Lista de Exercícios de Árvores e Ordenação de Dados
Árvores B+

1) (a) Mostre, passo a passo, o resultado de inserir as chaves 10, 20, 50, 80, 30, 40, 70, 60, 90, 200, 150, 75, 65, 62 em uma árvore B+, de ordem M = 5, inicialmente vazia. OBS: Opere sempre que o irmão adjacente da esquerda e indique sempre que operações estão sendo realizadas.

(b) Mostre, passo a passo, o resultado de remover as chaves 75, 30 da árvore B+ resultante da execução do item anterior. OBS: Opere sempre que o irmão adjacente da esquerda e indique sempre que operações estão sendo realizadas.

2) (a) Mostre, passo a passo, o resultado de inserir as chaves 1, 2, 5, 9, 3, 4, 7, 6, 10, 19, 17, 12, 18, 8, 30 em uma árvore B+, de ordem M = 5, inicialmente vazia. OBS: Opere sempre que o irmão adjacente da esquerda e indique sempre que operações estão sendo realizadas.

(b) Mostre, passo a passo, o resultado de remover as chaves 10, 4 e 6 da árvore B+ resultante da execução do item anterior. OBS: Opere sempre que o irmão adjacente da esquerda e indique sempre que operações estão sendo realizadas.

3) (a) Mostre, passo a passo, o resultado de inserir as chaves 110, 100, 90, 80, 60, 50, 40, 130, 120, 70, 150, 30, 200, 280, 500, 170,105,75, 190,145 em uma árvore B+ de ordem M = 5, inicialmente vazia. OBS: Opere sempre que o irmão adjacente da esquerda e indique sempre que operações estão sendo realizadas.

(b) Mostre, passo a passo, o resultado de remover as chaves 30, 40, 60, 90, 105, 130, 50, 80, 75 da árvore B+ resultante da execução do item anterior. OBS: Opere sempre que o irmão adjacente da esquerda e indique sempre que operações estão sendo realizadas.

4) Considere a definição de nó de árvore B+ e de árvore B+ abaixo.

```
class BNodePlus {
   private int n;
   private boolean ehFolha; // true é folha, false é nó intermediário
   private String [] chaves;  // É criado somente se for um nó intermediário
   private Aluno [] infos;     // É criado apenas se for um nó folha
   private BNodePlus [] pont;
   private BNodePlus prox;
   …
}

public class BtreePlus {
       private BNodePlus root;
        ....
}
```

(a) Implemente um método (**função**) privado na classe BTreePlus para, dado um valor de chave, encontrar a informação correspondente. Caso encontre, a função deve retornar o par (**nó**, **posição**), onde **nó** é a referência para o nó onde o valor foi encontrado e **posição** é a localização da informação dentro do nó. Caso não encontre, a função retorna o par (null, -1).

(b) Implemente um método (**função**) privado na classe BTreePlus para encontrar a chave de menor valor. A função deve retornar a referência para o nó que contém o menor valor.

(c) Implemente um método (**função**) privado na classe BTreePlus para encontrar a chave de maior valor. A função deve retornar a referência para o nó que contém o maior valor.

(d) Implemente um método (**procedimento**) para exibir todos os registros armazenados em uma árvore B+ por ordem crescente de chave.