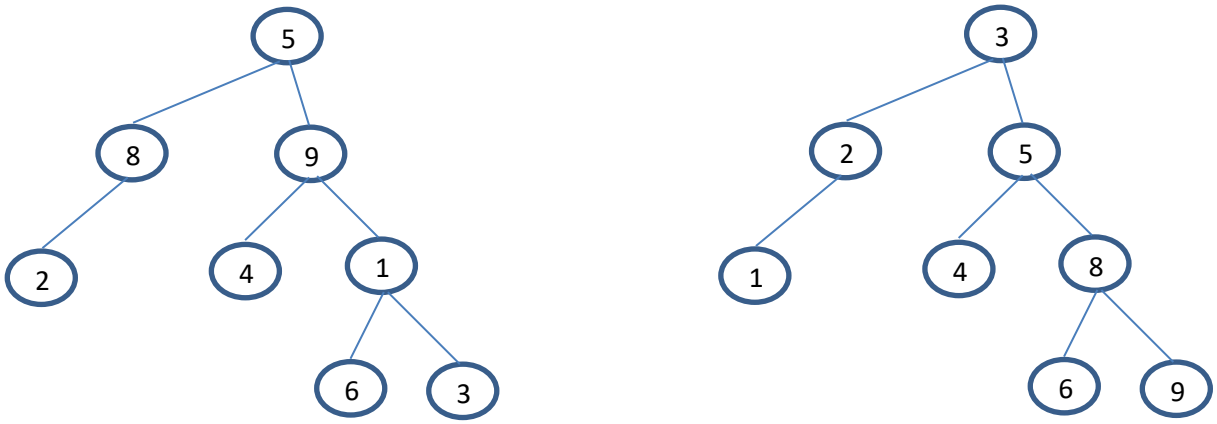
UNIVERSIDADE PRESBITERIANA MACKENZIE
- Faculdade de Computação e Informática –

Disciplina: Estrutura de dados II
Aula 2 – Profa. Valéria Farinazzo

Tipo Abstrato de Dados (TAD) - Árvore:

Árvores Binária de Busca

Um tipo especial de árvore binária, chama-se **árvore de busca binária**. Uma árvore de busca binária é uma árvore binária com a seguinte propriedade: todos os itens na sub árvore da esquerda de um nó possuem valores menores ou iguais ao valor do item do nó e todos os itens na sub árvore da direita de um nó possuem valores maiores ao valor do item do nó. A figura abaixo apresenta uma árvore de binária (a esquerda) e uma árvore de busca binária (a direita).



Exercícios:

1. *Percorra as duas árvores em pré-ordem, em-ordem e pós-ordem.*

2. *Que propriedades você verifica quando percorre a arvore de busca binária para o caso de em-ordem da questão anterior?*

3. *Crie um algoritmo que encontre o menor elemento de uma árvore binária de busca.*

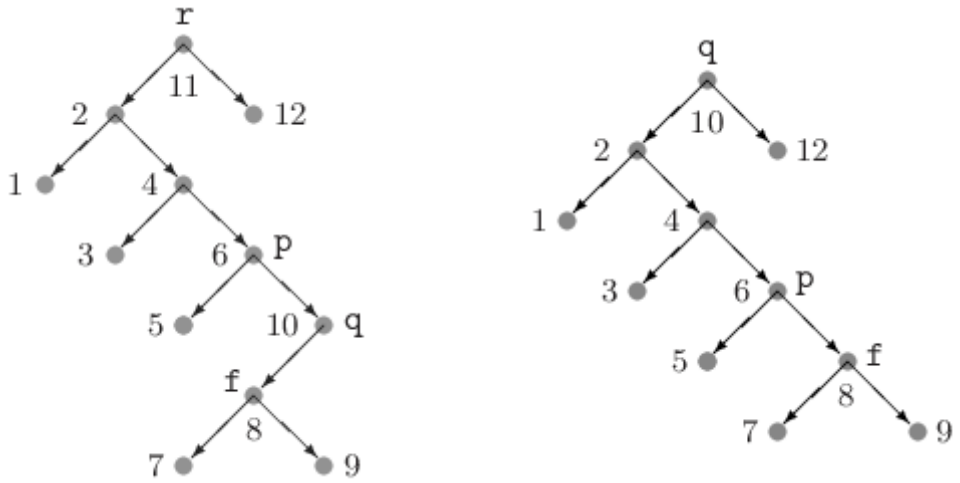
4. *Crie um algoritmo que encontre a altura de uma árvore binária.*

