Universidade de São Paulo
Instituto de Ciências Matemáticas e de Computação
Departamento de Ciências de Computação
SCC0221 – Introdução à Ciência da Computação I

Trabalho: SGBD

1. Descrição

Considere um sistema em que você recebe, como parâmetro, o nome de um arquivo textual contendo os campos e os tipos de dados de cada campo. Um exemplo do arquivo está descrito abaixo:

```
filename: registros.reg
key-name: codigo
key-type: int
field-name: nome
field-type: char[80]
field-name: idade
field-type: int
field-name: bolsa
field-type: double
```

Chamaremos esse arquivo de **metadados**. Ele está escrito no formato de texto puro e, portanto, pode ser lido com o fscanf ou qualquer outra função de mesmo propósito. O "filename" consiste no nome do arquivo onde os registros que você coletará estarão guardados, esse deverá ser escrito em formato binário. Os campos "key-name" e "key-type" representam respectivamente o nome do campo da chave do registro e o tipo de dado dessa chave. Já os campos "field-name" e "field-type" representam o nome do campo e seu tipo de dado para os demais campos do registro. Pense nesse arquivo como a estrutura da tabela de um banco de dados ou de uma planilha eletrônica(Excel, Calc).

Observe que os campos do tipo "char" conterão a quantidade máxima de bytes representados dentro dos colchetes. Para os outros tipos de dados não será fornecido nenhum tamanho em bytes, você deve ser capaz de administrar os tamanhos para cada tipo passado. Não haverá "pegadinhas" quanto aos tipos, eles estarão restritos aos: char, double, float e int. O exemplo é ilustrativo, os casos de teste irão validar a flexibilidade do seu código em adaptar-se aos campos.

Após o nome do arquivo de metadados seu programa poderá receber os seguintes comandos:

1) **insert:** utilizado para inserir dados no arquivo de registros passado como primeiro parâmetro no arquivo de metadados. As linhas a seguir exemplificam o comando:

```
insert 4, "Joao Mario", 15, 324.54
insert 1, "Pedro Carlos", 21, 56.34
insert 2, "Marcelo Isidoro", 34, 673.23
```

Vejam que não há acentos nos caracteres :)
Nesse exemplo, 3 registros são inseridos no final do arquivo, utilizando os dados descritos. Assuma inicialmente que os comandos insert estão sempre corretos, ou seja, que os campos e seus respectivos tipos estão corretamente descritos.

2) **index:** esse comando deve gerar um arquivo com o mesmo nome do arquivo de dados, ou seja, neste caso registros.reg contudo com outra extensão. Será usada a extensão idx. Sendo assim você deve criar um arquivo binário chamado registros.idx que conterá, para cada registro no arquivo registros.reg:

- os valores do campo chave (key) do arquivo registros.reg, e
- o offset (deslocamento em bytes) do registro nesse arquivo.

Neste exemplo, como cada registro é formado por um int, char[80], int e double, você terá registros com 96 bytes. Sendo assim, o primeiro registro inserido, i.e., o de chave 4 estará no offset 0, o registro de chave 1 estará no offset 96 e o registro de chave 2 estará no offset 192. Dessa maneira, temos inicialmente os seguintes pares <<chave>> e <<offset>>:

<<chave>>	<<offset>>
4	0
1	96
2	192

Note que os registros estão ordenados pelo offset, pois essa é a ordem original do arquivo de dados. O programa deverá ordenar esses dados, de maneira crescente, de acordo com as chaves, obtendo:

<<chave>>	<<offset>
1	96
2	192
4	0

Este conteúdo deverá ser gravado no arquivo registros.idx. OBS: o cabeçalho indicado no exemplo não existe no arquivo. Ele é usado aqui apenas para demonstrar o que significa cada coluna.

3) **search:** comando que realiza uma busca binária utilizando o arquivo registros.idx. Devido a isso a busca é sempre feita pela chave. Ao encontrar a chave no arquivo de índice, deve-se obter o offset do registro correspondente, abrir o arquivo registros.reg e imprimir seus dados. Por exemplo, ao receber:

search 2

Você irá carregar o arquivo registros.idx para a memória RAM e procurar pela chave 2, utilizando busca binária. Ao encontrá-la você utilizará o offset, que no exemplo é o 192 para localizar o registro contido em registros.reg. Em seguida deve abrir o arquivo registros.reg e carregar o registro relativo ao offset 192 e imprimí-lo como segue:

codigo: 2\n
nome: Marcelo Isidro\n
idade: 34\n
bolsa: 673.23\n\n

Em caso de busca sem sucesso não imprimir nada na tela. Note que não será possível fazer a busca sem a criação do índice antes. Caso uma busca seja requerida e não exista um índice, você deve automaticamente chamar o comando index para gerar o item. Lembre-se há dois pulos de linha no último campo.

4) **exit:** Este comando indica que nenhum outro comando adicional será recebido. Assim seu programa poderá finalizar.

2. Instruções complementares

Para a solução do problema é obrigatória a utilização de **Funções, Alocação dinâmica e Structs.** Desaloque a memória heap ao finalizar o programa. Arquivos Makefile serão aceitos para aqueles que souberem usar.

FAQ

1) Quais os tipos serão definidos no arquivo de metadados?

R: int, double, char, char[número] e float

2) Qual comando da linguagem C posso utilizar para saber qual o offset atual de um arquivo?

R: ftell

3) Qual comando da linguagem C posso utilizar para posicionar em um offset do arquivo?

R: fseek

4) O arquivo de metadados poderá ter qualquer formato de registro desde que tenha um campo chave?

R: Sim.

5) Mas como faço para criar dinamicamente um registro que represente o conteúdo do arquivo de metadados?

R: Isto é parte do trabalho. Pense um pouco.

Motivação: Este trabalho tem grande relação com as disciplinas de Organização de Arquivos e Banco de Dados que serão vistas nos próximos semestres. Ao implementá-lo, você estará melhor preparado para os próximos semestres.

3. Informações de entrada e saída

Seu programa receberá as entradas:

Entrada:

registros.dat
insert 4, "Joao Mario", 15, 324.54
insert 1, "Pedro Carlos", 21, 56.34
insert 2, "Marcelo Isidoro", 34, 673.23
search 2
index
exit

Saída:
codigo: 2
nome: Marcelo Isidoro
idade: 34
bolsa: 673.23