Tutorial: Prateleira Dourada

1 Solução do Problema

O problema de organizar os livros para preencher exatamente a prateleira é uma aplicação clássica do problema da **Soma de Subconjuntos** (*Subset Sum*), que é uma variação do problema da Mochila (0/1 *Knapsack*). Podemos resolvê-lo de forma eficiente utilizando **programação dinâmica**. A ideia é rastrear todos os comprimentos parciais possíveis que conseguimos formar com as combinações de livros avaliados até o momento.

1.1 Definição do Subproblema

Seja T o comprimento total da prateleira. Definimos o nosso estado da programação dinâmica como um vetor booleano:

$$dp[j] = 1 \text{ se for possível formar o comprimento exato } j \text{ com algum subconjunto dos livros.}$$

O nosso objetivo final é descobrir se, após testar todas as combinações de livros, o estado $dp[T]$ se torna verdadeiro.

1.2 Função de Transição

Para atualizar os nossos estados, iteramos e avaliamos um livro de cada vez. Seja x a espessura do livro atual. A lógica é simples: se já descobrimos que é possível formar uma pilha de livros com espessura exata j (ou seja, $dp[j] == 1$), então, ao colocar esse novo livro na pilha, também seremos capazes de alcançar a espessura $j + x$.

A transição, pode ser descrita como:

$$dp[j + x] = dp[j + x] \vee dp[j]$$

Detalhe crucial de implementação: Para garantir que o bibliotecário use cada livro **no máximo uma vez**, precisamos percorrer as capacidades de trás para frente (começando em $T - x$ e diminuindo até 0). Se iterássemos da esquerda para a direita, um livro poderia ser considerado mais de uma vez, o que não é o correto nesse problema.

1.3 Casos Base

O caso base representa a situação em que não selecionamos livro algum. A soma das espessuras de uma prateleira vazia é zero. Portanto, é sempre possível "formar" o comprimento 0:

$$dp[0] = 1$$

Todos os outros comprimentos no vetor de 1 a T nascem como 0 (falsos/impossíveis), pois ainda precisamos descobrir se eles podem ser alcançados utilizando os livros da pilha.