Estrutura de Dados – 2º período de 2019

Primeira Avaliação à Distância

1. **(1,0) Escreva as seguintes funções em notação: Θ**
   $n^2 + \log^2 n$; $2n^2 - 3$; $n^2 \log n$; $\log n + \sqrt{n}$; $n! + 2n$
   R: Podemos afirmar que as funções são, respectivamente:
   $\Theta(n^2)$, $\Theta(n^2)$, $\Theta(n^2 * \log n)$, $\Theta(\sqrt{n})$ e $\Theta(n!)$

2. **Para cada item abaixo, responda "certo" ou "errado", justificando em ambos os casos:**

   (a) **Se a complexidade de melhor caso de um algoritmo for $\Theta(f)$, então o número de passos que o algoritmo efetua no pior caso é $\Omega(f)$.**
   R: Certo. Como a complexidade de melhor caso de um algoritmo é Θ (f), então, o número de passos que o algoritmo efetua, qualquer que seja e entrada, será Ω(f).

   (b) **Se um limite inferior para um problema *P* é n^2, então todo algoritmo ótimo para *P* terá complexidade de pior caso $\Omega(n)$.**
   R: Correto. Se o limite inferior é n^2, então qualquer algoritmo, ótimo ou não, terá pior caso Ω (n^2). Logo, todo algoritmo também terá pior caso $\Omega(n)$.

   (c) **Se um algoritmo *A* que resolve um problema *P* tem o pior caso mais baixo assintoticamente dentre todos os algoritmos conhecidos que resolvem *P*, então *A* é ótimo.**

   R: Falso. Só podemos concluir que um algoritmo é ótimo quando sua complexidade de pior caso é igual ao limite inferior do problema. Neste exemplo, o fato do algoritmo ter o pior caso mais baixo dentre todos que resolvem P, não nos permite afirmar que este é o limite inferior de P.

3. **(0,5 cada) Considere a seguinte lista ordenada: 0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20. Utilizando busca binária, determine:**

   (a) **Um elemento cuja busca resulte em um número mínimo de comparações.**
   R: O elemento 10, pois resulta em apenas uma comparação

   (b) **Um elemento pertencente à lista cuja busca resulte em um número máximo de comparações. Determine quais comparações foram efetuadas.**
   R: O elemento 20, que resulta em 4 comparações

   (c) **Um elemento não pertencente à lista cuja busca resulte em um número máximo de comparações. Determine quais comparações foram efetuadas.**
   R: O elemento 25, que resulta em 4 comparações

   A tabela a seguir subsidia as respostas dos itens *a* e *b*. Além disso, também é possível observar que valores acima de 20 resultariam no máximo de comparações, subsidiando, portanto, o item *c*.

Cálculo	Resultado	Elemento
(01 + 11) / 2	6	10
(07 + 11) / 2	9	16
(10 + 11) / 2	10	18
(11 + 11) / 2	11	20

4. Considere a lista: 56 12 8 2 95 23 10. Desenhe todas as trocas de elementos e determine o número de trocas efetuadas, utilizando:

(a) **(0,8) Ordenação por seleção**
R: Segue abaixo a sequência de trocas obtidas a partir da ordenação por seleção.

56	12	8	2	95	23	10	Lista Original
2	12	8	56	95	23	10	Trocou 2 e 56
2	8	12	56	95	23	10	Trocou 8 e 12
2	8	10	56	85	23	12	Trocou 10 e 12
2	8	10	12	95	23	56	Trocou 12 e 56
2	8	10	12	23	95	56	Trocou 23 e 95
2	8	10	12	23	56	95	Trocou 56 e 95
2	8	10	12	23	56	95	Trocou 95 com ele mesmo. Lista Ordenada

(b) **(1,2) Ordenação por bolha**
R: Segue abaixo a sequência de trocas obtidas a partir da ordenação por bolha.

