Universidade Federal da Bahia
Departamento de Ciência da Computação
MATA54 - Estruturas de Dados e Algoritmos II

Relatório do Primeiro Trabalho Prático

Docente:
Flávio Assis

Discentes:
Eric dos Santos Sampaio
Gregory Silva Gley Santos
Lucas Natanael Brito Prates

1. Introdução

O trabalho prático proposto tem como objetivo familiarizar os alunos com a manipulação de arquivos binários (inserção, remoção, consulta e impressão). Para tal, será criada uma aplicação escrita em alguma linguagem de programação permitida pelo professor. No caso da nossa equipe, utilizamos a linguagem de programação C. O código fonte da aplicação acompanha este relatório, com o nome “trabalho1_final.c”.

2. Funcionamento

O programa armazena na memória secundária um vetor estático de registros. O tamanho desse vetor é dado pela constante *TAMANHO_ARQUIVOMAX,* que pode ser alterado no código fonte conforme necessário. Os registros são do tipo *struct Registro*, que contém: chave primária (tipo Inteiro), idade (tipo Inteiro), nome (vetor de caracteres) e um flag de controle (tipo Inteiro). Esta flag de controle pode assumir os valores:

- 0 - quando o registro nunca foi gravado (valor inicial);
- 1 - quando há um registro gravado;
- 2 - quando o registro foi apagado.

Para a manipulação desses registros no arquivo, escrevemos as funções que serão descritas a seguir:

***int* verificarArquivo()**

Esta função abre “arquivo.bin” para leitura. No caso do arquivo não existir, ela retorna o valor inteiro *0*. Caso o arquivo exista, retorna o valor inteiro *1*.

***void* criarArquivo(Registro reg[TAMANHO_ARQUIVOMAX])**

Esta função chama *verificarArquivo()* para verificar se “arquivo.bin” existe no diretório. Caso o retorno seja *0,* ela cria o denominado arquivo, e grava nele o vetor de registros passado por parâmetro. Este vetor será previamente inicializado com valor de flag=0 para todas as suas posições, no momento da execução do programa.

int checkIfKeyExists(int key, FILE *file)

Esta função percorre sequencialmente o arquivo passado por parâmetro em busca da chave também passada por parâmetro. Ela retorna o valor inteiro *1* caso encontre um registro com o valor de chave igual ao valor procurado, porém apenas se o valor de flag desse registro for igual a *1.* Caso as condições anteriores não sejam satisfeitas, retorna o valor inteiro *0.*

void insereRegistro(Registro reg)

Esta função chama *checkIfKeyExists()* para verificar se o valor de chave do registro passado por parâmetro já se encontra no arquivo. Caso o retorno seja 0, percorre sequencialmente o arquivo a partir do início até que encontre uma posição com valor de flag 0 ou 2. Caso encontre, insere o registro nesta posição.

void consultaRegistro(int chave)

Esta função percorre sequencialmente o arquivo a procura de um registro com o valor de chave igual ao valor passado por parâmetro, e com valor de flag = 1. Caso encontre, imprime no console os dados desse registro, conforme a especificação do trabalho. Caso não encontre, imprime no console uma mensagem informativa.

void removeRegistro(int chave)

Esta função percorre sequencialmente o arquivo a procura de um registro com o valor de chave igual ao valor passado por parâmetro, e com valor de flag = 1. Caso encontre, o valor de flag desse registro é modificado para 2, e esse registro passa a ser visto como vazio para inserção e consulta. Caso não encontre, imprime no console uma mensagem informativa.

void printAll()

Está função imprime na tela todas as posições do vetor gravado no arquivo. A saída será no formato: Posição do vetor - chave - nome - idade.

Exemplo:

Posição: 1 chave: 10 João 31

Caso uma posição esteja vazia (valor de flag 0 ou 2), ela imprimirá, por exemplo:

Posição: 2 vazia

3. Considerações

Após as descrições dos algoritmos que utilizamos nesse programa, é importante fazer algumas considerações:

1. Os registros ficam organizados no arquivo por ordem de inserção. Conforme descrito na função de inserção, um novo registro é inserido na primeira posição vazia que encontrar.
2. Quando um registro é removido, não é feita uma reorganização do arquivo. A posição é dada como vazia e os outros registros permanecem nas mesmas posições.
3. O atributo flag é importante para que um registro que foi apagado não seja dado como presente no arquivo quando é feita uma consulta ou inserção com o mesmo valor de chave, já que o atributo chave do registro apagado não é modificado.

4. Conclusão

Nessa atividade quisemos focar em nos familiarizar melhor com manipulação de arquivos, visto que é algo que ainda não tínhamos muito contato no decorrer do curso. Para tanto, a questão do desempenho foi deixada um pouco de lado, visto que aplicamos métodos sequenciais para todas as funções, o que pode ocasionar um elevado número de acessos ao disco. Esperamos que, com essa base mais solidificada e com o conhecimento teórico que estamos adquirindo nas aulas, possamos focar mais na questão do desempenho nas próximas atividades.