Estruturas de Dados – Lista de exercícios

1. No instante $t = 0$, uma cultura de bactérias contém 8×10^6 indivíduos. No instante $t = i$ (sendo i um inteiro positivo que expressa o número de horas), o número de indivíduos na cultura é o dobro do número de indíviduos no instante anterior menos 7×10^3. Escreva dois algoritmos, um recursivo e outro não-recursivo, que calculem o número de indivíduos presentes na cultura no instante i. Calcule as complexidades dos algoritmos.

2. A *seqüência de Fibonacci* é uma sequência de elementos $f_1, f_2, \ldots, f_n$, definida do seguinte modo: $f_1 = 0, f_2 = 1, f_j = f_{j-1} + f_{j-2}$. Elaborar algoritmos, recursivo e não recursivo, para determinar o elemento f_n da sequência.

3. Escreva uma versão NÃO RECURSIVA do Algoritmo das Torres de Hanói que se encontra no livro-texto. Sugestão: utilize pilhas!

4. Seja $1, 2, \ldots, n$ uma seqüência de elementos que serão inseridos e posteriormente retirados de uma pilha P uma vez cada. A ordem de inclusão dos elementos na pilha é $1, 2, \ldots, n$, enquanto a de remoção depende das operações realizadas. Por exemplo, com $n = 3$, a seqüência de operações "incluir em P, incluir em P, retirar de P, incluir em P, retirar de P, retirar de P" produzirá a permutação $2, 3, 1$ a partir da entrada $1, 2, 3$. Representando por I, R, respectivamente, as operações de inserção e remoção da pilha, a permutação $2, 3, 1$ pode ser denotada por $IIRIRR$. De um modo geral, uma permutação é chamada *admissível* quando ela puder ser obtida mediante uma sucessão de inclusões e remoções em uma pilha a partir da permutação $1, 2, \ldots, n$. Assim, por exemplo, a permutação $2, 3, 1$ é admissível. Pede-se:

   (i) Determinar a permutação correspondente a $IIIRRIRR$, $n = 4$.

   (ii) Dê um exemplo de permutação não admissível.

5. Um lava-rápido tem capacidade máxima de atendimento de 5 carros (um que está sendo lavado, e quatro em espera). Cada lavagem leva 3 minutos. Ao chegar um novo cliente, o sistema ou o atende imediatamente (caso esteja completamente livre), ou o coloca em espera, ou o rejeita (caso já existam 5 carros sendo atendidos). Escreva um algoritmo que leia um inteiro $n \geq 1$ e um vetor binário (por exemplo, 001011100111000111), onde o i-ésimo bit da esquerda para a direita vale "1" se um novo cliente chega no i-ésimo minuto, e "0" se nenhum novo cliente chega no i-ésimo minuto. A seguir, o algoritmo deve calcular quantos carros foram lavados pelo sistema até o n-ésimo minuto. (Suponha que o vetor tem pelo menos n bits.) Use uma fila para representar o sistema a cada minuto que passa.

6. Assinale V ou F, justificando:

   - Se T é uma árvore estritamente binária com 10 nós então T possui 4 nós no terceiro nível.
   - Se T é uma árvore binária com 10 nós então $4 \leq h(T) \leq 10$, onde $h(T)$ é a altura de T.

7. Desenhe uma árvore binária de busca que seja **estritamente binária, de altura 4, e com o menor número possível de nós**. Não se esqueça de colocar os valores das chaves dentro de cada nó.

8. Desenvolva uma estratégia para remover um nó de uma árvore binária de busca. Note que a estrutura resultante após a remoção do nó deve continuar sendo uma árvore binária de busca!

9. Desenhe uma árvore binária de busca *cheia* com altura 4, colocando dentro de cada nó o valor de sua chave. As chaves são $1, 2, \ldots, k$ (k é o número de nós da árvore, que é um valor que você deve deduzir). A seguir, escreva a sequência de chaves que corresponde ao percurso em *pós-ordem* desta árvore.

10. Prove ou dê um contra-exemplo: Uma árvore binária pode ser construída, de forma única, a partir dos seus percursos em *pós-ordem* e *em nível*.

