

MC202: Estruturas de Dados

Professores Cid C. de Souza e Hélio Pedrini

Instituto de Computação - UNICAMP – 2º semestre de 2009

Turmas A, B, C e D – 1ª Prova (15/10/2009)

Nome:

RA: Turma:

Questão	Valor	Nota
1	2,0	
2	2,0	
3	2,0	
4	2,0	
5	2,0	
Total	10,0	

Instruções: A duração da prova é de 110 minutos. **Não é permitida consulta** a qualquer material. *Somente serão consideradas respostas nos espaços indicados.* Use os versos das folhas como rascunho. Nas questões que solicitam que seja completada uma função cujo esboço já é fornecido, cada retângulo deverá conter **uma única expressão**, ou então, **um único comando simples** em linguagem C, conforme o contexto.

```
typedef struct Regaux{
    int info;
    struct Regaux *prox;
} RegFila, *ApFila;

typedef ApFila Fila;

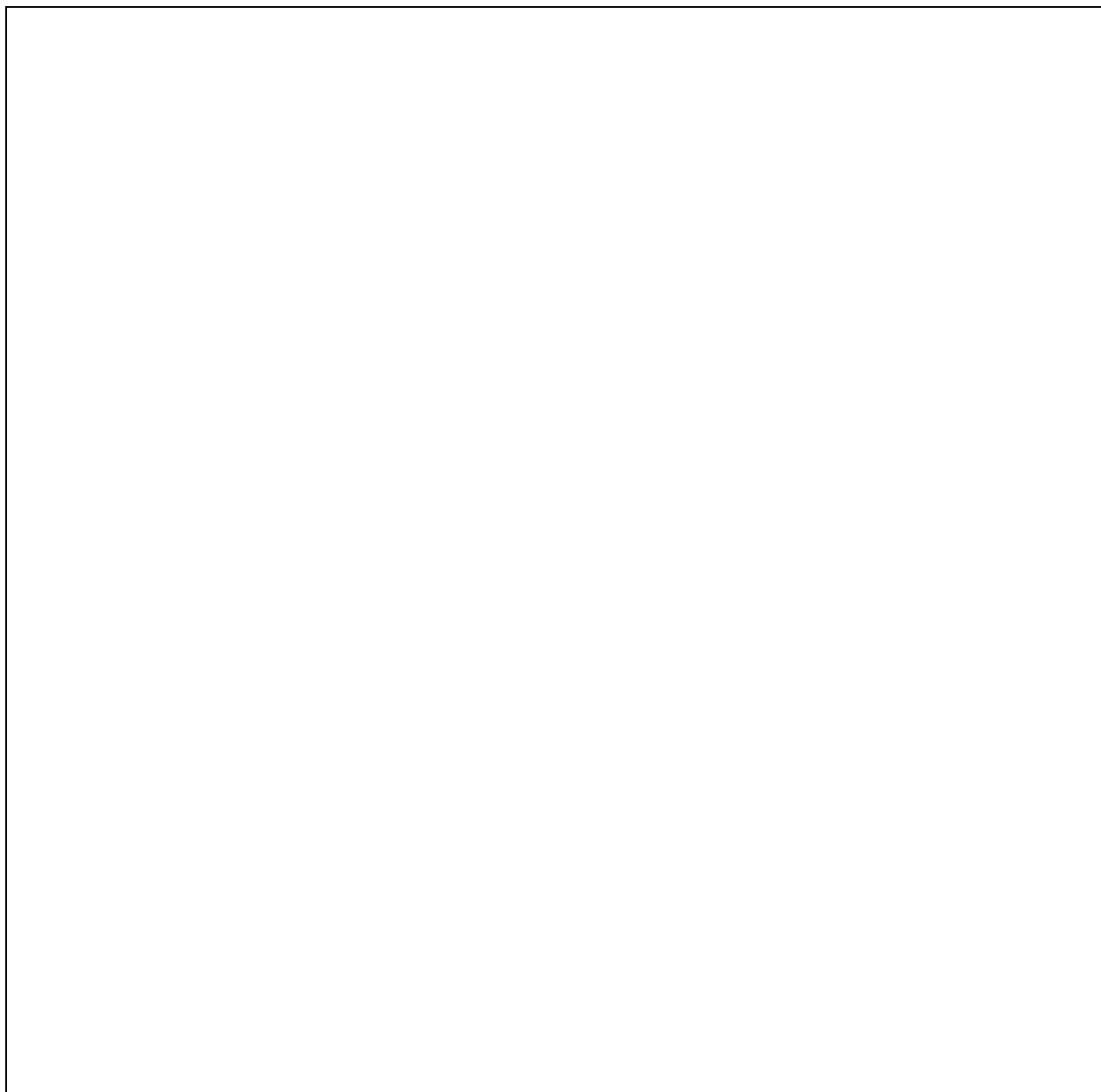
int main(){
    Fila f; int somaRec=0,somaIter=0;
    InicializaFila(&f); LeFila(&f);
    ...
    SomaImparIter(f,&somaRec); /* Recursivo */
    SomaImparRec(f,&somaIter); /* Iterativo */
    ...
    return 0; }
```

