MC202 — Estruturas de Dados

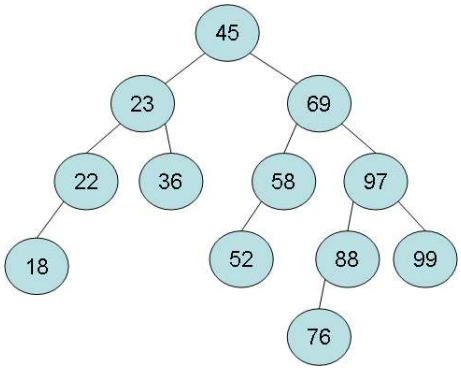
Professor

Ricardo da Silva Torres

<rtorres@ic.unicamp.br>

Quinta Lista de Exercícios

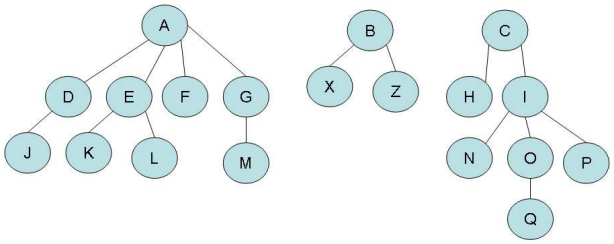
1. Dada uma árvore binária T, proponha um algoritmo que determine se T é uma árvore AVL.
2. Uma árvore AVL pode ter duas folhas cujas profundidades difiram de mais do que 2? Por exemplo, uma folha teria profundidade 10 e outra teria 7.
3. Desenhe a sequência de árvores AVL que se obtém depois da inserção das chaves a seguir (nesta ordem) em uma árvore inicialmente vazia: 170, 289, 110, 153, 128, 140, 303, 240, 264, 255.
4. Nesta questão, você deverá executar duas inserções e duas remoções de chaves da árvore AVL mostrada abaixo. Desenhe a árvore resultante da operação correspondente, seguindo estritamente os algoritmos explicados em classe.



   - (a) Insira 105.
   - (b) Insira 20.
   - (c) Remova 99.
   - (d) Remova 36.

5.
   - (a) Seja x o número formado pelos 2 últimos dígitos do seu RA. Construa uma árvore AVL (com chaves inteiras) de altura 3 de forma que a remoção do número $x + 20$ provoque uma rotação simples à direita. Mostre a árvore antes e depois desta remoção.
   - (b) Seja y o número formado pelos 2 dígitos centrais do seu RA. Construa uma árvore AVL (com chaves inteiras) de altura 3 de forma que a inserção do número $y + 20$ provoque uma rotação dupla. Mostre a árvore antes e depois desta inserção.

6. Considere a floresta (com três árvores) ilustrada abaixo.



- (a) Apresente a representação da floresta acima em forma de árvore binária.
- (b) Aplique os algoritmos de percursos em profundidade (inordem, pré-ordem e pós-ordem) para florestas
- (c) Aplique os algoritmos de percursos em profundidade para as árvores binárias obtidas em (a).
- (d) Há alguma similaridade entre os percursos em (b) e em (c)?

7. Qual o número mínimo e máximo de elementos que podem ser armazenados em um heap de profundidade d.
8. Construa um heap de máximo a partir de um vetor contendo os elementos 5, 3, 17, 10, 84, 19, 6, 22, 9.
9. Um professor do curso de Estrutura de Dados solicita aos alunos um trabalho de implementação. Ele deseja uma biblioteca para manipulação de heaps. Entre as funções pedidas está a função Remove_Heap(A, i,n), que remove do heap A o elemento na posição i do vetor. O algoritmo abaixo foi apresentado por um aluno como solução para esse problema. Infelizmente, esse algoritmo **não** está 100% correto. Você, como monitor da disciplina, precisa corrigir e explicar ao aluno qual foi o erro cometido. Portanto, terá que corrigir a função seguindo os seguintes passos:

```
A: vetor onde o heap está armazenado
n: tamanho do heap
i: índice do elemento a ser removido.
Remove_Heap( A, i, n) {
  Troca_Elemento(A, i, n);
  n = n-1;
  Desce_Heap(A, i, n);
}
```

- (a) Identifique o erro desse algoritmo, apresentando um caso de remoção em um heap para o qual ele não funcionaria.
- (b) Reescreva o algoritmo, corrigindo o problema encontrado no item (a). Qual a complexidade do algoritmo corrigido que você apresentou no item (b)?

10. Seja H um heap de máximo de tamanho n armazenado em um vetor. Em quais posições do vetor você pode encontrar:

- (a) a segunda menor chave?
- (b) a terceira menor chave?
- (c) a segunda maior chave?
- (d) a terceira maior chave?

11. Proponha uma implementação de fila de prioridade que não use heap. Apresente a estrutura de dados e os algoritmos para extração e inserção de elementos.