11. Considere o conjunto de chaves $S = \{1, 2, 3, 4, 5, 6, 7\}$. Pede-se responder as seguintes questões relativas a S:

    (a) Desenhar uma árvore binária de busca cheia, contendo os nós de S. Denote por T_1 esta árvore.

    (b) Desenhar uma árvore binária de busca zigue-zague, contendo os nós de S. Denote por T_2 esta árvore.

    (c) Escrever os nós de T_1 em pré-ordem, pós-ordem e ordem simétrica

    (d) Escrever os nós de T_2 em pré-ordem, pós-ordem e ordem simétrica.

12. Para cada sequência abaixo, responda se ela corresponde ou não a um heap (lista de prioridade). Justifique brevemente.

    (a) 33 32 27 31 29 26 25 30 28

    (b) 33 32 27 31 29 28 25 30 26

13. Desenhe e explique os passos intermediários do algoritmo de construção de um *heap* (lista de prioridades) que é executado em tempo $O(n)$, para o seguinte vetor de entrada: 34, 23, 89, 12, 67, 58, 45.

14. Desenhe e explique os passos intermediários do algoritmo de ordenação *Heapsort* ao seguinte vetor de entrada: 34, 23, 89, 12, 67, 58, 45.

15. Responda F ou V, justificando:

    (a) Seja T_1 um heap cuja raiz r tem prioridade igual a p. Seja T_2 o heap obtido de T_1 pela remoção de r, e seja T_3 o heap obtido de T_2 pela inserção de um nó com prioridade p em T_2. Então T_1 e T_3 são idênticos.

(b) O algoritmo de ordenação *Heapsort* é estável.

Observação: Um algoritmo de ordenação é dito *estável* quando a ordem relativa de dois elementos de mesmo valor se mantém após a ordenação. Isto é, se a lista L a ser ordenada contém dois elementos e_1 e e_2 tais que $e_1 = L[i]$, $e_2 = L[j]$, $e_1 = e_2$ e $i < j$, após a ordenação teremos e_1 e e_2 ocupando células $L[i']$ e $L[j']$ tais que $i' < j'$.

16. Dado um heap (lista de prioridades) T com n elementos, onde o elemento de prioridade máxima encontra-se na raiz (primeira posição de T), descrever o algoritmo para aumentar a prioridade de um dado elemento i de T. Suponha que a prioridade de um elemento j qualquer é denotada por $T[j].prio$

17. Seja o conjunto de chaves $S = \{1, 2, 3, 4\}$, que irá formar uma árvore binária de busca T. Desenhe todas as configurações possíveis que T pode assumir de modo que T seja também uma árvore AVL.

18. A partir de uma árvore inicialmente vazia, desenhe o passo a passo e a árvore AVL resultante da inserção de nós com chaves $20, 4, 25, 8, 7, 2, 10, 23$ (nesta ordem).

19. Desenhe uma árvore AVL de altura 4 que possua um número mínimo de nós (não se esqueça de colocar os valores das chaves dentro de cada nó).

20. Desenhe uma árvore B de ordem $d = 3$ e altura 3 contendo o menor número possível de chaves. (Os valores ficam à sua escolha.) A seguir, efetue a remoção de uma chave e desenhe a árvore B resultante da remoção.

21. Desenhe uma árvore B de ordem $d = 2$ com três níveis e o maior número possível de chaves. (Os valores das chaves ficam à sua escolha.). Responda (V ou F): Se uma árvore B tem o maior número possível de chaves, então ela tem o maior número possível de páginas.

22. Seja T uma árvore B de ordem 3 e altura 3. Dê exemplos de configurações que T poderia assumir, nos seguintes casos:

    (a) T tem número mínimo de chaves.

    (b) T tem número máximo de chaves.

    (c) T tem número mínimo de páginas, mas não de chaves.

    (d) T tem número máximo de páginas, mas não de chaves.

    Obs: Os valores das chaves ficam à escolha do aluno.

23. Seja T uma tabela de dispersão com 6 posições implementada por encadeamento exterior. A função de dispersão é $h(x) = x \bmod 6$. Desenhe a tabela após a inclusão das chaves 45,73,59,28,15,24,10,19, nesta ordem.

