212. 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. You may return the combinations in any order.The same number may be chosen from candidates an unlimited number of times. Two combinations are unique if the frequency of at least one of the chosen numbers is different.The test cases are generated such that the number of unique combinations that sum up to target is less than 150 combinations for the given input.

Example 1:
Input: candidates = [2,3,6,7], target = 7
Output: [[2,2,3],[7]]
Explanation:
2 and 3 are candidates, and 2 + 2 + 3 = 7. Note that 2 can be used multiple times.
7 is a candidate, and 7 = 7.
These are the only two combinations.
Example 2:
Input: candidates = [2,3,5], target = 8
Output: [[2,2,2,2],[2,3,3],[3,5]]

4. COMBINATION SUM 2:

PROGRAM:-

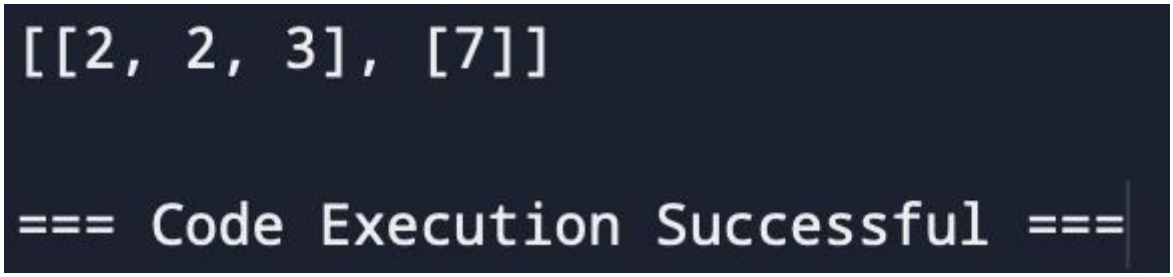
```
from typing import List

def combinationSum(candidates: List[int], target: int) -> List[List[int]]:
    def backtrack(start, target, path):
        if target == 0:
            res.append(path[:])
            return
        for i in range(start, len(candidates)):
            if candidates[i] > target:
                continue
            path.append(candidates[i])
            backtrack(i, target - candidates[i], path)
            path.pop()

    res = []
    candidates.sort()
    backtrack(0, target, [])
    return res

# Test the function with the provided example
candidates = [2, 3, 6, 7]
target = 7
print(combinationSum(candidates, target))
```

OUTPUT:-



TIME COMPLEXITY:-O(n log N)