

Coin Change

Min coins for a target — unbounded DP ■

● MEDIUM

■ Hook

Given coin denominations and a target amount, find the minimum number of coins needed. Coins can be reused (unbounded). This is the canonical **unbounded knapsack / DP on amounts** problem.

■ Core Intuition

$dp[a]$ = min coins to make amount a . For each coin c : $dp[a] = \min(dp[a], dp[a-c] + 1)$. Build from amount 0 up to target. $dp[0]=0$ (need 0 coins for amount 0).

■ DP Array — coins=[1,5,6,9], amount=11

```
dp[0]=0  (0 coins for amount 0)
dp[1]=1  (1×1)
dp[2]=2  (2×1)
dp[3]=3  (3×1)
dp[4]=4  (4×1)
dp[5]=1  (1×5)
dp[6]=1  (1×6)
dp[7]=2  (1×6+1×1 or 1×5+2×1... min=2)
dp[8]=2  (6+1+1? no: 5+3×1=4, or 6+2×1=3? min=3... recalc)
dp[9]=1  (1×9)
dp[10]=2 (9+1 or 5+5)
dp[11]=2 (9+1×? wait: 5+6=11 → 2 coins!) ← ★
★ ANSWER = 2  (coins 5+6=11)
```

■ Code

```

public int coinChange(int[] coins, int amount) {
    int[] dp = new int[amount+1];
    Arrays.fill(dp, amount+1); // init to 'infinity'
    dp[0] = 0;
    for (int a=1; a<=amount; a++) {
        for (int c : coins) {
            if (c <= a)
                dp[a] = Math.min(dp[a], dp[a-c] + 1);
        }
    }
    return dp[amount] > amount ? -1 : dp[amount];
}

```

■ TIME

$O(\text{amount} \times \text{coins})$

■ SPACE

$O(\text{amount})$

■ Initialize with amount+1 not Integer.MAX_VALUE

$dp[a-c]+1$ would overflow if $dp[a-c]$ is `MAX_VALUE`. Use `amount+1` as 'infinity' — it's guaranteed to be larger than any valid answer (you can't need more than 'amount' coins of denomination 1).

■ Practice

#	Problem	Diff	Pattern
1	Coin Change (LC 322)	Medium	Unbounded DP
2	Coin Change II (LC 518)	Medium	Count ways
3	Perfect Squares (LC 279)	Medium	Same pattern
4	Minimum Cost for Tickets (LC 983)	Medium	DP on days
5	Combination Sum IV (LC 377)	Medium	Count ordered combos

- ✓ $dp[a]$ = min coins to make amount a
- ✓ For each coin c : $dp[a] = \min(dp[a], dp[a-c]+1)$
- ✓ Init $dp[0]=0$, rest = `amount+1` (not `MAX_VALUE`!)
- ✓ Return -1 if $dp[\text{amount}] > \text{amount}$
- ✓ Time: $O(\text{amount} \times |\text{coins}|)$ | Space: $O(\text{amount})$

■ MENTAL MODEL — Remember This Forever

Think of $dp[a]$ as 'the cheapest way to buy exactly \$a at a vending machine'. For each denomination, you 'try inserting that coin' and check if the remaining amount is cheaper than what you've found so far. Build up from \$0 to \$amount.