**Topic 6.8: Combination Sum II**

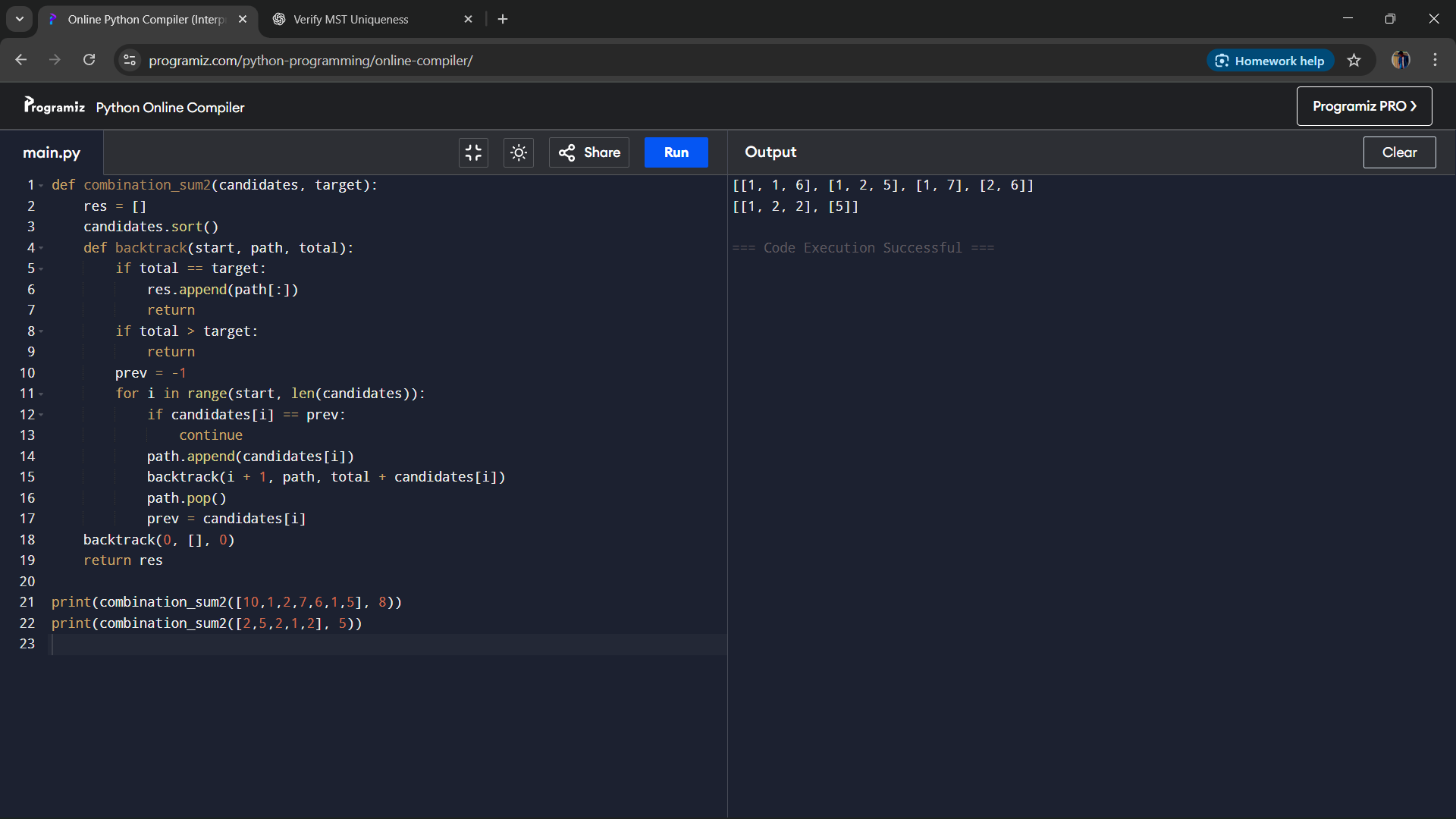
**Question**  
Given a collection of candidate numbers (candidates) and a target number (target), find all unique combinations in candidates where the candidate numbers sum to target.

* Each number in candidates may only be used **once** in the combination.
* The solution set must not contain duplicate combinations.

**Aim**  
To generate all unique combinations of candidate numbers that sum to the target, where each candidate is used at most once, and duplicate solutions are avoided.

**Algorithm**

1. Sort the array candidates to handle duplicates easily.
2. Use backtracking to explore possible combinations:
   * Keep track of the current path and remaining target.
   * If the target reaches zero, store the combination.
   * If the target goes negative, backtrack.
3. Skip over duplicate elements to ensure unique combinations.
4. Ensure that each number is used at most once by moving the recursive index forward.

**Output**

**Result**  
The program successfully generates all unique combinations where each candidate number can be used only once, and duplicate solutions are eliminated.

**Performance Analysis**

* Time Complexity: O(2^n), where n is the number of candidates, due to exploring subsets.
* Space Complexity: O(n) for recursion stack and temporary combinations.