Universidade Federal da Bahia
MATA54 - Estruturas de Dados e Algoritmos II
Primeiro Trabalho Prático
Prof. Flávio Assis
Semestre 2022.2 - 26 de setembro de 2022

Árvore B+

1 Descrição Geral do Trabalho

Neste trabalho deverá ser implementada uma árvore B+, baseada na descrição de árvore-B do livro de Cormen et al., *Introduction to Algorithms*.

Os campos de cada registro (dados da aplicação) a ser armazenado na árvore são: uma *chave*, de valor inteiro não negativo; uma cadeia de, no máximo, 20 caracteres, representando um *nome*; e um outro valor inteiro não negativo, representando uma *idade*. O programa deverá conter duas constantes:

1. GRAU_MINIMO_INDICE: *grau mínimo* (como definido no livro de Cormen et al.) para a parte de índice da árvore. O valor inicial desta constante deve ser 3.
2. FATOR_CONJSEQ: fator que define o número mínimo e máximo de registros em um nó do conjunto sequência: cada nó poderá ter, no minimo, $FATOR\_CONJSEQ - 1$ e, no máximo, $2 * FATOR\_CONJSEQ - 1$ registros. O valor inicial deve ser 2. Quando houver apenas um nó no conjunto sequência, o número mínimo de registros é 1.

O programa deve funcionar para diferentes valores para essas constantes.

2 Formato de Entrada e Saída

A entrada constará de uma sequência de operações sobre a árvore. As operações e seus formatos estão descritos abaixo:

1. **insere registro**: esta operação conterá quatro linhas. A primeira linha conterá a letra 'i'. A segunda conterá um valor de chave. A terceira conterá uma sequência de até 20 caracteres, que corresponderá ao campo *nome*. Essa sequência conterá qualquer sequência de letras (minúsculas, sem acento, nem cedilha) e espaços, sendo que o primeiro e último caracteres não serão espaço. A quarta linha conterá um valor de idade.

   Esta operação verifica se já há registro no arquivo com o valor de chave indicado. Se sim, esta operação gera na saída a sequência de caracteres *'chave ja existente:'*, seguida de um espaço, seguido do valor da chave. Se a chave não existir, a operação insere o registro na árvore e gera, na saída, a sequência de caracteres *'insercao com sucesso:'*, seguida de um espaço, seguido do valor da chave.

2. **consulta registro**: esta operação conterá duas linhas. A primeira linha conterá a letra 'c'. A segunda conterá um valor de chave.

   Se houver registro na árvore com o valor de chave indicado, esta operação gera na saída a sequência de caracteres *'chave:'*, seguida de um espaço, seguido no valor da chave. Em seguida, na próxima linha escreve a sequência de caracteres *'nome:'*, seguida de um espaço, seguido do valor do nome associado ao registro. Na linha seguinte, deve aparecer a sequência de caracteres *'idade:'*, seguida de um espaço, seguido do valor da idade associada ao registro. Se não houver registro na árvore com o valor de chave indicado, esta operação gera na saída a sequência de caracteres *'chave nao encontrada:'*, seguida de um espaço, seguido do valor da chave.

3. **remove registro**: esta operação conterá duas linhas. A primeira linha conterá a letra 'r'. A segunda conterá um valor de chave.

   Se houver registro na árvore com o valor de chave indicado, esta operação deve remover o registro da árvore e gerar na saída a sequência de caracteres *'chave removida com sucesso:'*, seguida de um espaço, seguido do valor da chave. Se não houver, esta operação gera na saída a sequência de caracteres *'chave nao encontrada:'*, seguida de um espaço, seguido do valor da chave.

4. **imprime árvore**: esta operação conterá apenas uma linha, contendo a letra 'p'. O formato da saída desta operação está indicado na seção 3.

5. **imprime chaves em ordem crescente**: esta operação conterá apenas uma linha, contendo a letra 'o'.

   Se a árvore estiver vazia, esta operação gera, na saída, a sequência de caracteres *'árvore vazia'*. Caso contrário, esta operação lista as chaves dos registros armazenados na árvore em ordem crescente, uma por linha.

   Esta operação deve ser implementada seguindo-se o conjunto sequência.

6. **término da sequência de comandos:** a sequência de comandos será terminada por uma linha com a letra 'e'.

3 Formato de Impressão da Árvore

A impressão da árvore será feita seguindo um percurso em largura. Cada nó receberá um número sequencial, seguindo a ordem do percurso (a raiz é o nó 1). A impressão de cada nó será feita em uma linha. Não deve haver espaço entre linhas.

A impressão de cada nó seguirá o seguinte formato. Primeiramente será escrita a sequência de caracteres *'No:'*, seguida de um espaço, seguido do número sequencial do nó, seguido do caractere ':' (dois pontos), seguido de um espaço. Em seguida, na mesma linha, será impresso o conteúdo do nó.

Cada nó será impresso da seguinte maneira:

- *se o nó pertencer à parte de índice da árvore:* os apontadores e chaves devem ser impressos seguindo a estrutura do nó. Cada apontador deve ser impresso da seguinte maneira: a sequência de caracteres *'apontador:'*, seguida de um espaço, seguido do número sequencial do nó para o qual o apontador aponta. Cada chave será impressa da seguinte maneira: a sequência de caracteres *'chave:'*, seguida de um espaço, seguido do valor da chave. As impressões de apontadores e chaves devem estar separadas por um espaço. Se em um nó estiverem armazenados k registros, apenas as chaves destes registros e os apontares adjacentes a eles devem ser impressos.

- *se o nó for do conjunto sequência:* para cada registro armazenado no nó, deve ser impressa a sequência *'chave:'*, seguida de um espaço, seguido do valor da chave. Os valores dos demais campos do registro não devem ser impressos.

.

4 Observações

- O programa deve manter as atualizações em arquivo. A correção levará em consideração que o estado dos dados é persistente. Com isto, um teste pode ser feito, por exemplo, inserindo-se um registro, terminando a execução do programa e fazendo uma consulta ao registro em nova invocação do programa. Neste caso o registro deve ainda estar no arquivo.

- Lembre-se de que é assumido que a memória principal é insuficiente para armazenar todos os dados. Portanto, por exemplo, uma implementação que mantém a estrutura do arquivo em memória principal e a salva por completo no arquivo será considerada inaceitável.

- O arquivo deve ser armazenado em formato binário.
- O programa não deve gerar nenhum caractere a mais na saída, além dos indicados acima. Em particular, o programa não deve conter menus.
- Não pode haver espaço entre linhas na saída. A saída deve apresentar os caracteres em letras minúsculas e sem acento.
- Trabalho individual.
- Data de entrega: **23/10/2022 - prazo inadiável! Somente serão aceitos trabalhos entregues até essa data, até 23:59h!**
- Linguagens de programação permitidas: C, C++, Java ou Python.
  - **Importante:** Para as linguagens C, C++ e Java, somente trabalhos feitos utilizando-se os seguintes compiladores serão aceitos:
    - C: gcc ou djgpp
    - C++: g++ ou djgpp
    - Java: compilador java do JDK (mais recente)

  No caso de Python, deve ser indicada a versão utilizada.

  Não serão compilados trabalhos em outros compiladores! Erros ocasionados por uso de diferentes compiladores serão considerados erros do trabalho!
- O aluno deverá submeter seu trabalho através do *moodle*.