

## Exercício

O objetivo do exercício é gerenciar um sistema de histórico escolar “limpo” de uma dada instituição. O sistema armazena as seguintes informações: ID do aluno, sigla da disciplina, nome do aluno, nome da disciplina, média e frequência. A chave primária é composta pela composição “ID do aluno+Sigla da disciplina”. O arquivo a ser criado deve ser de registros e campos de tamanho variável, com um inteiro (4 bytes) no início do registro indicando o tamanho do registro, e com campos separados pelo caractere ‘#’.

ID do aluno	Sigla da disciplina	Nome do aluno	Nome da disciplina	Média	Frequência
3 caracteres (fixo)	3 caracteres (fixo)	50 caracteres (máximo)	50 caracteres (máximo)	float (a ser bufferizado)	float (a ser bufferizado)

Ex.: 53001#ED2#Paulo da Silva#Estruturas de Dados 2#7.3#75.4

O programa conterá as seguintes opções:

- Inserir
- Remover
- Buscar

### Inserir (a)

A cada inserção o “ID do aluno+Sigla da disciplina” deverá ser adicionado ao índice primário estruturado como uma hash. Portanto, a cada nova inserção as seguintes mensagens deverão ser mostradas (note que mais de uma pode aparecer):

- “Endereço X”, endereço X gerado para a chave fornecida;
- “Chave X inserida com sucesso” deve ser impressa ao final da inserção indicando sucesso da operação;
- “Colisão”, sempre que um *home address* não estiver livre, gerando uma colisão;
- “Tentativa X”, X é o número da tentativa para tratar a colisão.

### Exemplo de Inserção

1234567891234

Endereço 5

Chave 1234567891234 inserida com sucesso

4321987654321

Endereço 5

Colisão

Tentativa 1

Chave 4321987654321 inserida com sucesso

Observações:

- Para tratar as colisões utilize Overflow Progressivo;
- Considere uma hash de 13 posições;
- Utilize o Método da Divisão Inteira para encontrar o endereçamento de uma dada chave (função hash);
- Cada endereçamento contém duas chaves (está sendo utilizado o conceito de Bucket);
- Lembre-se que o arquivo hash é um arquivo de registros fixos que contém, no mínimo, duas informações: Chave + RRN. Campos adicionais podem ser acrescentados se necessário.

### Remover (b)

Dado um “ID do aluno+Sigla da disciplina” o programa remove o respectivo registro do índice hash. Não é necessário realizar a remoção no arquivo principal (o que contém os registros, somente no índice). Para tanto, utilize o processo de remoção associado ao Overflow Progressivo.

### Buscar (c)

Dado um “ID do aluno+Sigla da disciplina” o programa retorna os dados do respectivo registro. Para tanto, a busca deve ser feita na hash. Além disso, as seguintes mensagens deverão ser exibidas em relação à busca:

- “Chave X encontrada, endereço E, N acessos” indica que a chave X foi encontrada no endereço E e que foram necessários N acessos para recuperar a informação na hash. Após a exibição dessa mensagem, os dados referentes ao registro deverão ser recuperados do arquivo principal;
- “Chave X não encontrada” indica que a Chave X não está presente na hash e, consequente, no arquivo principal.

#### Exemplo Pesquisa

1234567891234

*Chave 1234567891234 encontrada, endereço 5, 1 acesso*

4321987654321

*Chave 4321987654321 encontrada, endereço 6, 2 acessos*

1234567899999

*Chave 1234567899999 não encontrada*

### OBSERVAÇÃO

A fim de facilitar os testes, serão fornecidos três arquivos: (a) “insere.bin”, (b) “busca.bin” e “remove.bin”. O primeiro (a) conterá os dados a serem inseridos durante os testes (não necessariamente todos os dados serão inseridos). Para tanto, uma sugestão é carregar o arquivo em memória (um vetor de struct) e ir acessando cada posição conforme as inserções vão ocorrendo. Note que é possível encerrar a execução e recomendar a execução, sendo necessário marcar, de algum modo, quantos registros já foram utilizados do mesmo.

Em relação a (b), o arquivo conterá uma lista de “ID do aluno+Sigla da disciplina” a serem utilizados durante a pesquisa (opção (c)). A ideia é a mesma já descrita, ou seja, carregar o arquivo em memória (um vetor de struct) e ir acessando cada posição conforme as buscas vão ocorrendo. Note que é possível encerrar a execução e recomendar a execução, sendo necessário marcar, de algum modo, quantos registros já foram utilizados do mesmo. Em relação a (c), o arquivo conterá uma lista de “ID do aluno+Sigla da disciplina” a serem utilizados durante a remoção (opção (b)). Idem aos demais comentários em relação ao arquivo “busca.bin”.

### Observações

TODOS OS ARQUIVOS DEVERÃO SER MANIPULADOS EM MEMÓRIA SECUNDÁRIA.

Não criar os arquivos toda vez que o programa for aberto (fazer verificação).