SCC0503 (Algoritmos e Estruturas de Dados II) — Prof. Moacir P. Ponti Junior

Trabalho 4

Implemente sua atividade em duplas, sem compartilhar, olhar código de outras duplas, ou buscar na Internet. Procure usar apenas os conceitos já vistos nas aulas. Plágio nesse trabalho gera nota zero para todos os envolvidos.

Criação e Organização de Índices com Árvore-B

O colecionador de filmes, insatisfeito com o tempo gasto para realizar as operações de busca, resolveu contratá-lo para criar uma versão mais eficiente do programa de gerenciamento da coleção.

Tarefa

Desenvolva um programa que permita ao colecionador organizar seus filmes. O programa deverá permitir:

1. Inserir um novo filme no catálogo.
2. Modificar o campo **nota** de um filme a partir da chave primária.
3. Buscar filmes a partir da chave primária.
4. Listar todos os filmes no catálogo ordenado pela chave primária.

Para realizar essa tarefa será necessário organizar 2 arquivos distintos: (a) um arquivo de dados que conterá todos os registros e (b) um arquivo de índice primário. Não é necessário manter um arquivo de índice secundário.

Estrutura do arquivo de dados

O arquivo de dados segue a mesma estrutura utilizada no Trabalho 3, ou seja, deve ser ASCII (arquivo texto) e organizado em registros de tamanho fixo de 192 bytes. Os campos de títulos, diretor e país devem ser de tamanho variável. Os demais campos devem ser de tamanho fixo: ano (4), nota (1) e chave primária (5). A soma de bytes dos campos fornecidos (incluindo os delimitadores necessários) nunca irá ultrapassar 192 bytes. Os campos do registro devem separados pelo caractere delimitador `@` (arroba). Cada registro terá 7 delimitadores, mais 10 espaços ocupados pelos campos de tamanho fixo. Você poderá fixar um tamanho máximo para cada campo: título, título original, diretor e país de forma a garantir que ocupem, juntos, um máximo de 175 bytes.

Caso o registro tenha menos de 192 bytes, o espaço adicional deve ser marcado de alguma forma a completar os 192 bytes. Segue abaixo um exemplo com três registros. Os espaços adicionais foram preenchidos com o caractere `#`.

```
UND90@O Ataque dos Vermes Malditos@Tremors@Underwood, Ron@1990@E
UA@9@#######################################################
#######################################################
VIB73@Thriller, Um Filme Cruel@Thriller - A Cruel Picture Or The
y Call Her One Eye@Vibenius, Bo Arne@1973@Suécia@4@############
#######################################################
SAN98@Cinderela Baiana@Idem@Sanchez, Conrado@1998@Brasil@0@#####
#######################################################
#######################################################
```

Note que não há quebras de linhas no arquivo (elas foram inseridas aqui apenas para exemplificar a sequência de registros).

O arquivo de dados não deverá conter cabeçalho e deverá se chamar `movies.dat`.

Instruções para as operações com registros:

- Inserção: cada filme inserido no catálogo dever ser inserido no final do arquivo de dados e atualizado nos índices.
- Atualização: o registro deverá ser localizado acessando o índice primário. A nota deverá ser atualizada no registro na mesma posição em que está (não deve ser feita remoção seguida de inserção).

Índice

Deverá ser criado um arquivo **binário** chamado `ibtree.idx`, contendo as chaves primárias e os RRNs dos respectivos registros. A forma de organização do índice utilizada no arquivo deve ser uma **Árvore-B**. Os "ponteiros" para páginas armazenadas no arquivo devem ser os RRNs para localização das páginas em no arquivo. A árvore-B deve seguir as seguintes especificações:

- Cada página da árvore deve comportar 4 entradas de índice.
- Os endereços (RRN) devem ser do tipo `int` de 4 bytes (tanto os RRNs dos descendentes de cada página, quanto os RRNs dos registros do arquivo de dados). RRNs nulos (não utilizados) devem ter valor "-1".
- Novas páginas devem ser colocadas no final do arquivo.
- O cabeçalho do arquivo de índice deve conter o RRN da página raiz. A árvore começa imediatamente em seguida, sem caracteres delimitadores nem quebras de linha.

