---
MULTIPLICAÇÃO DE CADEIAS DE MATRIZES (Cormen, §15.2)
---

O problema da multiplicação de cadeias de matrizes é o de, dado um vetor D[0..n] de naturais positivos, encontrar uma parentetização para a cadeia de matrizes

A_1 * A_2 * A_3 * ... * A_n, onde A_i é uma matriz D[i-1] x D[i] ∀i,

que leve ao menor número total de operações numéricas para a computação do produto acima, supondo que a multiplicação de uma matriz "m x n" por uma "n x o" custa "m*n*o" operações numéricas. Na versão simplificada do problema, deseja-se apenas descobrir o número de operações numéricas de uma parentetização ótima.

O seguinte é um algoritmo de divisão-e-conquista direto para o problema:

```
==========================================================================
Algoritmo: cad_matr_dq
Entrada:   vetor D[0..n] de números naturais e índices "i" e "f".
Saída:     um número natural.
--------------------------------------------------------------------------
1. SE i = f
2. | RETORNE 0.
3. min_mult := +infinito
4. PARA k DE i A f-1
5. | num_mult := cad_matr_dq(D,i,k) + cad_matr_dq(D,k+1,f) + D[i-1]*D[k]*D[f]
6. | min_mult := min( min_mult, num_mult )
7. RETORNE min_mult.
==========================================================================
```

É fácil verificar que o algoritmo acima realiza chamadas recursivas repetidas. O algoritmo de programação dinâmica abaixo resolve esse problema:

```
==========================================================================
Algoritmo: cad_matr_pd
Entrada:   vetor D[0..n] de números naturais.
Saída:     um número natural.
--------------------------------------------------------------------------
01. PARA a DE 1 A n
02. | S[a][a] := 0   // Matriz que armazena as SOLUÇÕES dos subproblemas.
03. PARA t DE 2 A n  // t: tamanho da cadeia de matrizes em questão
04. | PARA i DE 1 A n -t +1
05. | | min_mult := +infinito , f := i +t -1
06. | | PARA k DE i A f-1
07. | | | num_mult := D[i-1]*D[k]*D[f] + S[i][k] + S[k+1][f]
08. | | | min_mult := mín( min_mult, num_mult)
09. | | S[i][f] := min_mult
10. RETORNE S[1][n].
==========================================================================
```

1. EXERCÍCIOS:

   a) IMPORTANTE: escreva uma extensão do algoritmo acima que retorne uma solução completa, isto é, alguma descrição da parentetização em si da cadeia de matrizes, e não apenas o número correspondente de multiplicações numéricas. A complexidade assintótica do tempo de execução do algoritmo não deve mudar.

   b) O algoritmo acima é $\theta(n^3)$ ou $o(n^3)$?

   c) Escreva uma versão memoizada do algoritmo acima.

---
LEITURA ADICIONAL
---

Existe um algoritmo O(n*lg n) para o problema; veja a respeito na Wikipédia

http://en.wikipedia.org/wiki/Matrix_chain_multiplication#Hu_.26_Shing_.281981.29

ou diretamente nos artigos correspondentes:

http://infolab.stanford.edu/pub/cstr/reports/cs/tr/81/875/CS-TR-81-875.pdf
http://dx.doi.org/10.1137/0211028
http://dx.doi.org/10.1137/0213017

---
O PROBLEMA DA MOCHILA 0-1 (Cormen, §16.2, exercícios)
---

No problema da mochila 0-1, temos "n" objetos, o i-ésimo de peso p[i] e valor v[i], e desejamos escolher alguns desses objetos, de forma que a soma dos pesos dos objetos escolhidos seja <= P, e de forma que a soma dos valores dos objetos escolhidos seja a maior possível. Os valores dos objetos são números reais, mas os pesos dos objetos e o valor "P" são todos naturais positivos.

2. EXERCÍCIOS:

a) Mostre que escolher os objetos pela razão valor/peso, em ordem, do objeto de maior razão para o de menor razão, nem sempre leva a uma solução ótima.

b) IMPORTANTE: Tente escrever um algoritmo de programação dinâmica para o problema. Considere, por exemplo, resolver o problema para uma mochila de capacidade "P" já tendo resolvido o problema para mochilas de capacidades menores.