Estrutura de Dados - 2o. período de 2014

Primeira Avaliação à Distância

1. (1,0) Escreva as seguintes funções em notação O: $n^{10} + 2^{\sqrt{n}}$, $n^3 + n^2 \log n$, 754, $\frac{1}{\sqrt{n}} + n$, $1^n + n^2$.

   Resposta: $n^{10} + 2^{\sqrt{n}} = O(2^{\sqrt{n}})$, $n^3 + n^2 \log n = O(n^3)$, $754 = O(1)$, $\frac{1}{\sqrt{n}} + n = O(n)$, $1^n + n^2 = O(n^2)$.

2. Para cada item abaixo, responda "certo" ou "errado", justificando:

   a. (0,5) Se a complexidade de caso melhor caso de um algoritmo for $O(n^3)$, então o pior caso deste algoritmo é $\Omega(n^3)$.

      Resposta: Errado. Considere um algoritmo cujo melhor caso seja dado por $n + 3$ e o pior caso por $n^2 + 2n$, por exemplo. Podemos afirmar que o melhor caso é $O(n^3)$, mas o pior caso não é $\Omega(n^3)$.

   b. (0,5) Se a complexidade de caso médio de um algoritmo for $\Theta(f)$, então o algoritmo é $\Theta(f)$.

      Resposta: Errado. Para afirmarmos que um algoritmo é $\Theta(f)$, esta afirmação tem que ser válida para todas as possíveis entradas do algoritmo. Seja um algoritmo cujo caso médio seja dado por $f = n^2$ e seu pior caso seja $g = n^3$. Apenas o caso médio deste algoritmo é $\Theta(n^2)$, pois seu pior caso é $\Theta(n^3)$.

   c. (0,5) Seja P um problema com limite inferior $n \log n$. Se um algoritmo resolve P em tempo $\Theta(n \log n)$, então este algoritmo é ótimo.

      Resposta: Certo. Se o algoritmo resolve P em $\Theta(n \log n)$, então seu pior caso também é $\Theta(n \log n)$. Logo, o algoritmo é ótimo.

3. Considere a seguinte lista ordenada: 1, 3, 5, 7, 9, 11, 15, 17, 19, 21, 23. Utilizando busca binária, determine:

   a. (0,5) Um elemento cuja busca resulte em um número mínimo de comparações.

      Resposta: O elemento 11, pois resulta em apenas 1 comparação.

   b. (0,5) Um elemento **pertencente** à lista cuja busca resulte em um número máximo de comparações. Determine quais comparações foram efetuadas.

      Resposta: O elemento 23, que resulta em 4 comparações.

   c. (0,5) Um elemento **não pertencente** à lista cuja busca resulte em um número máximo de comparações. Determine quais comparações foram efetuadas.

      Resposta: O elemento 24, que resulta em 4 comparações.

4. Considere a lista: 90 13 7 24 85 32. Desenhe as trocas de elementos e determine o número de trocas efetuadas, utilizando:

   a. (1,0) Ordenação por seleção

   Resposta:

   $7 \leftrightarrow 90$: 7 13 90 24 85 32

   $13 \leftrightarrow 13$: 7 13 90 24 85 32

   $90 \leftrightarrow 24$: 7 13 24 90 85 32

   $90 \leftrightarrow 32$: 7 13 24 32 85 90

   $85 \leftrightarrow 85$: 7 13 24 32 85 90

   $90 \leftrightarrow 90$: 7 13 24 32 85 90

   Total: 6 trocas.

   b. (1,0) Ordenação por bolha

   Resposta:

	90^* 13 7 24 85 32
	90 13^* 7 24 85 32
$13 \leftrightarrow 90$:	13^* 90 7 24 85 32
	13 90 7^* 24 85 32
$7 \leftrightarrow 90$:	13 7^* 90 24 85 32
$7 \leftrightarrow 13$:	7^* 13 90 24 85 32
	7 13 90 24^* 85 32
$24 \leftrightarrow 90$:	7 13 24^* 90 85 32
	7 13 24 90 85^* 32
$85 \leftrightarrow 90$:	7 13 24 85^* 90 32
	7 13 24 85 90 32^*
$32 \leftrightarrow 90$:	7 13 24 85 32^* 90
$32 \leftrightarrow 85$:	7 13 24 32^* 85 90

   Total: 7 trocas.

5. (2,0) Seja L uma lista encadeada que armazena uma palavra, de forma que cada nó de L contém uma letra. Elabore um algoritmo que, utilizando uma pilha, verifica se a palavra em L é um palíndromo.

   Resposta: Seja a pilha P, inicialmente vazia ($topo = 0$), com tamanho suficiente para armazenar o total de letras de L.

Algoritmo:

$no := L$

enquanto $no \neq \lambda$ **faça**

$topo := topo + 1$

$P[topo] := no \uparrow .info$

$no := no \uparrow .prox$

$no := L$

enquanto $no \neq \lambda$ **faça**

se $P[topo] \neq no \uparrow .info$ **então**

retornar "Não é um palíndromo"

senão

topo:= topo -1

$no := no \uparrow .prox$

retornar "É um palíndromo"

6. (2,0) Sejam A e B duas listas encadeadas ordenadas, com m e n elementos, respectivamente. Elabore um algoritmo que imprima os elementos que pertencem a ambas as listas. Seu algoritmo deverá executar em $\Theta(n+m)$ passos.

   Resposta: Como o algoritmo a seguir percorre cada lista exatamente uma vez, o número de passos que ele executa é $\Theta(n+m)$.

   Algoritmo:

   $pt1 := A$

   $pt2 := B$

   enquanto $pt1 \neq \lambda$ e $pt2 \neq \lambda$ **faça**

   se $pt1 \uparrow .info < pt2 \uparrow .info$ **então**

   $pt1 := pt1 \uparrow .prox$

   senão **se** $pt1 \uparrow .info > pt2 \uparrow .info$ **então**

   $pt2 := pt2 \uparrow .prox$

   senão

   imprimir $pt1 \uparrow .info$

   $pt1 := pt1 \uparrow .prox$

   $pt2 := pt2 \uparrow .prox$