

Trabalho 1

Disciplina: Estrutura de Dados II

Prof. Dr. Lucas C. Ribas

1 Criação e Organização de Arquivos e Indexação

Um colecionador deseja manter um cadastro de seus filmes em um arquivo. Para cada filme, o colecionador deseja manter os seguintes atributos:

- Código do filme (composição das três primeiras letras do último sobrenome do diretor e dois últimos dígitos do ano de lançamento do filme, e.g., WEL41) — esse campo é chave, não existirão valores idênticos.
- Título em português.
- Título original (se o título em português for o mesmo, deve-se utilizar a palavra Idem).
- Diretor (sobrenome e nome do diretor, e.g.: Welles, Orson).
- Ano de lançamento (e.g., 1941).
- País (país no qual o filme foi produzido).
- Nota (inteiro entre 0 e 9 com a nota dada pelo colecionador ao filme).

Tarefa

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

1. Inserir um novo filme no catálogo.
2. Remover um filme a partir da chave primária.
3. Modificar o campo nota de um filme a partir da chave primária.
4. Buscar filmes a partir da (a) chave primária ou do (b) título em português.
5. Listar todos os filmes no catálogo.

Para realizar essa tarefa será necessário organizar 3 arquivos distintos:

1. um arquivo de dados que conterá todos os registros,
2. um arquivo de índice primário,
3. um arquivo de índice secundário para o título em português.

Estrutura do arquivo de dados

O arquivo de dados deve ser ASCII (arquivo texto) e organizado em registros de tamanho fixo de 192 bytes (caracteres). 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 ser 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 (192-7-10). 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 #.

```
ROC64@Deus e o Diabo na Terra do Sol@Idem@Rocha, Glauber@1964@Brasil@8#####  
#####  
WEL41@Cidadão Kane@Citizen Kane@Welles, Orson@1941@Estados Unidos@9#####  
#####  
KUB64@Dr. Fantástico@Dr. Strangelove or: How I Learned to Stop Worrying and Love the Bomb@Kubrick, Stanley@1964@Estados Unidos@9#####
```

Figura 1: Exemplo de arquivo de dados com registros e campos separados pelo caractere @.

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 deve ser inserido no final do arquivo de dados e atualizado nos índices.
- **Remoção:** o registro deverá ser localizado acessando o índice primário. A remoção deve colocar os caracteres *| nas duas primeiras posições do registro removido. O espaço do registro removido **não** deverá ser reutilizado para novas inserções.
- **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).

Índices

Dois arquivos com índices serão criados:

- `iprimary.idx`: índice primário, contendo a chave primária e o RRN do respectivo registro, ordenado pela chave primária.
- `ititle.idx`: índice secundário, contendo o título e a chave primária do respectivo registro, ordenado pelo título.

Deverá ser criada uma rotina para a criação de cada índice. Os índices serão criados carregando para a memória principal os dados necessários, procedendo a ordenação em memória principal e a seguir gravando o arquivo de índice ordenado. Note que o ideal é que `iprimary.idx` seja o primeiro a ser criado.

Ao iniciar o programa, esse deverá carregar os índices (primário e secundário) para a memória principal e durante a execução do programa, manipular os índices na RAM. Ao fechar o programa, os arquivos de índice deverão ser atualizados no disco. Para que isso funcione corretamente, o programa, ao iniciar:

1. Verificar se existe um arquivo de dados.
 - a) Se existir: abrir para escrita e leitura e passar para o item 2.
 - b) Se não existir: i. criar o arquivo de dados no disco, abrindo para escrita e leitura.
2. Verificar se existem os arquivos de índice:
 - a) Se existirem: verificar se estão consistentes com o arquivo de dados (usar uma flag no cabeçalho para isso). i. Se estiverem consistentes: carregar os índices para a RAM. ii. Senão: refazer os índices na RAM e gravá-los no disco. Lembre-se de atualizar as flags de mudança nos índices quando realizar a primeira mudança no índice em RAM (nas demais modificações não é mais necessário já que a flag já estará setada).
 - b) Se não existirem: criar os índices na RAM e gravá-los no disco.

