

## Estruturas de Dados II

### Exercícios de Fixação (ÁRVORE ABB E AVL)

**1:-)** Faça um algoritmo que retorne por referência ou por return o nível (ou profundidade) de um dado nó de uma Árvore Binária de Busca.

**2:-)** Faça um algoritmo que retorne o pai de um dado nó de uma Árvore Binária de Busca.

**3:-)** Faça algoritmos de percurso em Árvores Binárias (algoritmos de travessia) de forma iterativa.

- a) Pré-ordem
- b) In-ordem
- c) Pós-ordem

Utilize para isso uma Pilha com as funções: **Init(P)**, **Top(P)**, **Push(P,x)**, **Pop(P,x)** e **IsEmpty(P)**.

**4:-)** Faça um algoritmo para torar uma árvore Vazia (free em todos os nodos da árvore).

**5:-)** Uma Árvore Binária de Busca (ABB) pode ser implementada em disco (arquivo binário) ou em um vetor de registros, considerando que cada nó da árvore é um registro de um arquivo binário ou vetor, conforme desenho a seguir:

**Arquivo da árvore**

|   | esq | info | dir |
|---|-----|------|-----|
| 0 | 4   | 10   | 1   |
| 1 | 2   | 12   | 3   |
| 2 | 0   | 11   | 0   |
| 3 | 0   | 14   | 0   |
| 4 | 5   | 8    | 6   |
| 5 | 0   | 6    | 0   |
| 6 | 0   | 9    | 0   |

Faça algoritmos para a ABB em disco e também em vetor de registros:

- a) inserir na árvore ABB.
- b) percorrer a árvore ABB em nível ou profundidade.
- c) percorrer a árvore ABB em pré-ordem, in-ordem, pós-ordem de forma recursiva e também iterativa.
- d) fazer o algoritmo da exclusão da árvore ABB.
- e) fazer o algoritmo de balanceamento da árvore ABB.
- f) excluir todos os nodos da árvore ABB.

6:-) Um vetor numérico pode ser utilizado para representar uma árvore ABB. Para isso você precisa percorrer a árvore utilizando as contas:

Ramificação esquerda:  $\text{pai} * 2 + 1$

Ramificação direita:  $\text{pai} * 2 + 2$

Faça algoritmos para:

- a) inserir na árvore ABB.
- b) percorrer a árvore ABB em nível ou profundidade.
- c) percorrer a árvore ABB em pré-ordem, in-ordem, pós-ordem de forma recursiva e também iterativa.

7:-) Faça um algoritmo para inserir um elemento em uma Árvore AVL de forma recursiva e iterativa. Os algoritmos de rotação à esquerda e rotação à direita são os mesmos do algoritmo recursivo. Você deverá utilizar uma Pilha para simular a recursividade com as funções: **Init(P)**, **Top(P)**, **Push(P,x)**, **Pop(P,x)** e **IsEmpty(P)**.

8:-) Faça um algoritmo para remover um elemento de uma Árvore AVL de forma recursiva e iterativa. Os algoritmos de rotação à esquerda e rotação à direita são os mesmos do algoritmo recursivo. Você deverá utilizar uma Pilha para simular a recursividade com as funções: **Init(P)**, **Top(P)**, **Push(P,x)**, **Pop(P,x)** e **IsEmpty(P)**.