Curso de Tecnologia em Sistemas de Computação
Disciplina: Estrutura de Dados
Gabarito AP3 - 2019/1

1) ***(2,0) Escreva um algoritmo que efetue a seguinte tarefa: dada uma lista sequencial não ordenada com n elementos, encontrar os dois maiores elementos da lista.***

R: Uma solução é dada pelo algoritmo a seguir:

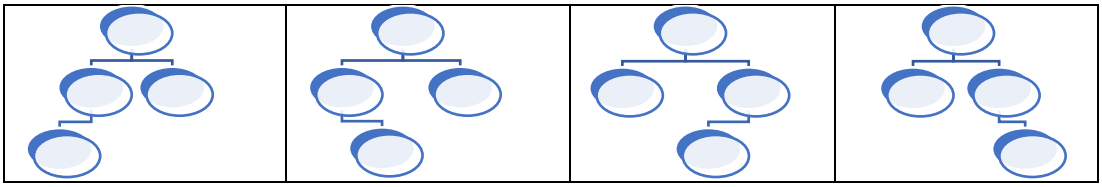
```
1   maior := L[1]
2
3   se maior > L[2]:
4       segMaior := L[2]
5   senão:
6       segMaior := maior
7       maior := L[2]
8
9   para i:= 3 até n faça:
10      se L[i] > maior:
11          segMaior := maior
12          maior := L[i]
13      senão:
14          se L[i] > segMaior:
15              segMaior := L[i]
16
17  imprimir ("Os dois maiores elementos são:", maior, "e", segMaior)
```

2) ***(2,0) Seja T uma árvore binária com n nós, representada por três vetores, E, D e R, onde para cada nó i, $1 \leq i \leq n$, E(i), D(i) e R(i) informam o índice do filho esquerdo de i, o índice do filho direito de i, e o rótulo de i, respectivamente. Além disso, a variável raiz contém o índice da raiz de T. Descreva (com palavras ou pseudo-código, como você preferir) um algoritmo que calcula o número de folhas de T. Você pode utilizar uma variável num-folhas para contar quantas folhas a árvore tem. (Suponha que esta variável está inicializada com zero.)***

R: Um nó *i* é folha quando não possui filhos: $E(i) = 0$ e $D(i) = 0$. Portanto, uma solução pode ser obtida pelo algoritmo abaixo que percorre os vetores *E* e *D*, somando a quantidade de nós que atendem a essa condição.

```
1   folhas := 0
2   para i := 1 até n faça
3       se E[i] = 0 e D[i] = 0 então
4           folhas := folhas + 1
5   imprimir(folhas)
```

3) ***(2,0) Desenhe todos os formatos de árvores AVL de altura 3 que contêm um número mínimo de nós.***
R: São apresentados abaixo os formatos de AVL possíveis considerando os requisitos em questão.

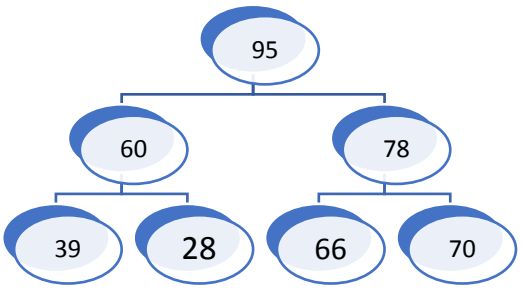
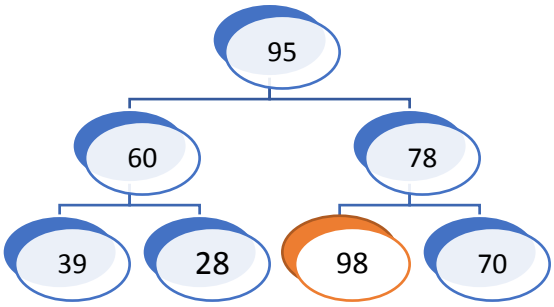
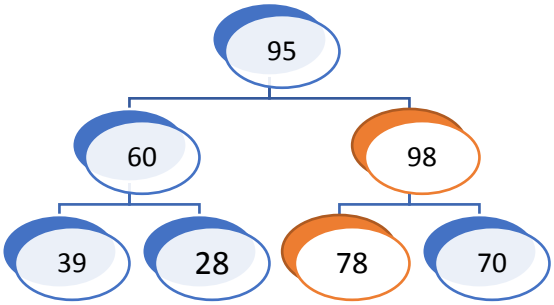
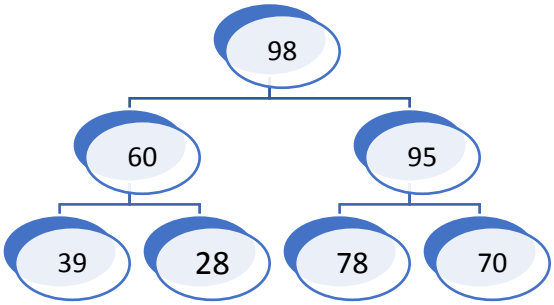


4) ***(2,0) Explique como funciona o procedimento de aumento de prioridade em um heap, fornecendo exemplo para um heap com 7 nós. Qual a complexidade do procedimento de aumento de prioridade?***
R: O aumento de prioridade consiste em trocar as posições dos nós filhos que forem maiores que seus pais. Formalmente temos:

> *Seja **V** o nó cuja prioridade foi aumentada. Caso a prioridade do pai de **V**, se existir, seja menor do que a do **V**, trocar de posições **V** e o pai de **V**. Iterativamente, repetir esta operação, tornando **V** igual a seu pai, até que o nó considerado seja a raiz da árvore, ou que sua prioridade seja menor ou igual que a prioridade do seu pai.*

Neste sentido, imagine que, dado o *heap* abaixo, desejamos aumentar a prioridade do nó 6, de 66 para 98. O custo de alteração das prioridades em um *heap* é, em pior caso, O (log n) e teríamos a seguinte sequência.

1	2	3	4	5	6	7
95	60	78	39	25	66	70

Heap original – Condição inicial 	Alteração da prioridade 66 para 68 
Troca entre 78 e 98 (desce 78) 	Troca entre 95 e 98 (desce 95) – Condição final 

5) ***(2,0) Determine uma árvore de Huffman para o seguinte conjunto de frequências: 2,2,4,4,4,8,8,16.***

R: Baseando-se nas frequências apresentadas no enunciado, podemos afirmar que temos 6 símbolos. Neste sentido, a seguinte árvore de Huffman pode ser construída.

Símb.	***Freq.***
S1	2
S2	2
S3	4
S4	4
S5	4
S6	8
S7	8
S8	16