322. Coin Change

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

Return *the fewest number of coins that you need to make up that amount*. If that amount of money cannot be made up by any combination of the coins, return `-1`.

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

Example 1:

```
Input: coins = [1,2,5], amount = 11
Output: 3
Explanation: 11 = 5 + 5 + 1
```

Example 2:

```
Input: coins = [2], amount = 3
Output: -1
```

Example 3:

```
Input: coins = [1], amount = 0
Output: 0
```

Example 4:

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

Example 5:

```
Input: coins = [1], amount = 2
Output: 2
```

```
import sys
class Solution:
    def coinChange(self, coins: List[int], amount: int) -> int:
        dp = [0]*(amount+1)
        dp[0] = 0
        if amount == 0:
            return 0
        for i in range(1,len(dp)):
            temp = []
            for coin in coins:
                if i==coin:
                    temp.append(1)
                else:
                    if i-coin>=0:
                        if dp[i-coin]!=0:
                            temp.append(dp[i-coin]+1)
            if len(temp):
                dp[i] = min(temp)
        return dp[amount] if dp[amount]!=0 else -1
```