O(n^3) Two nested loops O(n^2) + two-pointer search O(n). - Space Complexity: O(1), excluding the space for the result list.

```
class Solution:
    def fourSum(self, arr, target):
        n = len(arr)
        matrix = []
        arr.sort()

        for i in range(n):
            if i > 0 and arr[i] == arr[i - 1]:
                continue
            for j in range(i + 1, n):
                if j > i + 1 and arr[j] == arr[j - 1]:
                    continue
                k = j + 1
                l = n - 1
                while k < l:
                    total = arr[i] + arr[j] + arr[k] + arr[l]
                    if total < target:
                        k += 1
                    elif total > target:
                        l -= 1
                    else:
                        matrix.append([arr[i], arr[j], arr[k], arr[l]])
                        k += 1
                        l -= 1

                        while k < l and arr[k] == arr[k - 1]:
                            k += 1
                        while k < l and arr[l] == arr[l + 1]:
                            l -= 1

        return matrix
```