24. Suponha um conjunto S de 6 chaves, dispostos em uma tabela de dispersão T de tamanho 9, segundo uma função de dispersão h, onde o tratamento de colisões se realiza pelo método do encadeamento exterior. Determinar valores que as chaves devem possuir, bem como escolher a função de dispersão h e descrever a tabela T, em cada caso, para que T obedeça, respectivamente, às seguintes condições:

(a) Não existem colisões.

(b) Existe exatamente uma colisão.

(c) Existem exatamente duas colisões.

(d) Todas as inserções de chaves, a partir da segunda, geram colisões.

25. Esta questão refere-se ao grafo G_1 (veja a figura). Desenhe uma árvore de profundidade para G_1, indicando os valores $PE(v)$ e $PS(v)$. Desenhe as arestas de retorno, destacando-as. A raiz da busca é a. Os vizinhos de um vértice devem ser percorridos em ordem alfabética.

26. Esta questão refere-se ao digrafo G_2 (veja a figura). Desenhe uma árvore de profundidade para G_2, indicando os valores $PE(v)$ e $PS(v)$. Desenhe as demais arestas, classificando o tipo de cada uma delas. A raiz da busca é a. Os vizinhos de um vértice devem ser percorridos em ordem alfabética.

27. Esta questão refere-se ao grafo G_3 (veja a figura). Desenhe uma árvore de largura para G_3. Para cada vértice, indique a ordem em que ele entra na fila. Desenhe as demais arestas de G_3, classificando o tipo de cada uma delas. A raiz da busca é a. Os vizinhos de um vértice devem ser percorridos em ordem alfabética.

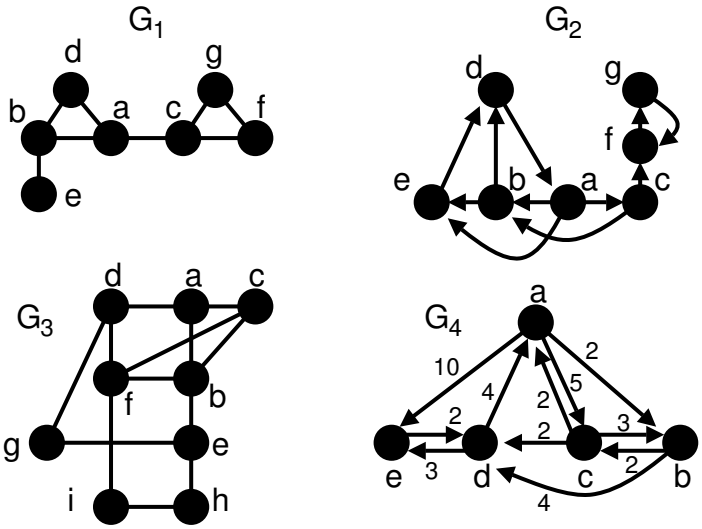
28. Num grafo conexo, a *distância* entre dois vértices é o número de arestas do menor caminho entre eles. Explique como a busca em largura pode ser usada para achar a distância entre dois vértices x e y.

29. Aplique o algoritmo de Dijkstra a G_4 (veja a figura) para encontrar caminhos mínimos do vértice a a todos os demais vértices. Faça uma tabela em que apareçam os rótulos provisórios $L(v)$ em cada iteração. Deixe claro qual o vértice escolhido em cada iteração.

    Desenhe a árvore geradora de caminhos mais curtos (com raiz a) de G_4, obtida implicitamente pela aplicação do algoritmo.

30. Verdadeiro ou falso? Em qualquer busca em profundidade de um grafo G, o número de arestas de retorno é sempre o mesmo. (Justifique brevemente.)

31. Verdadeiro ou falso? Todas as árvores de largura de um grafo G têm o mesmo número de níveis. (Justifique brevemente.)


G1
d
g
b
a
c
f
e
G2
d
g
f
e
b
a
c
G3
d
a
c
f
b
g
e
i
h
G4
a
10
2
4
2
5
2
3
2
3
2
e
d
c
b
4