Introdução a Análise de Algoritmos
2º semestre de 2014 - Turmas 02 e 14
Lista de exercícios 2

1. Usando as definições das notações assintótica Θ, O e Ω, mostre que:

   a) $\frac{n^2}{1000} = O(n^2)$
   b) $5n^3 + 1000n^2 = O(n^4)$
   c) $1000n^2 = \Omega(n^2)$
   d) $n^4 - 25n^2 = \Omega(n^3)$
   e) $2^{16}n^2 \neq \Omega(n^3)$
   f) $n^3 - 10n^2 \neq O(n^2)$
   g) $4n^3 - 300n^2 + 7000n = \Theta(n^3)$
   h) $n^2 + nlgn = \Theta(n^2)$

2. Resolva as seguintes recorrências (assuma, para o item (d), que n é uma potência de 4, e para os itens (e) e (f), que n é uma potência de 2):

   a)
   $$T(n) = \begin{cases} a, & \text{se } n = 1 \\ T(n-1) + b, & \text{se } n > 1 \end{cases} \quad (1)$$

   b)
   $$T(n) = \begin{cases} 0, & \text{se } n = 0 \\ k, & \text{se } n = 1 \\ T(n-2) + k, & \text{se } n > 1 \end{cases} \quad (2)$$

   c)
   $$T(n) = \begin{cases} 100, & \text{se } n = 1 \\ T(n-1) + 3n, & \text{se } n > 1 \end{cases} \quad (3)$$

   d)
   $$T(n) = \begin{cases} 1, & \text{se } n = 1 \\ 4T(\frac{n}{4}) + n, & \text{se } n > 1 \end{cases} \quad (4)$$

   e)
   $$T(n) = \begin{cases} 1, & \text{se } n = 1 \\ 2T(\frac{n}{2}) + n^3, & \text{se } n > 1 \end{cases} \quad (5)$$

f)

$$T(n) = \begin{cases} 1, & \text{se } n = 1 \\ T(\frac{n}{2}) + n, & \text{se } n > 1 \end{cases} \quad (6)$$

3. Faça a análise da complexidade dos algoritmos implementados no exercício 1 da lista 1. Para isso: (i) determine a equação de recorrência que descreva o tempo de execução de cada algoritmo implementado; (ii) resolva a recorrência para determinar a complexidade assintótica do mesmo.

4. Escreva um método iterativo (o uso de instruções de repetição está liberado) que recebe um vetor `c` de caracteres, e imprima todos os arranjos de tamanho 2 que podem ser formados com os caracteres presentes em `c`. Por exemplo, para `c` = { '`a`', '`f`' }, seu programa deve imprimir a seguinte saída:

   ```
   aa
   af
   fa
   ff
   ```

5. Idem a questão anterior, mas que imprima todos os arranjos com tamanho igual a 3.

6. Escreva agora um método que recebe um vetor `c` de caracteres, um valor inteiro `n` e imprima todos os arranjos de tamanho `n` que podem ser formados pelos caracteres presentes em `c`. Observe que, para este exercício, um algoritmo iterativo com um número fixo de laços encadeados não é viável (justamente a forma mais direta de resolver as questões 4 e 5). Embora seja possível elaborar um algoritmo iterativo, o mais recomendado é que vocês implementem uma versão recursiva. Apesar de este não ser exatamente um problema de tentativa e erro (pois não estamos interessados em encontrar uma solução dentre o universo de todas as tentativas possíveis), a estrutura do algoritmo recursivo será semelhante ao dos algoritmos estudados para os problemas de tentativa e erro uma vez que gerar todos os arranjos possíveis é um problema equivalente a gerar todas as tentativas possíveis a serem analisadas em um problema de tentativa e erro.

7. Faça a análise do algoritmo desenvolvido na questão 6.