

SCE-283 Linguagens de Programação e Aplicações

Professor responsável: *Fernando Santos Osório*

Semestre: 2008/2

Horário: Segunda 14h20

E-mail: fosorio .at. icmc.usp.br

fosorio .at. gmail.com

Web: [http://www.icmc.usp.br/~fosorio /](http://www.icmc.usp.br/~fosorio/)

TRABALHO PRÁTICO Nro. 02 (Atualizado em 29/10/2008)

[DEFINIÇÃO GERAL – Matriz Esparsa]

Escreva um programa, conforme definido a seguir, na Linguagem “C” usando um compilador compatível com o GNU GCC (e.g. GCC, DJGPP, DEV-C/C++, CodeBlocks). Este programa deve implementar uma **matriz esparsa usando uma LISTA ENCADEADA** (lista com ponteiros), ou seja, deve possibilitar que o usuário gerencie a alocação dinâmica na memória de uma matriz de dados que é de um tamanho bastante superior à quantidade de dados que ela realmente possui armazenada. O programa implementado deve seguir a especificação dada a seguir:

- Implemente a matriz esparsa usando uma lista encadeada com ponteiros;
- As operações definidas sobre esta matriz esparsa devem permitir:
 - + **Inserção** de um dado na matriz (insere);
 - + **Busca** e exibir um dado da matriz (busca/exibe);
 - + **Remoção** de um dado na matriz (remove);
 - + **Apagar** todos os dados da matriz (apagar);
 - + **Salvar** em disco todos os dados da matriz (salva);
 - + **Ler** do disco os dados da matriz (ler);
- Cada uma das operações listadas acima deve ser implementada em uma rotina independente, por exemplo, você deve implementar uma rotina do tipo *insere_dado* que deverá ser capaz de dada uma lista e uma informação (dados), criar um novo nodo, inserir os dados neste nodo, e incluir ele na lista encadeada;

Detalhamento sobre algumas decisões sobre a implementação a ser realizada:

Inserção:

- O modo de inserção poder de livre escolha, ou seja, inserção no início da lista, inserção no final da lista ou inserção ordenada. O modo de inserção é livre, desde que as demais rotinas sejam capazes de realizar as tarefas que cabe a cada uma delas (exemplo: busca, remoção, etc);
- Outra opção a ser feita é em relação a inserção, diz respeito aos dados duplicados na lista: se você aceitar a inserção duplicada, ao buscar/exibir/remover devem ser tratadas as múltiplas ocorrências de um mesmo dado; ou se você não aceitar dados duplicados, deve ser apresentada uma mensagem ao usuário no momento da inserção de dados, indicando que não foi possível inserir o dado pois este já existe na matriz;
- As informações armazenadas na matriz esparsa são compostas de basicamente 2 informações distintas: um conjunto de índices, que definem sua posição na matriz “virtual”, e o dados propriamente ditos. Por exemplo: em uma agenda, os índices indicam a hora, dia, mês e talvez até o ano de um compromisso (posição na matriz) associados a descrição do compromisso, ou seja, podemos ter um compromisso no dentista as 14h do dia 01/01/2009;

Remoção:

- A remoção poderá ser implementada de uma das seguintes formas (sua opção): remoção física, onde o nodo é excluído realmente da lista encadeada, ou remoção lógica, onde o dado é marcado como excluído. No caso da remoção lógica, nodos removidos através de uma marca lógica não poderão aparecer como presentes na lista, em qualquer das outras opções;
- Faz parte do trabalho buscar algoritmos que permitam implementar a remoção, pois sabidamente só foi apresentado o algoritmo de inclusão em listas encadeadas. Encontrar na literatura e/ou implementar um algoritmo próprio de remoção física ou remoção lógica é parte do trabalho que está sendo proposto;
- Note que a remoção precisa localizar (busca) previamente qual dado será removido, onde esta questão está sendo abordada mais abaixo, na apresentação da rotina busca;

Interface:

Exemplo da interface de entrada e saída do programa:

```
>> Matriz Esparsa <<

Menu de Opcoes:
1 - Inicializar ou Apagar toda a lista em memoria
2 - Inserir dados na matriz (lista encadeada)
3 - Consultar dados da matriz (buscar e exibir) - Tipo1
4 - Consultar dados da matriz (buscar e exibir) - Tipo2
5 - Remover um dado da matriz
6 - Salvar em disco
7 - Ler do disco
Entre com sua opcao: 1

Confirma que todos os dados em memoria serao apagados? (s/n)
s
>> Lista Inicializada

Tecla [enter] para voltar ao menu...
```

Dados Armazenados na Lista:

Estão sendo propostas 2 aplicações de matrizes esparsas (ver Aplicação 01 e Aplicação 02 abaixo), onde você poderá escolher qual das duas você irá implementar. Ambas as propostas são similares em termos de funcionalidades em relação a matriz esparsa (implementam inserção, remoção, salvar, ler... conforme descrito acima), onde o que vai mudar é o tipo de dados que serão armazenados na matriz.

