Exercício

O objetivo do exercício é gerenciar um sistema de cadastro geral de todas as seguradoras do país. Para tanto, as seguintes informações são armazenadas: código do segurado (chave primária), nome do segurado, nome da seguradora, tipo do seguro contratado. O objetivo é levantar dados sobre o setor. O arquivo a ser criado deve ser de registros fixos com campos fixos (fixo-fixo).

Código do Segurado	Nome do Segurado	Seguradora	Tipo do Seguro
int	50 caracteres	50 caracteres	30 caracteres

O programa conterá as seguintes opções:

a. Inserir
b. Listar os dados de todos os segurados
c. Listar os dados de um segurado específico
d. Carrega Arquivos (opcional)
e. Remoção (opcional) (ponto extra) (essa opção só será considerada se as demais funcionarem. Será necessária a explicação do código)

Inserção (a)

A cada novo cadastro o Código do Segurado deverá ser adicionado ao índice primário estruturado como uma árvore-B. Portanto, a cada nova inserção as seguintes mensagens deverão ser mostradas (note que mais de uma pode aparecer):

- "Divisão de nó" deve ser impressa sempre que um nó for dividido;
- "Chave X promovida" deve ser impressa sempre que uma chave for promovida. X é o valor da chave promovida;
- "Chave X inserida com sucesso" deve ser impressa ao final da inserção indicando sucesso da operação;
- "Chave X duplicada" deve ser impressa ao final da inserção e indica que a operação de inserção não foi realizada.

Observação: antes de inserir um registro no arquivo principal certifique-se de que a chave não existe no índice.

Exemplo de Inserção
C
Chave C inserida com sucesso
S
Chave S inserida com sucesso
D
Chave D inserida com sucesso
T
Divisão de nó
Chave S promovida
S
Chave S duplicada

Listar os dados de todos os segurados (b)

Nessa opção o índice árvore-B deverá ser percorrido em-ordem e a cada Código encontrado listar os dados associados ao mesmo. Desse modo, essa opção deverá imprimir os dados de todos os segurados cadastrados por ordem de Código.

Listar os dados de um segurado específico (c)

Dado um Código o programa retorna os dados do respectivo segurado. Para tanto, a busca deve ser feita na árvore-B. Além disso, as seguintes mensagens deverão ser exibidas em relação à busca na árvore:

- “Chave X encontrada, página Y, posição Z” indica que a Chave X foi encontrada e encontra-se na página Y, na posição Z da página. Após a exibição dessa mensagem, os dados referentes ao segurado deverão ser recuperados do arquivo principal;
- “Chave X não encontrada” indica que a Chave X não está presente na árvore-B e, consequente, no arquivo principal.

Exemplo Pesquisa
C
Chave C encontrada, página 0, posição 0
Z
Chave Z não encontrada

Carrega Arquivos (d)

A fim de facilitar os testes, serão fornecidos dois arquivos: (a) “insere.bin”, (b) “busca.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 recomeçar a execução, sendo necessário marcar, de algum modo, quantos registros já forma utilizados do mesmo.

Em relação a (b), o arquivo conterá uma lista de “Códigos” 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 recomeçar a execução, sendo necessário marcar, de algum modo, quantos registros já forma utilizados do mesmo.

Remoção (e)

Essa opção é opcional e vale um ponto extra na média. A cada Código deve-se fazer a remoção do mesmo no índice primário estruturado como uma árvore-B (não precisa remover do arquivo principal). Para tanto, utilizem as regras de remoção discutidas em sala de aula. A cada remoção as seguintes mensagens deverão ser mostradas (note que mais de uma pode aparecer):

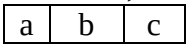
- “Caso 1: remoção simples” se cair no Caso 1 discutido em sala;
- “Caso 2: troca com o sucessor imediato” se cair no Caso 2 discutido em sala;
- “Caso 3: redistribuição” se cair no Caso 3 discutido em sala;
- “Caso 4: concatenação” se cair no Caso 4 discutido em sala.

Observação: a implementação deve ser genérica, independente da ordem da árvore. Antes de remover um registro certifique-se de que a chave existe no índice. Caso você opte por implementar essa opção, um arquivo adicional aos listados em (e) será fornecido, “remove.bin”, o qual conterá uma lista de “Códigos” a serem removidos. As mesmas observações feitas para os arquivos descritos em (d) valem aqui.

Observações

TODOS OS ARQUIVOS DEVERÃO SER MANIPULADOS EM MEMÓRIA SECUNDÁRIA.

Não criar os arquivos toda vez que o programa for aberto (fazer verificação). O seu programa deve realizar as operações sobre uma árvore-B de ordem 4 (ou seja, no máximo 3 chaves). Para padronizar, sempre promovam, quando houver overflow, a chave de índice 2, começando em zero.



Inserção de d causa overflow

Promova c (índice 2)

Para auxiliar o desenvolvimento do trabalho é fornecido um código que insere chaves em uma árvore-B de ordem 5. Vocês devem utilizar esse código como base. Entretanto, algumas alterações serão necessárias para que o mesmo funcione corretamente. Vocês deverão estudar e entender o código para que consigam fazer as alterações necessárias. No caso do procedimento de pesquisa básica, tome como base o pseudocódigo

discutido em sala de aula. Em relação ao procedimento de percurso em-ordem, o mesmo deverá ser desenvolvido por vocês.