[illegible]

2. (a) Reconstrua (desenhe) a árvore binária a partir dos seus percursos em pós-ordem e in-ordem.

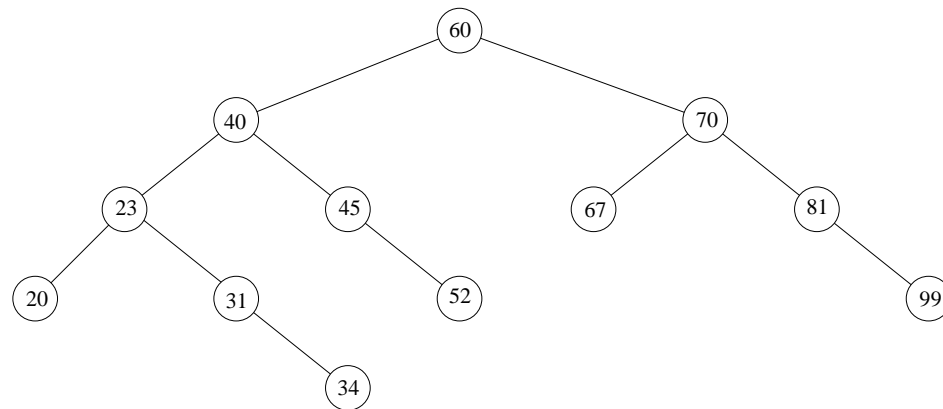
Pós-ordem: C K L Z A F D W E N Y B M

In-ordem: C A K L Z M F Y D E W N B



(b) Indique a pré-ordem para a árvore obtida acima:

3. Nesta questão você deverá executar duas inserções e duas remoções de chaves da árvore AVL mostrada abaixo. Desenhe em cada um dos retângulos em branco a árvore resultante da operação correspondente, seguindo **estritamente** os algoritmos explicados em aula. **Essas operações devem ser executadas sempre na árvore original.**



insira 18

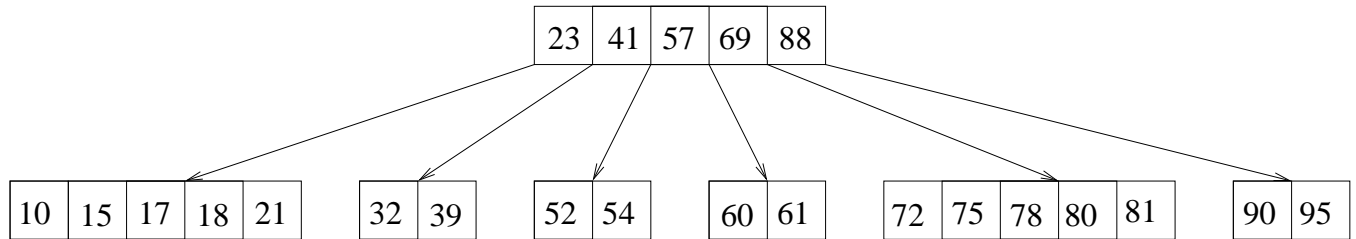
insira 87

remova 20

remova 67

4. Usando **exclusivamente** as operações vistas em sala de aula para árvores B , responda os itens abaixo. Suponha que todas as árvores do enunciado têm grau mínimo 3 ($t = 3$).

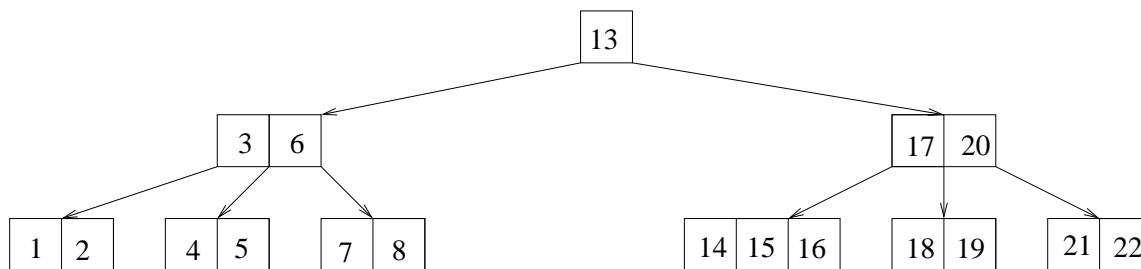
4.1 Considere a árvore B esquematizada abaixo.