Busca/Exibição:

- A busca de dados na matriz será sempre feita baseando-se nos índices que definem onde está armazenado o dado na matriz. Conforme o exemplo anterior, se tenho uma agenda de compromissos, a busca de dados deve considerar a horas e data do compromisso (índices) de modo a recuperar os detalhes do compromisso (exemplo: consulta no dentista);

Busca/Exibição – Usada na Remoção:

- A remoção irá fazer uso de uma rotina busca (a própria rotina, ou uma adaptação desta), pois é necessário buscar qual o dado se deseja remover. A remoção, assim como a busca, é baseada nos índices do dado a ser removido. O usuário informa qual é o índice do dado, a rotina verifica se este dado existe ou não na matriz, e caso encontre o dado a ser removido, é exibido ao usuário o conteúdo do nodo e solicitada uma confirmação da remoção do nodo;
- A remoção precisará encontrar o nodo a ser removido (ponteiro para este nodo), mas também o nodo anterior ao nodo a ser removido (ponteiro para o nodo anterior), pois ao remover é necessário ajustar o ponteiro do “próximo” do nodo anterior para o “próximo” do nodo removido.

[Aplicação 01]

O aluno que optar pela Aplicação 01 irá implementar uma AGENDA DE COMPROMISSOS, com as seguintes funcionalidades e respeitando as seguintes especificações:

- Agenda indexada pela: Hora [0 a 23], Dia [1 a 31], Mês [1 a 12] e Ano [2008 a 2012] – Matriz real de $24 \times 31 \times 12 \times 5 = 44.640$ compromissos podem ser armazenados;
- Agenda comporta compromissos que são descritos por um texto (até 80 caracteres por compromisso) – Matriz completa ocuparia: $44.640 \times 80 = 3.571.200$ bytes!!

Inserção: é solicitado ao usuário a hora, o dia, o mês e o ano do compromisso, seguido do texto que descreve o compromisso. Estes dados devem ser armazenados na estrutura encadeada. Você deve decidir se permitirá ou não a existência de mais de um mesmo compromisso para a mesma data e horário.

Busca_Tipo1: é solicitado ao usuário a hora, o dia, o mês e o ano do compromisso, sendo exibido(s) o(s) compromisso(s) cadastrados na lista encadeada para aquela data e horário. Caso não existam compromissos cadastrados deverá ser apresentada uma mensagem ao usuário do tipo: “agenda sem compromissos nesta data e horário”.

Busca_Tipo2: é solicitado ao usuário o dia, o mês e o ano do compromisso (sem a hora!), sendo exibidos todos os compromissos cadastrados na lista encadeada para aquela data (listar os compromissos daquele dia). Se não existirem compromissos cadastrados nesta data, exibir a mensagem “agenda com dia sem compromissos”.

Remoção: é solicitado ao usuário a hora, o dia, o mês e o ano do compromisso, sendo removido(s) o(s) compromisso(s) cadastrados na lista encadeada para aquela data e horário. A remoção deve: (1) buscar o compromisso na lista; (2) exibir na tela os dados do compromisso; (3) solicitar ao usuário para confirmar a remoção do compromisso da agenda. Se não existirem compromissos cadastrados nesta data e horário, exibir a mensagem “agenda sem compromissos nesta data e horário”.

Salvar: é solicitado ao usuário o nome do arquivo onde ele deseja salvar os dados (exemplo: agenda.txt), e devem ser gravados todos os dados (compromissos cadastrados na agenda) em um arquivo do tipo texto.

Ler: é solicitado ao usuário o nome do arquivo de onde ele deseja ler os dados (exemplo: agenda.txt), e devem ser lidos os dados (compromissos cadastrados no arquivo da agenda) de um arquivo do tipo texto, sendo inseridos na lista encadeada em memória. Note que os arquivos gravados e lidos devem possuir um formato compatível que sirva tanto para a gravação em disco, como para a leitura do disco (se eu salvei os dados em disco, devo poder ler este arquivo salvo previamente e recuperar os dados salvos).

