Combination Sum II

This problem is similar to Combination Sum I but with two key differences:

1. Each candidate can be used only once.
2. Duplicates must be avoided - multiple combinations with the same numbers in different order must not appear.

Key Points:

- Sort candidates first:
  - Ensures duplicates are adjacent, so we can skip them during iteration.
- Backtracking algorithm:
  - Reduce target at each recursive call.
  - Pass next index (i+1) instead of i (cannot reuse same element).
  - Skip duplicates: if i > start && candidates[i] == candidates[i-1] continue.

Complexity:

- Time: Exponential in worst case ($O(2^n)$) due to subsets generation but reduced with pruning.
- Space: $O(n)$ recursion depth.