Universidade Federal da Bahia
Graduação em Ciência da Computação
MATA54 - Estruturas de Dados e Algoritmos II
Primeiro Trabalho Prático
Prof. Flávio Assis
Semestre 2016.1 - 23 de agosto de 2016

Hashing Duplo

1 Descrição Geral do Trabalho

Neste trabalho o aluno implementará um arquivo estruturado como uma tabela *hash* em que o método de resolução de colisão utilizado é o *Hashing* Duplo (*Double Hashing*).

Os dados do usuário a serem armazenados no arquivo são: uma *chave*, de valor inteiro não negativo; uma cadeia de, no máximo, 20 caracteres, que irá armazenar um *nome*; e um outro valor inteiro não negativo, que irá armazenar uma *idade*. O programa deverá conter uma constante definida com o seguinte identificador:

TAMANHO_ARQUIVO: indica o número máximo de registros do arquivo.

O valor inicial da constante TAMANHO_ARQUIVO deve ser 11. O programa deve ser feito de forma que o valor desta constante possa ser modificado.

As funções de *hashing* a serem utilizadas, denominadas h_1 e h_2, são:

$$h_1(chave) = chave \ mod \ \text{TAMANHO\_ARQUIVO}$$
$$h_2(chave) = \max\{\lfloor chave/TAMANHO\_ARQUIVO \rfloor, 1\}$$

Observações importantes: O programa deve manter as atualizações em arquivo. A correção levará em consideração que o estado dos dados é persistente. Com isto, um teste pode ser feito, por exemplo, inserindo-se um registro, terminando a execução do programa e fazendo uma consulta ao registro em nova invocação do programa. Neste caso o registro deve ainda estar no arquivo.

Adicionalmente, lembre-se de que é assumido que a memória principal é insuficiente para armazenar todos os dados. Portanto, uma implementação que mantém a estrutura do arquivo em memória (em um vetor, por exemplo) e o salva por completo no arquivo será considerada inaceitável.

O arquivo deve ser armazenado em formato binário.

2 Formato de Entrada e Saída

A entrada conterá uma sequência de operações sobre o arquivo. As operações e seus formatos estão descritos abaixo:

1. **insere registro**: esta operação conterá quatro linhas. A primeira linha conterá a letra 'i'. A segunda conterá um valor de chave. A terceira conterá uma sequência de até 20 caracteres, que corresponderá ao campo *nome*. A quarta linha conterá um valor de idade. A sequência de caracteres da terceira linha conterá qualquer sequência de letras (minúsculas, sem acento, nem cedilha) e espaços, sendo que o primeiro e último caracteres não serão espaço.

   Esta operação verifica se já há registro no arquivo com o valor de chave indicado. Se sim, esta operação gera na saída, em uma mesma linha, a sequência de caracteres *'chave ja existente:'*, seguida de um espaço, seguido do valor da chave. Se a chave não existir, a operação insere o registro no arquivo, sem gerar saída.

2. **consulta registro**: esta operação conterá duas linhas. A primeira linha conterá a letra 'c'. A segunda conterá um valor de chave.

   Se houver registro no arquivo com o valor de chave indicado, esta operação gera na saída a sequência de caracteres *'chave:'*, seguida de um espaço, seguido do valor da chave. Em seguida, na próxima linha escreve o valor do nome associado ao registro, e, na linha seguinte, o valor da idade associada ao registro. Se não houver registro no arquivo com o valor de chave indicado, esta operação gera na saída a sequência de caracteres *'chave nao encontrada:'*, seguida de um espaço, seguido do valor da chave.

3. **remove registro**: esta operação conterá duas linhas. A primeira linha conterá a letra 'r'. A segunda conterá um valor de chave.

   Se houver registro no arquivo com o valor de chave indicado, esta operação causará a remoção do registro e não gerará saída. Se não houver registro no arquivo com o valor de chave indicado, esta operação gera na saída a sequência de caracteres *'chave nao encontrada:'*, seguida de um espaço, seguido do valor da chave.

4. **imprime arquivo**: esta operação conterá apenas uma linha, contendo a letra 'p'. Esta operação imprimirá o arquivo, da forma a seguir. Os registros serão apresentados, um em cada linha, em ordem, do registro de índice 0 até o registro de índice $TAMANHO\_ARQUIVO-1$. Cada linha conterá: o índice do registro, seguido de dois pontos (':'), seguido de um espaço. Se o registro estiver vazio, a sequência de caracteres *'vazio'* deverá ser apresentada. Se o registro contiver dados, deve ser apresentada a chave do registro, seguida de um espaço, seguida da sequência de caracteres (nome), seguida de um espaço, seguido da idade.

5. **média de acessos a registros do arquivo**: esta operação conterá apenas uma linha, contendo a letra 'm'. Esta operação apresenta, em uma linha, apenas o valor da média do número de acessos a registros do arquivo, considerando-se uma consulta a cada um dos registros armazenados no arquivo. Esta média deve ser apresentada sempre como um valor real, com uma única casa decimal.

6. **término da sequência de comandos:** a sequência de comandos será terminada por uma linha com a letra 'e'.

Importante: o programa não deve gerar nenhum caractere a mais na saída, além dos indicados acima. Em particular, o programa não deve conter menus.

Não deve haver espaço entre linhas na saída. A saída deve apresentar os caracteres em letras minúsculas.

3 Observações

Trabalho individual.

Data de entrega: 21/09/2016

Linguagens de programação permitidas: C, C++, Java ou Python.

Observação Importante: Para as linguagens C, C++ e Java, somente trabalhos feitos utilizando-se os seguintes compiladores serão aceitos:

- C: gcc ou djgpp
- C++: g++ ou djgpp
- Java: compilador java recente, disponibilizado pela Oracle.

Não serão compilados trabalhos em outros compiladores! Erros ocasionados por uso de diferentes compiladores serão considerados erros do trabalho!

O aluno deverá armazenar submeter seu trabalho através do *moodle*.