Tutorial: O problema das Flores

O problema pode ser resolvido utilizando uma abordagem de **programação dinâmica**, onde calculamos o número de maneiras válidas de preencher o canteiro até cada posição, levando em conta a cor da última flor e o tamanho do bloco atual de flores consecutivas.

A restrição principal é que não podem existir mais do que m flores consecutivas da mesma cor. Isso significa que, ao adicionar uma nova flor, precisamos garantir que a sequência resultante ainda seja válida, ou seja, o número de flores consecutivas da mesma cor não ultrapasse o limite imposto.

Definimos a DP da seguinte forma:

$$dp[i][c] = \text{número de maneiras válidas de formar um arranjo de comprimento } i \text{ cuja última flor tem cor } c$$

onde $c = 0$ representa a cor `V` (vermelha) e $c = 1$ representa a cor `B` (branca).

Para calcular $dp[i][c]$, consideramos todos os blocos possíveis de flores consecutivas da mesma cor que terminam na posição i. Ou seja, podemos ter adicionado 1, 2, ..., até m flores consecutivas da cor c, desde que o arranjo anterior termine com a cor oposta.

A função de transição é dada por:

$$dp[i][c] = \sum_{k=1}^{m} dp[i-k][1-c]$$

para todo $i \geq k$, pois estamos adicionando um bloco de k flores da cor c sobre um arranjo válido de comprimento $i - k$ que termina com a cor oposta.

As condições base são:

$$dp[0][0] = dp[0][1] = 1$$

representando o caso vazio, que conta como uma configuração válida inicial para permitir as transições.

O resultado final é a soma de todos os arranjos válidos que terminam em qualquer cor:

$$\text{Resposta} = dp[n][0] + dp[n][1]$$

Essa solução possui complexidade de tempo $O(n \times m)$, já que para cada posição i consideramos até m comprimentos de blocos consecutivos possíveis. O espaço necessário é $O(n)$, podendo ser otimizado para $O(m)$ se apenas os estados recentes forem mantidos.