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

```
RRNp1<Cr1,RRNr1>RRNp2<Cr2,RRNr2>RRNp3<Cr3,RRNr3>Op<Cr3,RRNr3>RRNp4
```

sendo que: RRNp$_i$ é o RRN das página descendente i, e <Cr$_j$,RRNr$_i$> é um par: Cr$_i$ = chave do registro j, e RRNr$_j$ = RRN do registro j. Note que os símbolos < e > foram colocados apenas para facilitar a leitura, ou seja, 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.

Interação com o usuário

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

Inserção de filme : O usuário deve ser capaz de inserir um novo filme. Seu programa deve ler os seguintes campos: **título em português**, **título original**, **diretor**, **ano de lançamento**, **país** e **nota**. Note que a chave **não é** inserida pelo usuário, você precisa gerar a chave para gravá-la no registro.

Busca de filme : O usuário deve ser capaz de buscar por um filme pela chave, ou seja, solicitar ao usuário a chave. Caso o filme não exista, seu programa deve informar tal fato ao usuário. Caso o filme 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.

Implementação

Implementar uma biblioteca de manipulação de arquivos para o seu programa, contendo as seguintes funcionalidades:

- Uma estrutura de dados Árvore-B.
- Verificar se o arquivo de dados existe
- Verificar se o índice primário existe
- Criar/recriar o índice primário: deve refazer o índice primário a partir do arquivo de dados e substituir, caso haja, um índice existente no disco.
- Inserir um registro: modificando o arquivo de dados no disco, e o índice na memória principal.

Utilizar as linguagens `C` ou `C++`. Separar a interface da implementação (arquivos `c`/`cc` e `h`, e montar um `Makefile` para compilar o código.

- Mais informações sobre como usar Makefiles e gerar bibliotecas pode ser encontrado em: `http://wiki.icmc.usp.br/images/0/0a/ApostilaMakefiles2011.pdf`

Pontos Extra!!!

Implementar uma função que permita a remoção de um registro, modificando o arquivo de dados no disco, e o índices (Árvore-B). Caso o filme não exista, seu programa deve informar tal fato ao usuário. Para remover um filme, seu programa deve solicitar como entrada ao usuário somente o campo chave. Na remoção, o registro deverá ser localizado acessando o índice primário. A remoção deve colocar os caracteres `*|` nas primeiras posições do registro removido do arquivo de dados. O espaço do registro removido **não** deverá ser reutilizado para novas inserções. Implementar a remoção na Árvore-B.

A implementação correta dessa função vale 3 pontos na nota final do trabalho, ou seja, você pode ficar com até nota 13 nesse trabalho. Note que não será considerada a implementação parcial ou não funcional da remoção. Ou você implementa tudo corretamente e ganha 3 pontos ou ganha 0 ponto.

ATENÇÃO

Leia atentamente os item a seguir:

- O projeto deverá ser entregue apenas pelo Sistema de Submissão de Programas (SSP)[1], escolhendo as seguintes opções:
  - No campo "Título do Exercício": `Trabalho04`;
  - No campo "Linguagem de Programação": `Zip`.

  Caso o trabalho seja feito em duplas, **TODOS** os membros do grupo devem submeter uma cópia do projeto (caso apenas um dos membros submeta o trabalho, a nota será **dividida** por 2).
- Para simplificar, compacte o *diretório* que contém os arquivos implementados.
- Não utilize acentos nos nomes de arquivos.
- Dúvidas conceituais deverão ser colocadas nos horários de atendimento. Dificuldades em implementação, por favor, envie e-mail para o estagiário PAE com o assunto `[trab4]duvida`, anexando o código, especificando o problema e apresentando o número USP.
- A detecção de cópia de parte ou de todo código-fonte, de qualquer origem, implicará reprovação direta no trabalho. Partes do código cujas **ideias** foram desenvolvidas em colaboração com outro(s) aluno(s) devem ser devidamente documentadas em comentários no referido trecho. O que **NÃO** autoriza a cópia de trechos de código nem a codificação em conjunto. Portanto, compartilhem ideias, soluções, modos de resolver o problema, mas não o código. Qualquer dúvida entrem em contato com o professor.

[1] `http://netuno.icmc.usp.br/ssp01/`