Lista de Exercícios 1
Técnicas de Programação Avançada

Prof. Dr. Jefferson O. Andrade
Ifes – Campus Serra

2019/2

Sumário

1 Introdução

As questões enunciadas na seção seguinte (Seção 2) se referem ao conteúdo visto em sala de aula, que corresponde aos capítulos de 1 a 5 do livro texto.[1]

2 Questões

1. Para cada função $f(n)$ e tempo t na tabela abaixo, determine o maior tamanho n de problema que pode ser resolvido no tempo t, assumindo que o algoritmo para resolver o problema demora $f(n)$ microsegundos.

	1 segundo	1 minuto	1 hora	1 dia	1 mês	1 ano	1 século
$\lg n$							
$\sqrt{n}$							
n							
$n \lg n$							
n^2							
n^3							
2^n							
$n!$							

2. Considere a seguinte definição para o **problema de pesquisa:**

   Entrada: Uma sequência de n números $A = \langle a_1, a_2, \ldots, a_n \rangle$, e um valor v.

[1] Thomas H. Cormen, *et al.* "Altoritmos: Teoria e Prática." 3ª ed. Editora Campus, 2012.

Saída: Um índice i tal que $v = A[i]$, ou o valor especial NIL (significando "nulo"), se v não ocorrer em A.

Escreva o pseudocódigo para **pesquisa linear** que faça a varredura da sequência procurando por v. Usando uma invariante de loop prove que seu algoritmo é correto. Certifique-se de que sua invariante de loop satisfaz às três propriedades necessárias.

3. Considere a ordenação de n números armazenados no arranjo A, localizando primeiramente o menor elemento de A e permutando este elemento com o elemento contido em $A[1]$. Em seguida, encontre o segundo menor elemento de A e o troque pelo elemento contido em $A[2]$. Repita esta operação para os primeros $n-1$ elementos de A. Este algoritmo é conhecido como **ordenação por seleção**, ou *selection sort*.
   (a) Escreva o pseudocódigo para o algoritmo **selection sort**.
   (b) Que invariante de loop este algoritmo mantém?
   (c) Por que ele só precisa se executado para os primeiros $n-1$ elementos?
   (d) Forneça os tempos de execução do melhor caso e do pior caso da ordenação por seleção em notação Θ.

4. Considerando o *problema de pesquisa* (Exercício 2) observe que, se a sequência A estiver ordenada, podemos comparar o ponto médio da sequência com v e eliminar metade da sequência do restante da pesquisa. A **pesquisa binária** é um algoritmo que repete esse procedimento, dividindo ao meio a porção restante da sequência a cada vez. Escreva o pseudocódigo, sendo iterativo ou recursivo, para a pesquisa binária. Demonstre que o tempo de execução no pior caso da pesquisa binária é $\Theta(\lg n)$.

5. A ordenação por inserção (*InsertionSort*) pode ser expressa sob a forma de um procedimento recursivo, como explicado a seguir. Para ordenar o vetor $A[1..n]$, ordenamos recursivamente $A[1..(n-1)]$ e depois inserimos $A[n]$ no arranjo ordenado $A[1..(n-1)]$.
   (a) Escreva o pseudocódigo para a versão recursiva do *InsertionSort*.
   (b) Escreva a função de custo, $T(n)$, da versão recursiva do *InsertionSort* como uma recorrência.
   (c) Escreva a forma fechada, i.e., sem recorrência, da função de custo do *InsertionSort* recursivo. Justifique a forma como obteve este resultado.

6. Resolva a recorrência $T(n) = 2T\left(\sqrt{n}\right)$, fazendo uma troca de variáveis. Sua solução deve ser assintoticamente restrita. Não se preocupe em saber se os valores são integrais. Use a estratégia de "expandir, conjecturar, e demonstrar".

7. Use uma árvore de recursão com o objetivo de fornecer uma solução assintoticamente restrita para a recorrência $T(n) = T(n-a) + T(a) + cn$, onde $a \geq 1$ e $c > 0$ são constantes.

8. Use o método mestre para fornecer limites assintóticos restritos para as recorrências a seguir.
   (a) $T(n) = 4T(n/2) + n$
   (b) $T(n) = 4T(n/2) + n^2$
   (c) $T(n) = 4T(n/2) + n^3$

9. O método mestre pode ser aplicado à recorrência $T(n) = 4T(n/2) + n^2 \lg n$? Por que, ou por que não? Em qualquer dos dois casos, forneça um limite superior assintótico para essa recorrência.

10. O professor Kelp decide escrever um procedimento que produzirá, ao acaso, qualquer permutação de um arranjo, exceto pela permutação de identidade. Ele propõe o seguinte procedimento:

```
procedure PermuteWithoutIdentity(A)
    n ← length(A)
    for i ← 1..n do
        j ← RANDOM(i + 1, n)
        Trocar A[i] ↔ A[j]
    end for
end procedure
```

    (a) Este procedimento faz o que o professor Kelp deseja?

    (b) Suponha que, ao invés de trocar o elemento $A[i]$ por um elemento aleatório no subarranjo $A[(i+1)..n]$, nós o trocamos por um elementos aleatório de qualquer lugar do arranjo:

```
procedure PermuteWithAll(A)
    n ← length(A)
    for i ← 1..n do
        j ← RANDOM(1, n)
        Trocar A[i] ↔ A[j]
    end for
end procedure
```

    Esse código produz uma permutação aleatória uniforme? Por que, ou por que não?

3 Entrega

A resolução das questões deve ser preparada utilizando-se o sistema LaTeX.

Há excelentes opções de implementações do LaTeX para microcomputadores, tais como o MiKTeX, para windows, ou o TeX Live, para Linux. Há ainda algumas possibilidades de utilizar o LaTeX on-line, como o Overleaf, por exemplo.

A resposta deve usar como base o modelo disponibilizado pelo professor.

Para esta lista, apenas arquivo PDF gerado à partir do fonte em LaTeX deve ser enviado para a atividade no AVA.