5. *Crie um algoritmo que busque determinado elemento na árvore binária de busca.*

6. *Suponha que as chaves 50 30 70 20 40 60 80 15 25 35 45 36 são inseridas, nesta ordem, numa árvore de busca inicialmente vazia.*

REMOÇÃO

Problema: Remover um nó de uma árvore de busca de tal forma que a árvore continue sendo de busca. Comecemos tratando do caso em que o nó a ser removido é a raiz da árvore. Se a raiz não tem um dos filhos, basta que o outro filho assuma o papel de raiz. Senão, faça com que o nó anterior à raiz na ordem e-r-d assuma o papel de raiz.



A figura ilustra o antes-e-depois da remoção do nó r. O nó anterior a r na ordem e-r-d é q (os nós estão numerados em ordem e-r-d). O nó q é colocado no lugar de r, os filhos de r passam a ser filhos de q, e f passa a ser filho (direito) de p.

```
No * removeraiz (No *r) {
         No *p, *q;
         if (r->getEsq() == NULL) {
                     q = r->getDir();
                     free (r);
                     return q;
        }
        p = r; q = r->getEsq();
        while (q->getDir() != NULL) {
              p = q; q = q->getDir();
        }
    // q é nó anterior a r na ordem e-r-d
    // p é pai de q
       if (p != r) {
                     p->setDir(q->getEsq());
                     q->setEsq(r->getEsq());
       }
       q->setDir(r->getDir());
       free (r);
       return q;
}
```

Suponha agora que queremos remover um nó que não é a raiz da árvore. Para remover o filho esquerdo de um nó x faça

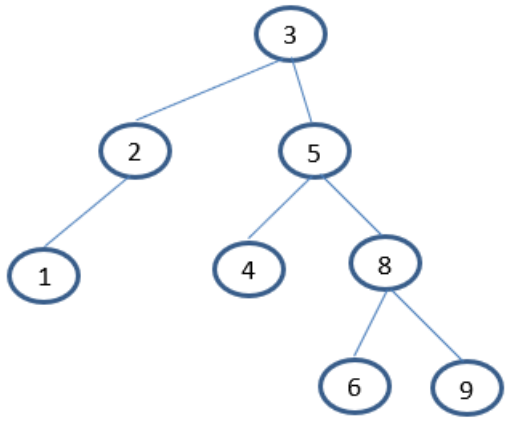
```
x->esq = removeraiz (x->esq);
```

e para remover o filho direito de x faça

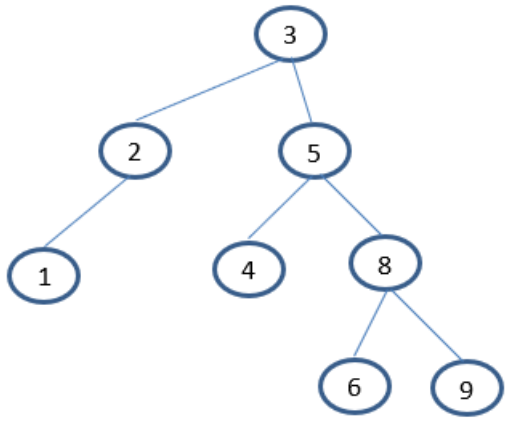
```
x->dir = removeraiz (x->dir);
```

Exercícios (cont.)

7. *Dada a árvore abaixo, faça a remoção do elemento 2.*



8. *Dada a árvore abaixo, faça a remoção do elemento 5.*



9. *Utilizando a árvore criada no exercício 6, remova o nó que contém a informação 30.*

10. *Faça um método que remova qualquer informação presente na árvore.*

11. *Em algumas aplicações, é interessante armazenar os dados do nó-pai. O que seria modificado na classe nó? Na classe arvore Binária, quais e como seriam os métodos afetados?*