IMPLEMENTAÇÃO DE ÁRVORE BINÁRIA

O objetivo do trabalho é a implementação das rotinas de manipulação de uma árvore binária A. A chave da árvore binária A será uma outra árvore binária secundária B. Isto é, deverá suportar 1 árvore binárias principal, e N árvores binárias B, sendo N igual ao número de nós da árvore A. A ordenação da árvore principal será dada através da soma dos valores dos nós da árvore secundária.

ESPECIFICAÇÃO DETALHADA:

ÁRVORE A (PRINCIPAL)

ESTRUTURA DE DADOS

Deverá conter 1 ponteiro para esquerda, 1 para direita, 1 ponteiro para o pai, e um ponteiro para a chave. Este ponteiro será do tipo da árvore B (secundária).

FUNCIONALIDADES

Deverá conter funções de manipulação: 1-busca, 2-inclusão, 3-exclusão e 4-impressão em ordem, em formato de saída especificado abaixo. A chave de busca, inclusão e exclusão será dada através de uma árvore B passada como parâmetro (secundária).

Uma aplicação deverá ser implementada para usar as funções de manipulação da árvore. Esta aplicação será um programa que recebe instruções textuais e as executa, gerando uma resposta. Neste programa a árvore inicialmente está vazia, a cada comando a árvore é modificada, e ao final do arquivo (ou comandos) de entrada o programa termina.

ÁRVORE B (SECUNDÁRIA)

ESTRUTURA DE DADOS

Deverá conter 1 ponteiro para esquerda, 1 para direita e um inteiro armazenando a chave (o ponteiro para o pai é opcional).

FUNCIONALIDADES

Esta árvore deverá ter uma função de 1-criação através de entrada especificada, 2-impressão em ordem.

ENTRADA:

A entrada deve conter comandos. Os comandos são i (inserção), b, (busca) e r, (remoção). Todos os comando tem um parâmetro, que aparece separado do comando por um espaço. Os comandos aparecem um por linha. O parâmetero será no formato de parênteses aninhados, representando a árvore secundária B.

i (1) **Obs:** o valor de indexação desta sub-árvore é 1
i (10(8)(30)) **Obs:** o valor de indexação desta sub-árvore é 48
i (11(10)(17)) **Obs:** o valor de indexação desta sub-árvore é 38
i (15) **Obs:** o valor de indexação desta sub-árvore é 15
i (20(10(30)())()) **Obs:** o valor de indexação desta sub-árvore é 40
b (20(10(30)())()) **Obs:** o valor de indexação desta sub-árvore é 40
b (20(10(30)())()) **Obs:** o valor de indexação desta sub-árvore é 40
b (10(7)(23)) **Obs:** o valor de indexação desta sub-árvore é 40
r (11(10)(17)) **Obs:** o valor de indexação desta sub-árvore é 38
r (15)

Saída:

A cada comando executado uma saída deve ser gerada (execução da função 4). A saída deve representar o estado das árvores após a execução do comando. Para o comando de busca, a saída deve também indicar se o número total procurado está ou não na árvore e os nós percorridas (impressão das árvores secundárias). O formato da descrição da árvore é de colchetes aninhados, com separador de linhas entre os nós para a árvore principal A, e parênteses aninhados em uma única linha para as árvores secundárias B.

Exemplos:

A saída, para cada comando de alteração (“i” ou “r”), será a descrição da árvore resultante. Os comandos de inclusão acima resultarão na árvore abaixo:

[[(1) : 1
[
]
[(10(8)(30)) : 48
[(11(10)(17)) : 38
[15 : 15
]
[(20(10(30)())()) : 40
]
][
]
]

O comando de exclusão resultará na árvore abaixo

[[(1) : 1
[
]
[(10(8)(30)) : 48
[(11(10)(17)) : 38
[//**O nó (15)** estava nesta posição
]
[(20(10(30)())()) : 40
]
][
]
]

A entrada para os comandos de busca será uma sub-árvore B. Note que mais de uma árvore de entrada pode retornar a mesma sub-árvore. A saída para o comando de busca será a seqüência de nós visitados. O comando de busca resultará na impressão abaixo:

(1) : 1
(10(8)(30) : 48
(11(10)(17)) : 38
(20(10(30)())()) : 40

Requisitos:

O nome do executável deve ser **busca**.

Não deve ter nenhuma opção de linha comando. O resultado deve sempre lido e escrito na saída padrão.
O que deve ser entregue:

Além dos arquivos fonte, deve acompanhar um makefile e um arquivo README explicando o que foi feito e com o nome dos autores.

Para testar:

busca < teste.in > teste_stdin.out
ou
busca teste.in > teste_arq.out

ENTREGA

Entregar os arquivos fontes e o Makefile. Este deve ser compilado facilmente nos servidores do Dinf, através de comando make.

IMPORTANTE! DATA DE ENTREGA : 21 de novembro
MODO DE ENTREGA : enviar os arquivos por email para marcos.ddf _at_ inf.ufpr.br (até as 24h).
O trabalho pode ser individual ou em duplas. No assunto, preencher com “Entrega trabalho 057”.