Rio de Janeiro, 27 de Junho de 2011.
PROVA FINAL DE ANÁLISE DE ALGORITMOS
PROFESSOR: EDUARDO SANY LABER
DURAÇÃO: 2:00h

1. (1.5pt) Resolva as equações de recorrência abaixo

a) $T(n) = 4T(n/2) + n^2$, para $n > 1$ e $T(1) = 1$.

b) $T(n) = 2T(n/2) + n$, para $n > 1$ e $T(1) = 1$.

2.(1.0pt) Considere o procedimento abaixo. Na linha (*), $L_i \cdot L_j$ indica a lista obtida ao contacatenar as listas L_i e L_j. Assuma que essa concatenação pode ser realizada em $O(1)$. Seja $T(n)$ a complexidade de tempo do algoritmo para uma lista de tamanho n, determine a equação de recorrência para $T(n)$. Não é necessário resolvê-la.

MULT(L: lista de números)
    Se $|L| \leq 3$
        Return 1
    Senão
        Divida L em três sublistas L_1, L_2 e L_3 de mesmo tamanho.
        $m_1 \leftarrow$ média dos números de L_1
        $m_2 \leftarrow$ média dos números de L_2
        $m_3 \leftarrow$ média dos números de L_3
        Return $m_1 + m_2 + m_3 +$ MULT($L_1 \cdot L_2$) + MULT($L_1 \cdot L_3$) + MULT($L_2 \cdot L_3$) (*)
    Fim Se

3. (2.0pt) Seja G um grafo direcionado, fortemente conexo, com pesos nas arestas e seja $c(u, v)$ o custo do caminho de custo mínimo entre os nós u e v.

a) Assuma todos os pesos das arestas são números reais não-negativos. Como seria um algoritmo para encontrar a média

$$\frac{\sum_{u,v \in V} c(u, v)}{n(n-1)}$$

entre os caminhos de custo mínimo? Analise sua complexidade.

b) Assuma que todas as arestas tem peso igual a 1. Como poderíamos resolver o item anterior de forma mais eficiente? Por que?

4. (2pt) Considere os pseudo-códigos abaixo.

a) Faça a análise assintótica do procedimento abaixo, ou seja, determine uma função $f(n)$ tal que $T(n) = \theta(f(n))$.

Pseudo1

```
t ← 0
Cont ← 1
Para  i=1 até n
    Cont ← cont+1
Fim Para
Enquanto cont ≥ 1
    Cont ← cont/2
    Para j = 1 a n
        t++
    Fim Para
Fim Enquanto
```

(b) Faça a análise assintótica do procedimento abaixo, ou seja, determine uma função $f(n)$ tal que $T(n) = \theta(f(n))$.

Pseudo2

```
t ← 1
Enquanto t < n
    t ← 2t
    Para i = 1 a t
        cont ← 0
    Fim Para
Fim Enquanto
```

5.(3.0pt) Dada duas cadeias $X = x_1x_2...x_n$ e $Y = y_1y_2...y_m$, queremos encontrar o tamanho da maior subcadeia comum entre X e Y, ou seja, o maior valor de k tal que existem índices i e j satisfazendo $x_ix_{i+1}...x_{i+k-1} = y_jy_{j+1}...y_{j+k-1}$. Como exemplo, a maior subcadeia comum entre X=ABABD e Y=AABABBA é ABAB. Neste caso, $k = 4, i = 1$ e $j = 2$. Seja X_r a subcadeia de X com os r primeiros símbolos de X e Y_s é a subcadeia de Y com os s primeiros símbolos de Y. Por exemplo, se X=ABABD, então $X_3 = ABA$ e $X_2 = AB$.

Definimos $OPT(r, s)$ como o **tamanho** da maior subcadeia comum entre X_r e Y_s que **inclui** x_r e y_s. Sabemos que se $x_r = y_s$ e $r, s \geq 1$, então $OPT(r, s) = 1 + OPT(r-1, s-1)$. Se $x_r \neq y_s$ ou $r = 0$ ou $s = 0$, então $OPT(r, s) = 0$,

a) Descreva uma algoritmo **recursivo e eficiente** para calcular $OPT(n, m)$. Qual a complexidade do algoritmo?

b) Assuma que recebemos uma matriz $M[0..n, 0..m]$ preenchida com os valores de $OPT(,)$, ou seja, $M[i, j] = OPT(i, j)$ para todo i, j. Como podemos obter a **maior subcadeia comum** entre X e Y que inclui x_r e y_s a partir da matriz M? Note que não está sendo pedido o tamanho da maior subcadeia comum.

c) Assuma que recebemos uma matriz $M[0..n, 0..m]$ preenchida com os valores de OPT, ou seja, $M[i, j] = OPT(i, j)$ para todo i, j. Como podemos obter o **tamanho** da maior subcadeia comum entre X e Y? Qual a complexidade do procedimento?

6.(2.0pt). Seja G um grafo não direcionado com n vértices e m arestas.

a) Como seria um algoritmo para testar em tempo polinomial se existe uma cobertura para G de tamanho 3? Não é necessário se preocupar com a eficiência do algoritmo, basta garantir que ele é polinomial em n e m.

b) Analise a complexidade do algoritmo proposto.