A implementação dos índices em RAM é livre, podendo usar listas, listas invertida, vetores e matrizes. Em arquivo, o índice primário pode ser salvo em registros com dois campos: chave primária e RRN. O índice secundário pode ser salvo em arquivos com registros com os campos: título e chave primária. Em ambos os arquivos, pode ser usado tamanho fixo ou variável e separador @. As buscas em RAM no índice podem ser feitas por qualquer algoritmo de busca. Lembre-se que os índices devem estar sempre ordenados.

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.

- **Remoção de referência:** O usuário deve ser capaz de remover um filme. Caso ele 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. Lembre-se que além da marcação do registro removido no arquivo de dados (como descrito acima), deve-se também ser removido do índice primário.
- **Busca de filme:** O usuário deve ser capaz de buscar por um filme.
 - **pela chave:** 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.
 - **pelo título:** solicitar ao usuário um título. Caso não hajam filmes com o título, o programa deve informar ao usuário. Caso existam um ou mais filmes com o título informado, o registro completo desses filmes deverão ser mostrados na tela de forma formatada. A busca deve ser realizada primeiro no índice secundário para obtenção das chaves, em seguida, os RRN são obtidos com a busca no arquivo primário.
- **Finalizar a execução:** O usuário deve ser capaz de encerrar a execução do programa. Ao final da execução, atualize os índices no disco, feche todos os arquivos e libere toda a memória alocada pelo seu programa.

Lembre-se: enquanto os índices primários e secundários podem ser mantidos em RAM e somente salvos na finalização do programa, os dados devem ser mantidos no arquivo e só carregado um registro através do RRN obtido por busca realizada nos índices.

Implementação

O seu programa deve ter as seguintes funcionalidades implementadas, que podem, opcionalmente, serem colocadas em uma biblioteca de manipulação de arquivos:

- Uma estrutura de dados para armazenar os índices na memória principal.
- Verificar se o arquivo de dados existe.
- Verificar se o índice primário existe.
- Verificar se o índice secundário existe.
- Criar o índice primário: deve refazer o índice primário a partir do arquivo de dados e substituir, caso haja, um índice existente no disco.
- Criar o índice secundário: deve refazer o índice secundário a partir do arquivo de dados e substituir, caso haja, um índice existente no disco.
- Carregar os índices do disco para a memória principal.
- Inserir um registro: modificando o arquivo de dados no disco, e os índices na memória principal.

- Alterar um registro: modificando o arquivo de dados no disco, e os índices na memória principal.
- Remover um registro: modificando o arquivo de dados no disco, e os índices na memória principal.
- Buscar pela chave primária
- Buscar pela chave secundária
- Atualizar todos os índices: deverá ser chamada ao finalizar o programa

Pontos Extra!!!

Implementar uma função que permita compactar o arquivo de dados, de forma a deletar fisicamente os registros marcados como removidos e movendo os registros subsequentes. A implementação correta dessa função e sua inserção no programa (seu programa deve permitir ao usuário executar essa compactação a qualquer momento), vale 2 pontos na nota final do trabalho, ou seja, você pode ficar com até nota 12 nesse trabalho.

2 ATENÇÃO

Leia atentamente os itens a seguir:

- O trabalho é individual.
- O projeto deverá ser entregue apenas pelo classroom até a data limite de **22/09**.
- Submissão deve ser realizado no seguinte formato: arquivos de dados e códigos em C compactado como NomeSobrenome_RA.zip.
- Para cada dia de atraso na entrega, 2 pontos serão descontados da nota máxima do trabalho (10 + 2 pontos extra).
- Dúvidas deverão ser esclarecidas pelo classroom. Preferencialmente, envie a dúvida para que todos possam ver, pois pode auxiliar outros colegas também. Uma aula de dúvidas também será agendado em Setembro.
- A detecção de cópia de parte ou de todo código-fonte, de qualquer origem, implicará reprovação direta no trabalho.
- Entrevistas podem ser agendadas para correção do trabalho.
- Este trabalho poderá servir como base para os próximos, então o seu desenvolvimento e entrega é imprescindível.

¹Adaptação de Notas de Aula Prof. Moacir Ponti. ICMC/USP.