Universidade Federal da Bahia
MATA54 - Estruturas de Dados e Algoritmos II
Terceiro Trabalho Prático
Prof. Flávio Assis
Semestre 2022.1 - 9 de maio de 2022

Árvore *k-d*

1 Descrição Geral do Trabalho

Neste trabalho deverá ser implementada uma estrutura de armazenamento de registros baseada em *árvores k-d*, para busca por chave secundária. A estrutura será utilizada para armazenar dados sobre obras literárias.

O programa irá receber uma sequência de registros e criará uma estrutura em arquivos para acessá-los. Como nos trabalhos anteriores, todos os dados devem ser armazenados em arquivo, de maneira persistente. Ou seja, os dados devem poder acessados fazendo-se mais de uma execução do programa.

A árvore *k-d* será usada como índice para busca a registros armazenados em *páginas*. Embora tenha que ser armazenada em memória secundária, neste trabalho, especificamente, será considerado que **a parte de índice da árvore *k-d* pode ser armazenada completamente na memória principal**. Da maneira usual, não se poderá assumir que o conteúdo das páginas cabe na memória principal, apenas o índice da árvore.

O programa deve ter uma constante, de nome $NREGSPORPAGINA$, que indicará o número máximo de registros que podem ser armazenados em uma página. Essa constante deve ser iniciada com o valor 3. O programa poderá ser testado com diferentes valores desta constante.

2 Formato de Entrada e Saída

A entrada constará de uma sequência de operações. Na primeira invocação do programa, a primeira linha da entrada conterá um número inteiro. Esse número indicará o número de registros que serão usados para se criar o índice da árvore *k-d*. Em seguida, haverá os dados dos registros que serão usados como índice, um campo por linha. Além de serem usados para a criação do índice, esses registros já devem ser também armazenados nas páginas. Após a criação da árvore *k-d*, o programa deve gerar na saída a sequência de caracteres *arvore k-d gerada*. Em seguida, haverá uma lista de operações. Em outras invocações do programa, a entrada conterá apenas uma sequência de operações.

A árvore a ser criada será uma árvore 2-*d*. A dimensão relativa aos níveis ímpares da árvore (considere que a raiz da árvore está no nível 1) representará os nomes de autores. A dimensão relativa aos níveis pares representará os anos de publicação da obra.

As operações e seus formatos estão descritos abaixo. Os formatos descritos descrevem como as entradas, de fato, serão. Não há necessidade de validar se seguem o formato indicado.

1. **insere registro**: esta operação será usada para inserir registros adicionais, além daqueles usados para a criação do índice.

   Essa operação conterá cinco linhas. A primeira conterá a letra 'i'. A segunda linha conterá uma sequência de caracteres, representando um *nome de autor*. A terceira linha conterá uma outra sequência de caracteres, representando um *título de uma obra literária*. A quarta linha conterá um valor inteiro não negativo, representando uma *ano*. A quinta linha conterá outra sequência de caracteres, representando o *nome de um arquivo*.

As sequências de caracteres terão tamanho máximo de 20 caracteres. As sequências representando nome de autor e título de obra poderão conter qualquer letra (minúsculas, sem acento, nem cedilha) e espaços, sendo que o primeiro e último caracteres serão letras. Sequências representando nome de arquivo não poderão conter espaços e, opcionalmente, poderão conter um único ponto ('.'). O campo relativo ao nome de arquivo será, nesse trabalho, apenas mais um dado que faz parte do registro. Ele será usado no quarto trabalho, que será uma continuação desse.

Após a inserção, o programa deve gerar na saída a sequência de caracteres *inserido registro com nome:*, sequido de um espaço, seguido do valor do campo *nome*. Nesse trabalho iremos assumir que não haverá condição de insucesso de inserção (observe que o campo *nome* não é chave).

2. **consulta simples**: esta operação conterá duas linhas. A primeira linha conterá a letra 'c'. A segunda conterá uma sequência de caracteres representando um nome de autor.

   Se houver registros na árvore com o valor de nome indicado, esta operação gera na saída, para cada registro encontrado, um após ou outro, a sequência de caracteres *'nome:'*, seguida de um espaço, seguido no valor da nome. Em seguida, na próxima linha escreve o valor do campo relativo ao título da obra. Na próxima linha escreve o valor referente ao ano. Por fim, na linha seguinte, escreve o valor do campo relativo ao nome de arquivo.

   Se não houver registro com o valor de nome indicado, esta operação gera na saída a sequência de caracteres *'nao foi encontrado registro com nome:'*, seguida de um espaço, seguido do valor de nome informado na operação.

