sqtpm
[jhdapr]

voltar

**Trabalho:** 24-arvores
Data de abertura: 20/10/2021 12:00:00
Data limite para envio: 27/10/2021 23:59:59 (aberto)
Número máximo de envios: 31
Casos-de-teste abertos: casos-de-teste.tgz
Arquivos a enviar: entre 1 e 3

**Enviar:**

Linguagem: C
Arquivos: Escolher arquivos Nenhum arquivo selecionado
Enviar

---

# Árvore binária de busca

Escreva um programa que lê comandos da entrada e manipula dados armazenando-os em uma árvore de busca binária. A árvore deve ser representada de forma explícita.

Os comandos que seu programa deve interpretar são da seguinte forma:

- criar

  Ao ler o comando criar o programa deve criar uma árvore binária de busca vazia. Se já houver uma árvore sendo processada, todos os nós dela devem ser removidos.

- inserir x

  Ao ler o comando inserir x, onde x é um inteiro, o programa deve inserir a chave x na árvore de busca binária. A árvore não deve ter elementos repetidos, então se x já pertence à árvore ele não deve ser inserido de novo. Se não houver memória para essa operação, o programa deve imprimir "memoria insuficiente" em uma linha e deve continuar a execução.

- remover x

  Ao ler o comando remover x, onde x é um inteiro, o programa deve remover a chave x da árvore binária de busca. Se x não estiver na árvore o programa não deve fazer nada. O programa deve usar o sucessor de um nó para a substituição de um nó que tenha dois filhos.

- buscar x

  Ao ler o comando buscar x, onde x é um inteiro, o programa deve buscar a chave na árvore e imprimir "x esta na arvore" ou "x nao esta na arvore" em uma linha.

sqtpm
[jhdapr]

voltar

- pre-ordem
  em-ordem
  pos-ordem

  Ao ler um desses comandos o programa deve imprimir as chaves na ordem em que forem visitadas por um percurso em profundidade em pre-ordem, em-ordem ou em pós-ordem. As chaves devem ser impressas seguidas por um espaço, em uma única linha. Se a árvore estiver vazia então o programa deve imprimir "arvore vazia" em uma linha.

- minimo
  maximo

  Ao ler um desses comandos o programa deve imprimir a menor chave ou a maior chave na árvore, respectivamente, no formato "minimo: z" ou "maximo: z". Se a árvore estiver vazia então o programa deve imprimir "arvore vazia" em uma linha.

- sucessor x
  predecessor x

  Ao ler um desses comandos, onde x é um inteiro, o programa deve imprimir "sucessor de x: z" ou "predecessor de x: z", respectivamente. Se x não estiver na árvore ou se x não tiver sucessor ou predecessor o programa deve imprimir "nao ha sucessor de x" ou "nao ha predecessor de x" em uma linha.

- info

  Ao ler esse comando o programa deve imprimir o número de nós da árvore, o número de folhas na árvore e a altura da arvore, no formato "nos: x, folhas: z, altura: h". Se a árvore for vazia, o comando deve considerar a altura igual a zero.

- terminar

  Ao ler esse comando o programa deve desalocar a árvore e todos os nós dela e terminar.

## Exemplos de entrada e saída

| Entrada |
| --- |
| criar<br>inserir 30<br>inserir 50<br>inserir 40<br>inserir 10<br>inserir 20<br>remover 50 |

| Saída |
| --- |
| pre-ordem: 30 10 20 40<br>em-ordem: 10 20 30 40<br>pos-ordem: 20 10 40 30<br>nos: 4, folhas: 2, altura: 2 |

sqtpm
[jhdapr]

voltar

```
pre-ordem

em-ordem
pos-ordem
info
terminar
```

| Entrada |
| --- |
| criar<br>inserir 30<br>inserir 50<br>inserir 40<br>inserir 10<br>inserir 20<br>info<br>sucessor 15<br>sucessor 50<br>predecessor 50<br>buscar 30<br>buscar 5<br>minimo<br>maximo<br>terminar |

| Saída |
| --- |
| nos: 5, folhas: 2, altura: 2<br>nao ha sucessor de 15<br>nao ha sucessor de 50<br>predecessor de 50: 40<br>30 esta na arvore<br>5 nao esta na arvore<br>minimo: 10<br>maximo: 50 |

## Requisitos adicionais

- Não pode haver qualquer variável global. Uma variável é global se estiver declarada fora de alguma função (variáveis declaradas dentro da main não são globais, são locais à função main).

## Observações

- Vai ser mais fácil organizar seu programa se cada operação for realizada por uma função e cada função no programa implementar apenas uma funcionalidade.
- Não se esqueça do princípio KISS!

## Sobre organização do código e comentários

- Faça um programa organizado, bem indentado e que seja fácil de ler.
- Adicione comentários que vão ser úteis para entender o programa se você for relê-lo daqui a alguns anos: comentar cada linha vai ser redundante; documentar blocos de código e a estratégia usada na solução vai ser muito útil.

---