322. Coin Change

You are given coins of different denominations and a total amount of money amount. Write a function to compute 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.

Example 1:

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

```
class Solution {
    public int coinChange(int[] coins, int amount) {
        int[] dp = new int[amount + 1]; // 0...amount
        int max = amount + 1;
        Arrays.fill(dp, max);

        // bottom up iteration
        // F(i) = minF(i - cj) + 1, while j = 0...n-1
        // dynamic programming
        dp[0] = 0;

        for (int i = 1; i <= amount; i++) {
            for (int j = 0; j < coins.length; j++) {
                if (coins[j] <= i) {
                    dp[i] = Math.min(dp[i], dp[i-coins[j]] + 1);
                }
            }
        }

        return dp[amount] > amount ? -1 : dp[amount];
    }
}
```