3. **consulta por faixa de nomes de autores**: esta operação conterá três linhas. A primeira linha conterá a letra 'n'. A segunda e terceira linhas conterão, cada uma, uma sequência de caracteres. Essas sequências seguem o mesmo formato de nomes, como indicado na descrição da operação *insere registro*.

   Seja $n1$ o nome indicado na segunda linha da operação. Seja $n2$ o nome indicado na terceira linha. Essa operação apresentará os registros armazenados na árvore cujos valores para o campo de nome do autor sejam maiores ou iguais a $n1$ e menores ou iguais a $n2$, considerando-se a ordem alfabética.

4. **consulta por faixa de anos**: esta operação conterá três linhas. A primeira linha conterá a letra 'a'. A segunda e terceira linhas conterão, cada uma, um número inteiro não negativo

   Seja $a1$ o número indicado na segunda linha da operação. Seja $a2$ o número indicado na terceira linha. Essa operação apresentará os registros armazenados na árvore cujos valores para o campo de ano sejam maiores ou iguais a $a1$ e menores ou iguais a $a2$.

5. **consulta por faixa de nomes de autores e anos**: esta operação conterá cinco linhas. A primeira linha conterá a letra 'q'. A segunda e terceira linhas conterão, cada uma, uma sequência de caracteres. Essas sequências seguem o mesmo formato de nomes, como indicado na descrição da operação *insere registro*. A quarta e quinta linhas conterão, cada uma, um númeor inteiro não negativo.

   Seja $n1$ o nome indicado na segunda linha da operação. Seja $n2$ o nome indicado na terceira linha. Seja $a1$ o número indicado na segunda linha da operação. Seja $a2$ o número indicado na terceira linha. Essa operação apresentará os registros armazenados na árvore cujos valores para o campo de nome do autor sejam maiores ou iguais a $n1$ e menores ou iguais a $n2$, considerando-se a ordem alfabética, **e** cujos valores para o campo de ano sejam maiores ou iguais a $a1$ e menores ou iguais a $a2$. Ou seja, aqueles registros que estarão, ao mesmo tempo, nas faixas de nome e de ano indicadas.

6. **imprime índice da árvore:** esta operação conterá apenas uma linha, contendo a letra 't'. Essa operação lista os valores referentes a cada nó, um por linha, seguindo a ordem de visita de um percurso *em ordem* da árvore. Para cada nó, os dados devem ser apresentados no seguinte

formato: a sequência de caracteres *'nome:'* ou *'ano:'*, dependendo do valor associado à dimensão do nível, seguida de um espaço, seguido do valor correspondente (nome ou ano), seguido de um espaço, seguido pela sequência de caracteres *'fesq:'*, seguida por um espaço, seguido pelo valor (nome ou ano) armazenado no nó filho à esquerda, seguido por um espaço, seguido pela sequência de caracteres *'fdir:'*, seguida por um espaço, seguido pelo valor (nome ou ano) armazenado no nó filho à direita. Caso o filho à esquerda ou à direita corresponda a uma página, a sequência de caracteres *'pagina'* deve ser gerada.

7. **imprime página:** essa operação conterá duas linhas. A primeira conterá a letra 'p'. A segunda conterá um valor inteiro não negativo. Esse valor indicará um *índice* de página. Esse índice corresponde à ordem em que a página seria visitada, seguindo um percurso em-ordem da árvore. A primeira página terá índice 0.

   Essa operação apresentará inicialmente a sequência de caracteres *'pagina:'*, seguido de um espaço, seguido do valor inteiro indicado. Em seguida, apresentará os registros armazenados na página, um valor de campo por linha.

   Quando uma página fica cheia, páginas adicionais são criadas, formando uma lista encadeada. Na impressão de uma página de índice i, os registros de todas as páginas encadeadas (se houver) devem também ser impressos.

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

3 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. Nesse trabalho, assume-se que o índice da árvore k-d cabe na memória principal, *mas não o conteúdo das páginas*. Portanto, por exemplo, uma implementação que mantém uma estrutuda com todos os dados em memória principal e a salva por completo no arquivo será considerada inaceitável.

- Os arquivos devem ser armazenados 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.

- Todos os arquivos criados pelo programa dever ter extensão ".dat".

- Trabalho individual ou em dupla.

- Além do código fonte do programa, deve ser entregue também um relatório sucinto, indicando a lógica de criação da árvore k-d e como ela foi armazenada em memória secundária.

  Importante: Se não houver entrega desse relatório, a nota do trabalho pode ser **bastante** afetada, uma vez que o relatório é necessário para se entender como a implementação foi feita.

- Data de entrega: **05/06/2022**

- 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 trabalho deve ser entregue através do *moodle*.