322. Coin Change
Medium

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.

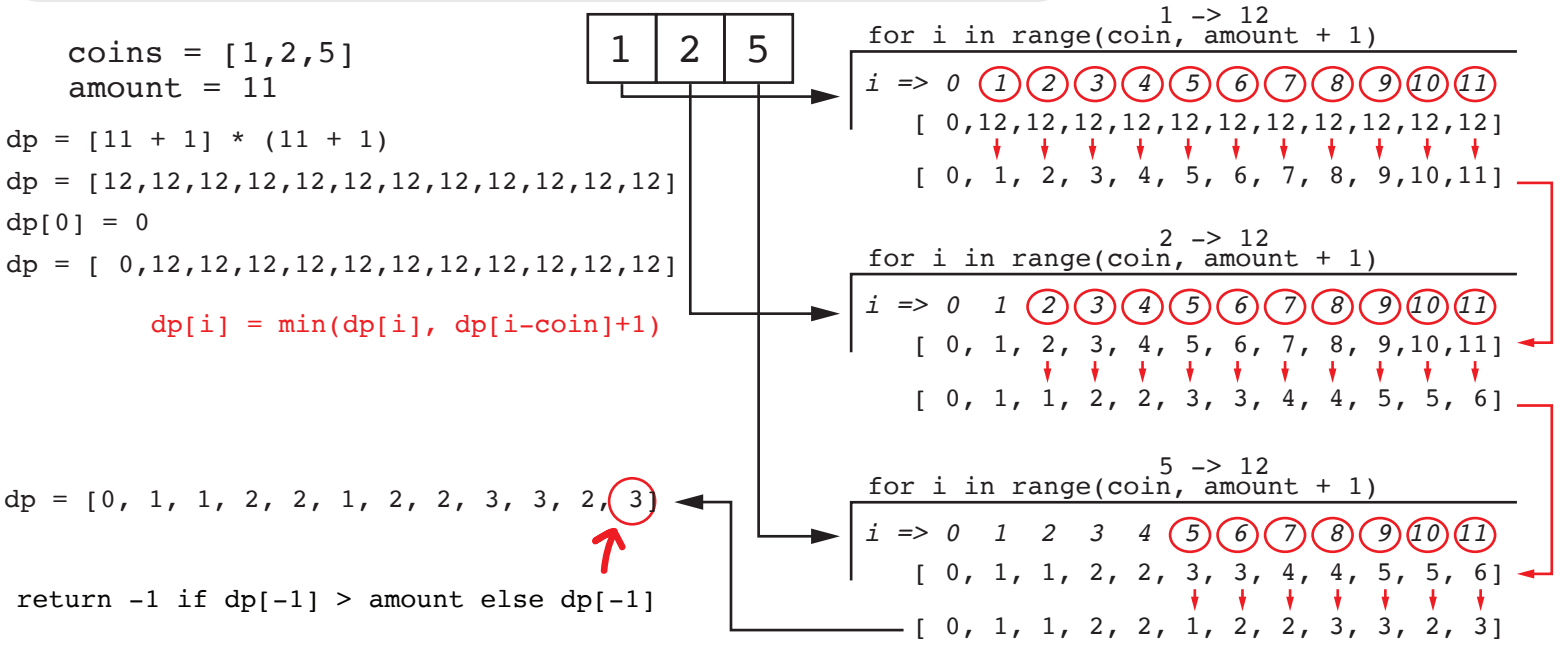
You may assume that you have an infinite number of each kind of coin.

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

```python
class Solution:
    def coinChange(self, coins: List[int], amount: int) -> int:
        '''Bottom-up tabulation'''
        dp = [amount + 1] * (amount + 1)
        dp[0] = 0
        for coin in coins:
            for i in range(coin, amount+1):
                dp[i] = min(dp[i], dp[i-coin]+1)
        return -1 if dp[-1] > amount else dp[-1]
```




```python
class Solution:
    def coinChange(self, coins: List[int], amount: int) -> int:
        '''Top-down memoization'''
        def coinChangeInner(rem, cache):
            if rem < 0:
                return math.inf
            if rem == 0:
                return 0
            if rem in cache:
                return cache[rem]

            cache[rem] = min(coinChangeInner(rem-x, cache) + 1 for x in coins)
            return cache[rem]

        ans = coinChangeInner(amount, {})
        return -1 if ans == math.inf else ans
```