56	12	8	2	95	23	10	Lista Original
12	56	8	2	95	23	10	Troca 12 e 56
12	8	56	2	95	23	10	Troca 8 e 56
8	12	56	2	95	23	10	Troca 8 e 12
8	12	2	56	95	23	10	Troca 2 e 56
8	2	12	56	95	23	10	Troca 2 e 12
2	8	12	56	95	23	10	Troca 2 e 8
2	8	12	56	23	95	10	Troca 23 e 85
2	8	12	23	56	95	10	Troca 23 e 56
2	8	12	23	56	10	95	Troca 10 e 95
2	8	12	23	10	56	95	Troca 10 e 56
2	8	12	10	23	56	95	Troca 10 e 23
2	8	10	12	23	56	95	Troca 10 e 12. Lista Ordenada

5. (1,5) Sejam L_1 e L_2 duas listas ordenadas, simplesmente encadeadas com nó-cabeça. Escreva um algoritmo que construa uma 3a lista ordenada (sem alterar L_1 e L_2) contendo os elementos que pertencem a apenas uma das listas de entrada, mas não a ambas.

R: O algoritmo abaixo apresenta uma solução que atende ao solicitado nesta questão.

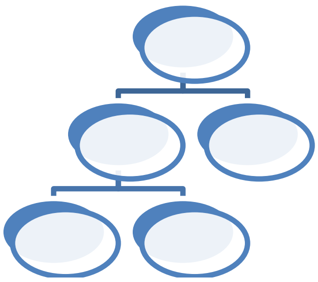
```
pont1 := ptlista1↑.prox      % ponteiro para a listaL1
pont2 := ptlista2↑.prox     % ponteiro para a listaL2

ocupar(ptnovo)                % a nova lista resultante iniciará com nó cabeça
ptnovo↑.prox = λ
ptaux := ptnovo

enquanto pont1 ≠λ e pont2 ≠ λ faça:
     se pont1↑.info < pont2↑.info então
          ocupar(pt)
          pt↑.info:=pont1↑.info
          pt↑.prox:=λ
          ptaux↑.prox:=pt
          ptaux:=pt          % ptaux aponta para o ultimo  nó
          pont1 :=pont1↑.prox

    senão
      se pont1↑.info > pont2↑.info então
          ocupar(pt)
          pt↑.info:=pont2↑.info
          pt↑.prox:=λ
          ptaux↑.prox:=pt
          ptaux:=pt
          pont2 :=pont2↑.prox
       senão
          pont1 :=pont1↑.prox
          pont2 :=pont2↑.prox

enquanto pont1 ≠ λ faça
          ocupar(pt)
          pt↑.info:=pont1↑.info
          pt↑.prox:=λ
          ptaux↑.prox:=pt
          ptaux:=pt
          pont1 :=pont1↑.prox

enquanto pont2 ≠ λ faça
          ocupar(pt)
          pt↑.info:=pont2↑.info
          pt↑.prox:=λ
          ptaux↑.prox:=pt
          ptaux:=pt
          pont2 :=pont2↑.prox
```

6. **(1,5) Seja *V* um vetor com *n* posições. Escreva um algoritmo que construa uma lista encadeada *L*, com nó cabeça, a partir de *V*, de forma que os elementos de *L* sejam os mesmos de *V* , de forma ordenada crescente. Por exemplo, se *V* contiver os elementos 1 9 3 5 7, nesta ordem, a lista *L* deverá conter os elementos 1 3 5 7 9, nesta ordem.**
   R: O algoritmo abaixo apresenta uma solução que atende ao solicitado nesta questão.

```
Para i := 1 até n faça:
   V_aux[i] =  V[i];

Ordena(V_aux, 1, n);   % Procedimento de ordenação (por exemplo, seleção) %

ocupar(PTLISTA);
Pont := PTLISTA;

Para i := 1 até n faça:
   ocupar(PT);
   PT↑.info = V_aux[i];
   PT↑.prox = λ;
   Pont↑.prox = PT;
   Pont := Pont.prox;
```

7. **(0,5 cada) Para cada item abaixo, desenhe uma árvore binária *T* que satisfaça os requisitos pedidos.**

   a. ***T* é uma árvore estritamente binária, com 3 níveis e número mínimo de nós.**
   R: Segue abaixo um exemplo de árvore que atende ao solicitado.



   b. ***T* é uma árvore completa, mas não cheia, com altura 4 e número máximo de nós.**
   R: Segue abaixo um exemplo de árvore que atende ao solicitado.