2ª Lista de Exercícios de Introdução à Análise de Algoritmos
Prof. Glauber Cintra – Entrega: 20/jun/2011

Esta lista deve ser feita por grupos de no **mínimo** 3 e no **máximo** 4 alunos.

1) **(1,5 pontos)** Resolva as seguintes fórmulas de recorrência:
   a) $T(n) = T(n-1) + n^2$, $T(1) = 1$
   b) $T(n) = T(n/4) + n$, $T(1) = 1$
   c) $T(n) = 2T(n-1) + n$, $T(1) = 1$

2) Considere o seguinte algoritmo:

```
Algoritmo enigma
    Entrada: n (natural)
    se n = 0
        devolva 0
    senão
        devolva enigma(n – 1) + n + n – 1
```

   a) **(0,5 pontos)** Simule o cálculo de *enigma*(4), exibindo o parâmetro de entrada e o valor retornado por cada chamada ao algoritmo *enigma*.
   b) **(1 ponto)** Determine a complexidade de tempo e de espaço do algoritmo *enigma* (mostre os cálculos realizados para determinar tais complexidades). Esse algoritmo é eficiente? É de cota inferior? Para que serve esse algoritmo?
   c) **(1 ponto)** Prove que o algoritmo é correto.

3) **(2 pontos)** Indique quais são o melhor caso e o pior caso do algoritmo abaixo e sua região crítica. Qual a complexidade de tempo desse algoritmo no pior e no melhor caso? Qual a complexidade de espaço desse algoritmo? O algoritmo é eficiente? Pesquise e diga se este algoritmo é de cota inferior? Justifique suas respostas.

```
Algoritmo Mediana
    Entrada: um vetor v com n posições indexadas a partir do 1
    Saída: a mediana do vetor v
    para i = 2 até n
        j = i
        pivo = v[ i ]
        enquanto j > 1 e v[ j - 1 ] > pivo
            v[ j ] = v[ j - 1 ]
            j = j - 1
        v[ j ] = pivo
    se n é impar
        devolva v[(n+1)/2]
    senão
        devolva (v[n/2] + v[n/2+1])/2
```

4) **(2 pontos)** Escreva um algoritmo recursivo para somar os valores contidos num vetor de números que seja de cota inferior. Prove que seu algoritmo é correto e é de cota inferior.

5) **(1,5 pontos)** Discorra sobre as principais técnicas de análise amortizada: método da agregação, método potencial e método contábil.

6) **(1,5 pontos)** Faça a análise amortizada do algoritmo abaixo, utilizando pelo menos dois dos métodos citados na questão 5, e determine o *custo amortizado* de cada chamada ao algoritmo. Informe também o custo total para realizar *n* chamadas consecutivas ao algoritmo, supondo que inicialmente a pilha está vazia.

```
Algoritmo empilhaSemOverflow
   Entrada: uma pilha p (implementada num vetor) e um valor x.
   Saída: insere x em p.

   Se p.topo = p.tamanho - 1
      duplique p   // requer tempo linear no tamanho da pilha
   p.topo = p.topo + 1
   p.vetor[p.topo] = x
```