Combination Sum

This problem is solved efficiently using backtracking (DFS).

Key Observations:

1. We can reuse each candidate unlimited times.
2. Combinations must be unique — order does not matter, so we avoid permutations by:
    - Processing candidates in sorted order.
    - Passing an index to ensure we don't revisit previous numbers.

Steps:

1. Sort candidates (optional, helps pruning).
2. Use recursive dfs (index, current Sum, path):
    - Base case: if current Sum == target, store path.
    - If current Sum > target or index == candidates.size(), return.
3. For each candidate at position i:
    - Include it (stay at i since unlimited usage allowed)
    - Exclude it (move to i+1).

Complexity:

- Time: Potentially exponential ($O(2^n)$) but pruned by target constraint.
- Space: O(target) recursion depth.