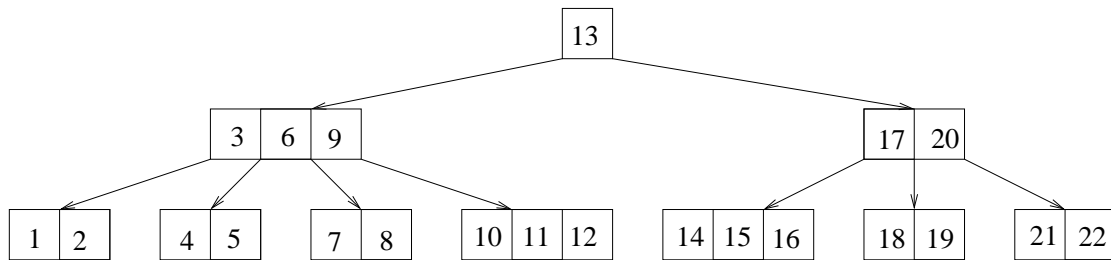
Indique no quadro a seguir a quantidade de operações de *split* que serão realizadas ao se **inserir** na árvore **original** a chave de valor: (a) 19; (b) 56 e (c) 85.

Pelo menos uma das chaves acima requer mais de um *split* para ser inserida. Escolha uma destas chaves, insira na árvore **original** e mostre no quadro abaixo a árvore resultante desta operação. Não deixe de indicar claramente qual foi a chave que você inseriu.

4.2 Considere a árvore B esquematizada abaixo. Desenhe no quadro apropriado a árvore B obtida desta árvore ao ser removida a chave 6.



4.3 Considere a árvore B esquematizada abaixo. Desenhe no quadro apropriado a árvore B obtida da árvore original ao ser removida a chave indicada em cada um dos três casos abaixo.



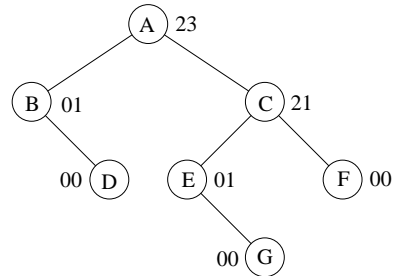
remove 9

remove 6

remove 17

5. Um caminho **zigzag direito** (esquerdo) a partir de um nó t de uma árvore binária é uma seqüência de nós $(t = t_0, t_1, t_2, \dots, t_q)$ onde t_{i+1} é filho direito (esquerdo) de t_i para todo i par e filho esquerdo (direito) de t_i para todo i ímpar. O **comprimento do caminho zigzag** é igual ao tamanho da seqüência menos uma unidade, ou seja, q (a seqüência começa de zero).

A **altura zigzag direita** (esquerda) de um nó t de uma árvore binária é o comprimento do maior *caminho zigzag direito* (esquerdo) que começa em t . A **altura zigzag direita** (esquerda) de uma árvore binária é dada pela *altura zigzag direita* (esquerda) de sua raiz. Convencionou-se que as alturas *zigzag* direita e esquerda de uma árvore *vazia* são ambas iguais a -1 . Na figura abaixo vê-se uma árvore com as *alturas zigzag esquerda e direita* indicadas (na ordem) ao lado dos respectivos nós.



Complete a rotina **recursiva** ZIGZAG abaixo que calcula as *alturas zigzag esquerda e direita* (segundo e terceiro parâmetros de entrada, respectivamente) de uma árvore binária com raiz em um nó apontado pela variável t (primeiro parâmetro da entrada). Os registros da árvore são definidos por:

```
typedef struct RegAux{
    int info;
    struct RegAux *esq, *dir;
} Reg, *ApReg;
```

```
typedef ApReg ArvoreBin;
```

```
void ZIGZAG(ArvoreBin t, int *ze, int *zd) {
    int zee, zed, zde, zdd;
    if (  ){
         ;
         ;
    } else {
         ;
         ;
         ;
         ;
    }
}
```