

Professores: Renata de Matos Galante

Exercícios Árvores AVL

01 – Insira em uma árvore AVL, itens com as chaves apresentadas nos itens a seguir (na ordem em que aparecem). Desenhe a árvore resultante da inserção, sendo que uma nova árvore deve ser desenhada quando houver uma rotação. Indique qual a rotação que foi executada.

- a) 30, 40, 24, 58, 48, 26, 11, 13, 14
- b) 20, 15, 25, 10, 30, 24, 17, 12, 5, 3
- c) 40, 30, 50, 45, 55, 52
- d) 20, 15, 25, 12, 17, 24, 30, 10, 14, 13
- e) 20, 15, 25, 12, 17, 30, 26

02 – Quantas árvores binárias de pesquisa (**ABP**) diferentes podem armazenar as chaves {1,2, 3, 4}?

03 – Um certo professor Amongus afirma que a ordem pela qual um conjunto fixo de elementos é inserido em uma árvore AVL não interessa – sempre resulta na mesma árvore. Apresente um pequeno exemplo que prove que ele está errado.

04 – Analise uma árvore **T** que armazena 100.000 itens. Quais são o pior e o melhor casos em relação à altura de **T** das seguintes árvores:

- **T** é uma árvore binária de pesquisa (ABP);
- **T** é uma árvore AVL.

05 – Para qual ordem de inserção dos elementos o caminhamento pré-fixado à esquerda se iguala ao caminhamento central à esquerda?

06 – Analise os trechos de código apresentados a seguir, identificando a rotação que eles realizam e corrigindo-os quando for o caso.

```
Type
  PNode = ^node;
  node = record
    info: Telem;
    fator :integer
    esq, dir: PNode;
  End;

Type tree: PNode;
```

```
procedure rotacaoA(var pt: PNode);
  var ptu: PNode;
begin
  ptu:= pt^.esq;
  pt^.esq:= ptu^.dir;
  ptu^.dir:= pt;
  pt^.fator:= 0;
  pt:= ptu;
end;
```

```
procedure rotacaoB(var pt: PNode);
  var ptu: PNode;
begin
  ptu:= pt^.dir;
  pt^.dir:= ptu^.esq;
  ptu^.esq:= pt;
  pt^.fator:= 0;
  pt:= ptu;
end;
```

<pre> procedure rotacaoC (var pt: PNode); var ptu, ptv: PNode ; begin ptu:= pt^.dir; ptv:= ptu^.esq; ptu^.esq:= ptv^.dir; ptv^.dir:= ptu; pt^.dir:= ptv^.esq; ptv^.esq:= pt; if ptv^.fator = 1 then pt^.fator:= -1 else pt^.fator:= 0; if ptv^.fator = -1 then ptu^.fator:= 1; else ptu^.fator:= 0; pt:= ptv; end; </pre>	<pre> procedure rotacaoD (var pt: PNode); var ptu, ptv: PNode; begin ptu:= pt^.esq; ptv:= ptu^.dir; ptu^.dir:= ptv^.esq; ptv^.esq:= ptu; pt^.esq:= ptv^.dir; ptv^.dir:= pt; if ptv^.fator = -1 then pt^.fator:= 1 else pt^.fator:= 0; if ptv^.fator = 1 then ptu^.fator:= -1; else ptu^.fator:= 0; pt:= ptv; end; </pre>
---	--

3 – A seguir, é apresentada o procedimento que faz o balanceamento e atualização dos fatores para os dois casos de inserção (à direita e à esquerda) apresentados em aula.

<pre> Proc InsereAVL (var a:Arvore; x:Info; ok:lógico); { Insere nodo em uma árvore AVL, onde A representa a raiz da árvore, x, a chave a ser inserida e h a altura da árvore } se a = nil então início InicioNo(a,x); {aloca o novo nodo e insere a informação} ok:=verdadeiro fim; senão se x = a^.Info então pare {se o nodo já existe então não será inserido novamente} se x < a^.chave então início {o nodo será inserido à esquerda} InsereAVL(a^.esq,x,ok); se ok então caso a^.bal seja -1: a^.bal := 0; ok := falso; 0: a^.bal := 1; 1: Caso1(a,ok); fim; {faz o balanceamento} senão início {o nodo será inserido à direita} InsereAVL(a^.dir,x,ok); se ok então caso a^.bal seja 1: a^.bal := 0; ok := falso; 0: a^.bal := -1 -1: Caso2(a,ok); fim; {faz o balanceamento} fim; </pre>
--

Para os casos 01 e 02 identifique nos pontos de interrogação quais são as rotações que devem ser feitas. Justifique sua resposta.

<pre> Proc Caso1 (var a:Arvore, ok:lógico); var ptu: Arvore; início ptu = a^.esq; se ptu^.bal = 1 então rotacao_???; senão rotacao_???; a^.bal := 0; ok := falso; fim; </pre>	<pre> Proc Caso2 (var a:Arvore, ok:lógico); var ptu: Arvore; início ptu = a^.dir; se ptu^.bal = -1 então rotacao_???; senão rotacao_???; a^.bal := 0; ok := falso; fim; </pre>
--	---