518. Coin Change 2

You are given an integer array `coins` representing coins of different denominations and an integer `amount` representing a total amount of money.

Return *the number of combinations that make up that amount*. If that amount of money cannot be made up by any combination of the coins, return `0`.

You may assume that you have an infinite number of each kind of coin.

The answer is **guaranteed** to fit into a signed **32-bit** integer.

Example 1:

```
Input: amount = 5, coins = [1,2,5]
Output: 4
Explanation: there are four ways to make up the amount:
5=5
5=2+2+1
5=2+1+1+1
5=1+1+1+1+1
```

Example 2:

```
Input: amount = 3, coins = [2]
Output: 0
Explanation: the amount of 3 cannot be made up just with coins of 2.
```

Example 3:

```
Input: amount = 10, coins = [10]
Output: 1
```

```
def change(self, amount: int, coins: List[int]) -> int:
        dp = [0]*(amount+1)
        dp[0] = 1
        if amount == 0:
            return 1
        for coin in coins:
            for i in range(1,len(dp)):
                if i-coin>=0:
```

```
                    dp[i] = dp[i]+dp[i-coin]

        return dp[amount]
```