67. . Combination Sum

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.

PROGRAM:-

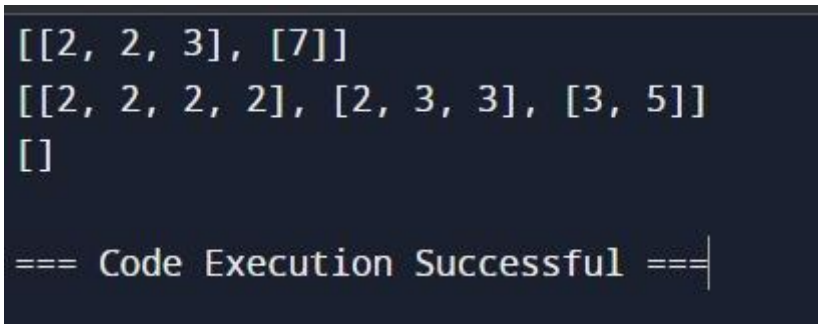
```
def combinationSum(candidates, target):
    def backtrack(start, target, path):
        if target == 0:
            result.append(path)
            return
        if target < 0:
            return
        for i in range(start, len(candidates)):
            backtrack(i, target - candidates[i], path + [candidates[i]])

    result = []
    backtrack(0, target, [])
    return result

# Example usage and output
candidates1 = [2, 3, 6, 7]
target1 = 7
```

print(combinationSum(candidates1, target1)) # Output: [[2, 2, 3], [7]]

candidates2 = [2, 3, 5]
target2 = 8
print(combinationSum(candidates2, target2)) # Output: [[2, 2, 2, 2], [2, 3, 3], [3, 5]]

candidates3 = [2]
target3 = 1
print(combinationSum(candidates3, target3)) # Output: []

OUTPUT:-



TIME COMPLEXITY:-O(2^n)