ALGORITMOS E ESTRUTURAS DE DADOS II

Trabalho – Primeiro Semestre – 2011

Prof. Ricardo J. G. B. Campello

Estagiário do PAE: Danilo Horta

Um pesquisador deseja manter um cadastro de suas referências bibliográficas (artigos e livros) em disco, e acabou de contratá-lo para resolver o problema. Para cada referência, o pesquisador deseja manter os seguintes atributos:

- Código da referência (composição das três primeiras letras do nome do primeiro autor e ano da publicação, por ex. SHI90 – esse campo é chave, não existirão valores idênticos).
- Título (*string* com o título).
- Autor (sobrenome e iniciais do primeiro autor, por ex.: Schimman, D.E.).
- Ano de publicação (por ex.: 1990).
- Veículo (*string* com o nome do congresso ou periódico e outros dados adicionais, como páginas, volume e número, local, etc.).

Você deverá desenvolver um programa que permita ao pesquisador:

- Inserir uma nova referência, atualizando o índice.
- Remover uma referência a partir da chave. O registro deverá ser localizado acessando o índice. A remoção deve colocar os caracteres ***|** nas primeiras posições do registro removido. O espaço do registro removido não deverá ser reutilizado para novas inserções, as quais deverão ser colocadas sempre no fim do arquivo (supõe-se, assim, que um programa de compactação eliminará os espaços disponíveis em outro momento. Não é necessário implementar esse programa).
- Buscar uma referência a partir da chave. A busca deve ser realizada acessando o índice.

Ao final da execução, dois arquivos deverão existir: um **arquivo de índice** e um **arquivo de dados**.

Arquivo de Índice

O arquivo de índice deve ser **binário**. A forma de organização do índice utilizada no arquivo **deve** ser uma **árvore-B**. Note que os "ponteiros" para páginas armazenadas no arquivo devem ser *bytes offsets* para a localização da página em questão no arquivo. A árvore-B da sua implementação deve seguir as seguintes especificações:

- Cada página da árvore deve comportar 4 entradas de índice.
- Os endereços (*offsets*) devem ser do tipo *int* de 4 *bytes* (tanto os *offsets* das próximas páginas da árvore, quanto os *offsets* dos registros). *Offsets* nulos (não utilizados) devem ter valor "-1".
- Páginas excluídas devem ser marcadas com os caracteres ***|** nas primeiras posições, seguido do *offset* da próxima página livre. As páginas livres devem ser mantidas em uma

pilha (conforme visto em aula). Sempre que uma página é excluída, ela passa a ser o topo da pilha, e o *offset* do antigo topo deve ser colocado após os caracteres ***|**.

- Novas páginas devem ser colocadas na primeira página livre (topo da pilha de disponíveis). Caso não haja nenhuma página livre, novas páginas devem ser colocadas no final do arquivo.
- O cabeçalho do arquivo de índice deve conter o *offset* da página raiz, seguido do *offset* do topo da pilha de disponíveis. A árvore começa imediatamente em seguida, sem caracteres delimitadores nem quebras de linha.
- Caso a pilha de disponíveis esteja vazia, o *offset* deve ser "-1".

Como os campos do índice possuem tamanho fixo, tudo deve ser armazenado sem caracteres delimitadores nem quebras de linha, utilizando a estrutura tradicional de árvore-B, do seguinte modo:

$$O_P\,«C_R\,O_R»\;O_P\,«C_R\,O_R»\;O_P\,«C_R\,O_R»\;O_P\,«C_R\,O_R»\;O_P$$

Cada O_P representa o *offset* para outra página da árvore, e cada par $«C_R\,O_R»$ representa o par composto pela chave de um registro e seu *offset* no arquivo de dados. Note que os símbolos *«* e *»* foram colocados apenas para facilitar a leitura, as páginas devem ser armazenadas sem delimitadores. A linha acima representa uma única página da árvore-B, a próxima página deve começar imediatamente após o término da primeira página, sem caracteres delimitadores nem quebras de linha. Seu arquivo de índice deverá se chamar **index.dat**.

