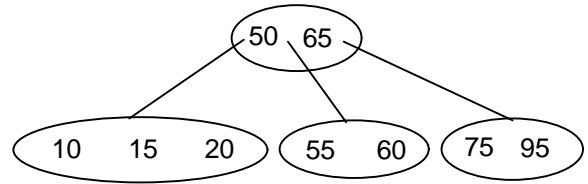
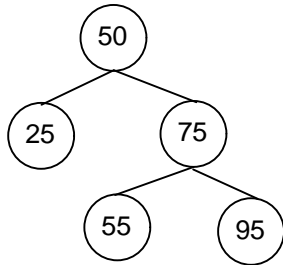


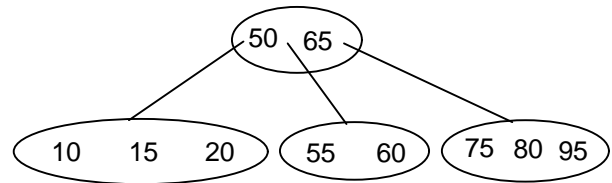
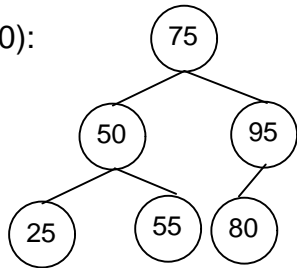
INE 5384 - Estruturas de Dados – Turma: 0332B – Semestre: 2002/2
Gabarito da Prova 2

Questão 1:

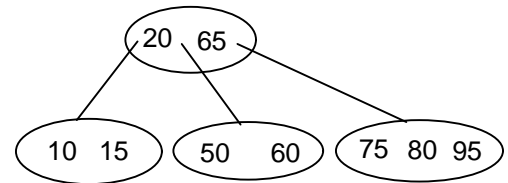
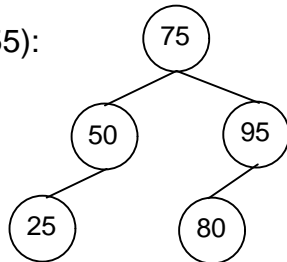
Inclui(95):



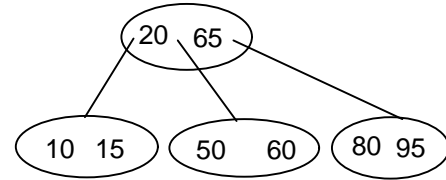
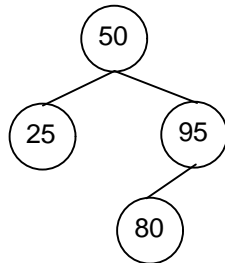
Inclui(80):



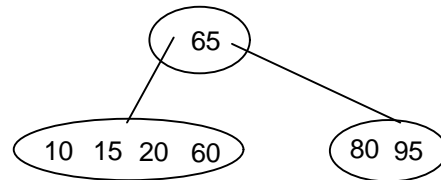
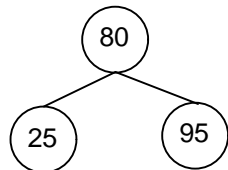
Exclui(55):



Exclui(75):



Exclui(50):



Questão 2:

a)

Método inclui(filho Object, pai Object) retorna booleano;

Início

aux ← Árvore;

resposta booleano;

se *this* = NULL então retorna FALSO;

se dado <> pai e filhos = NULL e irmãos = NULL então retorna FALSO;

se dado = pai então

início

se filhos <> NULL então

início

aux ← filhos;

enquanto aux.irmãos <> NULL faça

aux ← aux.irmãos;

aux.irmãos ← NOVO Árvore(filho);

fim

senão filhos ← NOVO Árvore(filho);

resposta ← VERDADEIRO;

fim

senão início

resposta ← irmãos.inclui(filho, pai);

se resposta = FALSO então

resposta ← filhos.inclui(filho, pai);

fim;

retorna resposta;

Fim;

b)

Método altura() retorna inteiro;

Início

resposta, maiorAltura inteiro;

aux ← Árvore;

se *this* = NULL então retorna -1;

maiorAltura ← 0;

aux ← *this*;

enquanto aux <> NULL faça

início

resposta ← aux.filhos.altura() + 1;

se resposta > maiorAltura então maiorAltura ← resposta;

aux ← aux.irmãos;

fim;

retorna maiorAltura;

Fim;

c)

Método exclui(d Object) retorna booleano;

Início

ant, ptr Árvore;

se dado = d então

início

dado ← NULL;

filhos ← NULL;

irmãos ← NULL;

fim

senão se filhos = NULL retorna FALSO

senão se filhos.dado = d então filhos ← filhos.irmãos

senão início

ant ← filhos;

ptr ← filhos.irmãos;

enquanto ptr <> NULL faça

início

se ptr.dado = d então

início

ant.irmãos ← ptr.irmãos;

retorna VERDADEIRO;

fim

senão início

ant ← ptr;

ptr ← ptr.irmãos;

fim;

fim;

ptr ← filhos;

enquanto ptr <> NULL faça

início

resposta ← ptr.exclui(d);

se resposta = FALSO então ptr ← ptr.irmãos

senão retorna resposta;

fim;

retorna FALSO;

fim;

retorna VERDADEIRO;

Fim;

Questão 3:

Método UnificaTabelasHash(t1 TabelaHash, t2 TabelaHash) retorna TabelaHash;

Início

```
    novoM, i inteiro;
    ptr Elemento;
    resultado TabelaHash;

    se t1.nroChaves = 0 e t2.nroChaves = 0 então retorna NULL;
    se t1.nroChaves = 0 então retorna t2;
    se t2.nroChaves = 0 então retorna t1;
    se t1.obtémM() > t2.obtémM() então novoM ← t1.obtémM()
    senão novoM ← t2.obtémM();
    resultado ← NOVO TabelaHash(novoM);
    para i de 0 até t1.obtémM() – 1 faça
        início
            ptr ← t1[i].obtemPrimeiroElemento();
            enquanto ptr <> NULL faça
                início
                    resultado[i].inclui(ptr.obtemChave(), ptr.obtemDado());
                    ptr ← ptr.obtemProx();
                fim;
            fim;
        fim;
    para i de 0 até t2.obtémM() – 1 faça
        início
            ptr ← t2[i].obtemPrimeiroElemento();
            enquanto ptr <> NULL faça
                início
                    resultado[i].inclui(ptr.obtemChave(), ptr.obtemDado());
                    ptr ← ptr.obtemProx();
                fim;
            fim;
        fim;
    retorna resultado;
```

Fim;