**Topic 6.7: Combination Sum Problem**

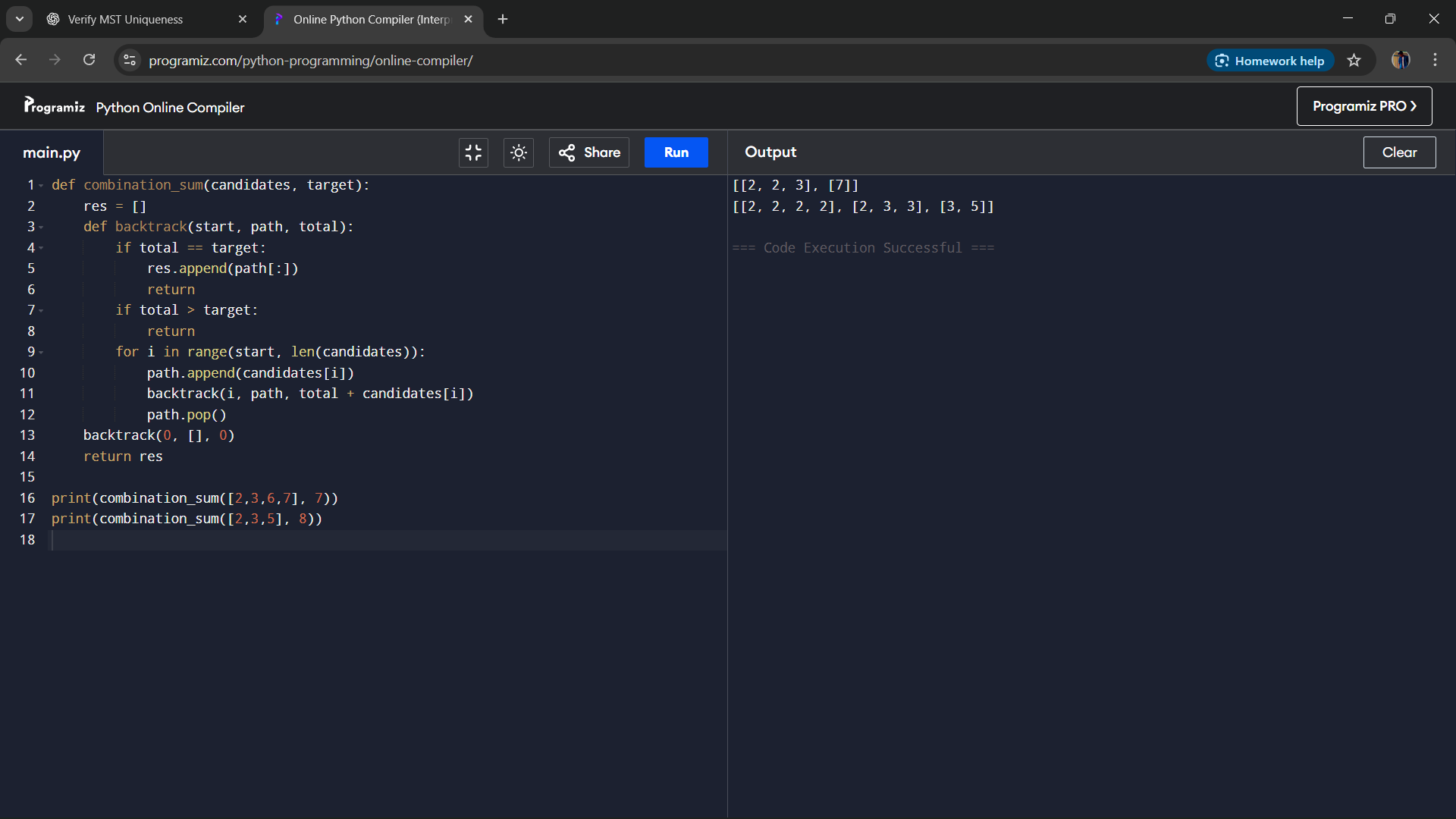
**Question**  
Given an array of distinct integers candidates and a target integer target, return a list of all unique combinations of candidates where the chosen numbers sum to target.

* The same number may be chosen from candidates an unlimited number of times.
* Two combinations are unique if the frequency of at least one chosen number is different.
* The number of unique combinations is guaranteed to be less than 150.

**Aim**  
To generate all unique combinations of candidate numbers that sum to a given target using backtracking.

**Algorithm**

1. Sort the candidates for consistent combination building.
2. Define a recursive function that:
   * Keeps track of the current sum and the current combination.
   * If the sum equals the target, stores the combination.
   * If the sum exceeds the target, backtrack.
3. At each step, try including the current candidate (with unlimited use).
4. Move to the next candidate to avoid duplicate combinations.
5. Return all valid stored combinations.

**Output**

**Result**  
The program successfully finds all unique combinations of numbers that sum to the target, allowing unlimited reuse of elements.

**Performance Analysis**

* Time Complexity: O(2^t), where t is the target value, due to exploring subsets.
* Space Complexity: O(t) for recursion stack and storing combinations.