Estrutura do arquivo de dados

O arquivo de dados deve ser ASCII (**arquivo texto**) e organizado em registros de tamanho fixo de 256 *bytes.* Os campos título, autor e veículo devem ser de tamanho variável. Os demais campos devem ser de tamanho fixo. A soma de *bytes* dos campos fornecidos (incluindo os delimitadores necessários) nunca irá ultrapassar 256 *bytes*. Os campos do registro devem separados pelo caractere delimitador **@** (arroba). Note que caso o registro tenha menos de 256 *bytes*, o espaço adicional deve ser preenchido de forma a completar os 256 *bytes*. Segue abaixo um exemplo com dois registros. Os espaços adicionais foram preenchidos com o caractere **#** (sustenido).

key01@título1@autor1@ano1@veículo1@##...##key02@título2@autor2@ano2@veículo2@##...

O arquivo de dados não deverá conter cabeçalho e deverá se chamar **data.txt.**

Interação com o usuário

Seu programa deve permitir interação com o usuário pelo console/terminal (modo texto). As seguintes operações devem estar disponíveis:

- Inserção de referência: O usuário deve ser capaz de inserir uma nova referência. Seu programa deve ler os seguintes campos: **chave**, **título**, **autor**, **ano** e **veículo.** Note que a chave é também inserida pelo usuário, você não precisa gerar a chave.

- Remoção de referência: O usuário deve ser capaz de remover uma referência. Caso ela não exista, seu programa deve informar tal fato ao usuário. Para remover uma referência seu programa deve solicitar como entrada ao usuário somente o campo **chave**.
- Busca de referência: O usuário deve ser capaz de buscar por uma referência. Caso ela não exista, seu programa deve informar tal fato ao usuário. Para buscar uma referência seu programa deve solicitar como entrada ao usuário somente o campo **chave**. Caso a referência exista, todos os seus dados devem ser impressos na tela de forma formatada.
- Finalizar a execução: O usuário deve ser capaz de encerrar a execução do programa. Ao final da execução, feche todos os arquivos e libere toda a memória alocada pelo seu programa.

Observações Importantes

1. Implemente seu programa usando a linguagem **C padrão ANSI**. Caso estiver usando o compilador GCC, utilize a *tag* -ansi. Só utilize bibliotecas padrões da linguagem **C ANSI**;
2. A clareza da sua implementação será avaliada;
3. Use comentários **relevantes** (e apenas relevantes) em seu código, pois esse aspecto será avaliado;
4. Os trabalhos deverão ser feitos em **duplas** e devidamente identificados com os nomes e números USP de cada um dos alunos;
5. Trabalhos contendo apenas o código, sem comentários detalhados, não serão considerados;
6. Serão anulados aqueles trabalhos nos quais for detectado qualquer tipo de plágio, independentemente da sua origem;
7. A avaliação abrangerá aspectos como exatidão, qualidade do conteúdo e do documento, esforço e participação de todos os alunos, interação com o usuário, dentre outros. Para a avaliação, o código-fonte entregue será compilado por meio do compilador GCC (em ambiente GNU/Linux) usando as *tags* **-ansi**, para verificar compatibilidade com o padrão ANSI da linguagem C, e **-Wall**, para verificar a existência de algum *warning*, e executado frente a casos de teste.
8. Vazamento de memória, referência a valores de variáveis não inicializadas e outros defeitos serão levados em conta na avaliação (se estiver usando GNU/Linux, pode-se usar o programa Valgrind para ajudá-lo na detecção de tais problemas).

Entrega

- O trabalho deverá ser entregue pessoalmente e em formato digital ao estagiário PAE. **A data final de entrega é 06/06/2011 das 18:30 às 20:30 na sala 6-206**. Os alunos que desejarem entregar o trabalho antes da data limite devem entrar em contato com o estagiário PAE via email (horta@icmc.usp.br) para agendar a entrega, também pessoalmente.
- Após a data de entrega do trabalho, o professor poderá chamar a seu critério duplas para fazer entrevista, com o intuito de conferir o conhecimento de ambos os componentes da dupla quanto ao trabalho.