Universidade Federal do Ceará
Campus Crateús
Professor: Bruno C. H. Silva

Nota:
Disciplina: Estruturas de Dados
Turma: _____

Aluno: _____

Trabalho

Duração: 72 horas Data: / /

Este trabalho contém 2 página(s), incluindo esta capa, e 1 questões, formando um total de 10 pontos.

Tabela (para uso EXCLUSIVO do professor)

Questão:	1	Total
Valor:	10	10
Pontuação:		

1. (10 pontos) Uma Árvore Binária de Busca (ABB) possuí as seguintes propriedades: seja $S = \{s_1, ..., s_n\}$ o conjunto de valores dos nós v_i de uma árvore T, esse conjunto satisfaz $\{s_1 < ... < s_n\}$, e, a cada nó $v_i \in T$ está associado um valor distinto $s_i \in S$. Seja v um nó de T, se $v_i \in T$ está a esquerda de v, então $v_i.s_i < v.s$. Se $v_i \in T$ está a direita de v, então $v_i.s_i > v.s$. Em outras palavras, os nós pertencentes à sub-árvore esquerda possuem valores inferiores do que o valor associado ao nó-raiz r. Já os nós nós pertencentes à sub-árvore direita possuem valores maiores do que o valor associado ao nó-raiz r. Você deve então implementar um TAD de Árvore Binária de Busca e testá-lo em um função *main*. A lógica das funções *create*, *clear*, *printAllPos*, *printAllPre*, *printAllIn* para uma Árvore Binária de Busca é a mesma da Árvore Binária implementada no último trabalho. Porém a lógica da inserção (*add*) e pesquisa (*find*) é diferente. O projeto de implementação destas duas funções é explicado como se segue:

   1.) Lógica da função de inserção:
   - Procure um local para inserir o novo nó, começando a procura a partir do nó-raiz;
   - Para cada nó-raiz de uma sub-árvore, compare: se o novo nó possui um valor menor do que o valor no nó-raiz (vai para sub-árvore esquerda), ou se o valor é maior que o valor no nó-raiz (vai para sub-árvore direita);
   - Se um ponteiro (filho esquerdo/direito de um nó-raiz) nulo é atingido, coloque o novo nó como sendo filho do nó-raiz.

   2.) Lógica da função de pesquisa:
   - Comece a busca a partir do nó-raiz;

- Para cada nó-raiz de uma sub-árvore compare: se o valor procurado é menor que o valor no nó-raiz (continua pela sub-árvore esquerda), ou se o valor é maior que o valor no nó-raiz (sub-árvore direita);
- Caso o nó contendo a chave pesquisada seja encontrado, retorne o nó pesquisado, caso contrário retorne *NULL*.

Para entender o algoritmo de inserção considere a seguinte sequência de inserções $\{17, 99, 13, 1, 3, 100, 400\}$. Teríamos a seguinte composição:

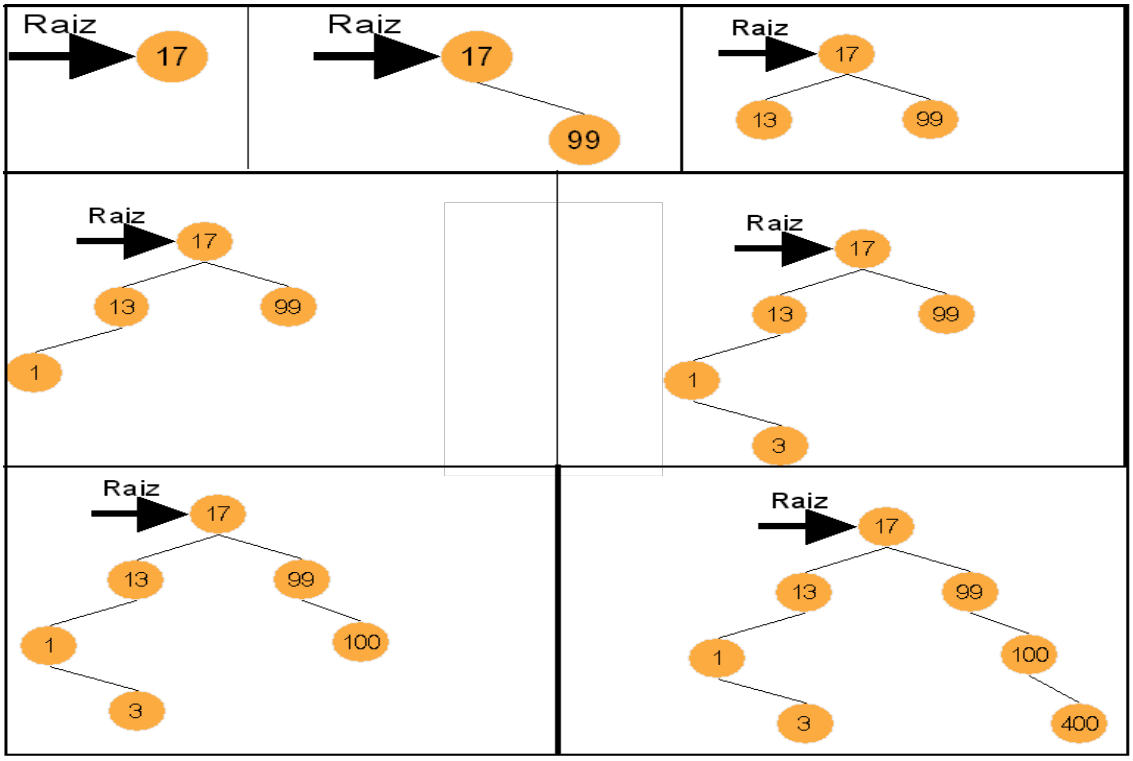

Figura 1: Ilustração da operação de inserção em uma ABB.

Você deve explicar a complexidade de cada função que implementar. A descrição da complexidade de cada função deve ter pelo menos 3 linhas. Todas as respostas devem ser fornecidas em um arquivo PDF. Inclusive o código fonte.