[Aplicação 03]

O aluno que optar pela Aplicação 02 irá implementar uma MODELAGEM TRIDIMENSIONAL POR DEFINIÇÃO DE PONTOS, com as seguintes funcionalidades e respeitando as seguintes especificações:

- Agenda indexada pela coordenada: X [0 a 999], Y [0 a 999], Z [0 a 999], Ângulo – Matriz real de $1.000 \times 1.000 \times 1.000 = 1.000.000.000$;
- Agenda comporta dados armazenados em cada coordenada, sendo descritos por duas variáveis do tipo inteiro: cor do objeto e tamanho do objeto posicionado naquele ponto (XYZ) – Matriz completa ocuparia $1.000.000.000 * 2 * \text{sizeof}(\text{int})!!$

Inserção: é solicitada ao usuário a coordenada X, Y e Z do objeto, seguido de dois valores numéricos, a saber, a cor e o tamanho deste objeto. Estes dados devem ser armazenados na estrutura encadeada, considerando que o índice de acesso ao dado é dado pela sua coordenada XYZ.

Busca_Tipo1: é solicitada ao usuário a coordenada X, Y e Z do objeto, sendo exibido(s) o(s) objeto(s) cadastrado(s) na lista encadeada para aquele ponto XYZ. Caso não existam objetos cadastrados deverá ser apresentada uma mensagem ao usuário do tipo: “coordenada XYZ sem objeto inserido”.

Busca_Tipo2: é solicitada ao usuário a cor de um determinado objeto (sem a coordenada XYZ), sendo exibidos todos os dados dos objetos cadastrados na lista encadeada que possuem definida aquela cor (listar os objetos da cor indicada pelo usuário, informando coordenada XYZ, cor e tamanho). Se não existirem objetos cadastrados desta cor, exibir a mensagem “cor de objeto inexistente no cadastro”.

Remoção: é solicitada ao usuário a coordenada X, Y e Z do objeto, sendo removido(s) o(s) objeto(s) cadastrado(s) na lista encadeada para aquele ponto XYZ. A remoção deve: (1) buscar o ponto XYZ na lista; (2) exibir na tela os dados associados ao ponto XYZ; (3) solicitar ao usuário para confirmar a remoção do objeto da lista. Se não existirem objetos cadastrados nesta coordenada XYZ, exibir a mensagem “coordenada XYZ sem objeto inserido”.

Salvar: é solicitado ao usuário o nome do arquivo onde ele deseja salvar os dados (exemplo: modelo3d.txt), e devem ser gravados todos os dados (informações dos objetos cadastrados: coordenada XYZ, cor e tamanho) em um arquivo do tipo texto.

Ler: é solicitado ao usuário o nome do arquivo de onde ele deseja ler os dados (exemplo: modelo3d.txt), e devem ser lidos os dados (informações dos objetos cadastrados: coordenada XYZ, cor e tamanho) de um arquivo do tipo texto, sendo inseridos na lista encadeada em memória. Note que os arquivos gravados e lidos devem possuir um formato compatível que sirva tanto para a gravação em disco, como para a leitura do disco (se eu salvei os dados em disco, devo poder ler este arquivo salvo previamente e recuperar os dados salvos).

ENTREGA DO TRABALHO:

* Envie um e-mail com o(s) programa(s) fonte anexado(s) (somente os arquivos .c) ao prof. Osório.

E-MAIL TO: **work2usp@yahoo.com** (Enviar o original para este email)
EMAIL CC: **fosorio@gmail.com** (Enviar com cópia para este email)
SUBJECT: **[SCE283] TP02 <nome_aluno>** (Assunto do email)

* Escreva no corpo da mensagem de e-mail:

NOME: <seu nome> + <nome dos componentes do grupo> (Max. 3 Alunos)
AMBIENTE USADO: <nome e versão do ambiente de compilação usado>
APLICAÇÃO: <Tipo 01 ou Tipo 02>

* Exemplo:

Nome: Fernando Osório + Fulano da Silva
Ambiente usado: DJGPP com Rhide – GCC versão: 4.2.3
(para saber a versão no DOS use o comando “gcc --version”)
Ambiente usado: Dev-C++ 5.0 beta 9.2 (4.9.9.2)
(versão conforme indicado no site de onde você baixou, ou no próprio software menu “help-about”)

Se você for enviar o executável (.exe), o arquivo terá obrigatoriamente que estar compactado em formato **.rar** ou **.bz2** (OUTROS FORMATOS NÃO SERÃO ACEITOS, POIS SÃO RECUSADOS PELO SERVIDOR DE E-MAIL).

* Entregar até a data indicada no Site do CoTeia

<http://coteia.icmc.usp.br/mostra.php?ident=527> (ver em Trabalhos Práticos)

===== THAT'S ALL FOLKS !!! =====