Exercícios 3 – Listas Estáticas, Encadeadas, Pilhas, Filas

OBS: Nos problemas a seguir, é interessante a construção de uma biblioteca em C para implementar o Tipo Abstrato de Dados (TAD) Lista Linear e suas operações clássicas.

PARTE 1 – Listas Lineares com Alocação Estática

1- Implemente um algoritmo que retira um elemento da posição pos_1 e coloca na pos_2. Use os algoritmos de inserção e remoção definidos anteriormente (chame as funções implementadas no TAD Lista).

2- Os sistemas operacionais e outros aplicativos normalmente registram em arquivo as atividades relevantes dos usuários. Esta relevância é especificada pelo administrador. Por exemplo, um arquivo do tipo *log*, em formato texto, pode ter registrado todos os nomes de usuários que efetuaram *login* em um período, a data do *login*, o horário do *login* e o horário do *logout*. Este arquivo pode ser examinado em busca de atividades suspeitas. Escreva um programa para auxiliar o administrador nesta tarefa.

- Considere a existência de um arquivo fictício *login.log*, onde estão armazenados, em cada linha, nome (*login*) do usuário, data de *login*, horário de *login* e horário de *logout*.
- Construir as estruturas de dados apropriadas para conter os dados.
- Ler o arquivo e preencher uma lista estática.
- Implementar as seguintes funções para o usuário, que serão mostradas em um menu:
    1. *mostraUsuarios*: visualizar na tela todos os nomes que aparecem no arquivo, desprezando repetições de nomes de *login*.
    2. *mostraAposHora*: visualizar na tela todos os nomes que efetuaram *login* após um horário indicado pelo administrador.
    3. *mostraAcessoUsuario*: visualiza todos os acesso feitos por um determinado nome indicado pelo administrador.

3- Implemente uma nova função de inserção em lista seqüencial de forma que ela insira *NrCopias* cópias de um certo dado a partir de uma posição *Pos*.

```
void InsereNC(TipoLista *Lista, TipoItem Item, Apontador Pos, int NrCopias);
```

4- Implemente uma nova função de remoção em lista seqüencial de forma que ela remova de uma só vez *Qte* elementos a partir de uma posição *Pos*.

```
void RemoveNC(TipoLista *Lista, Apontador Pos, int Qte);
```

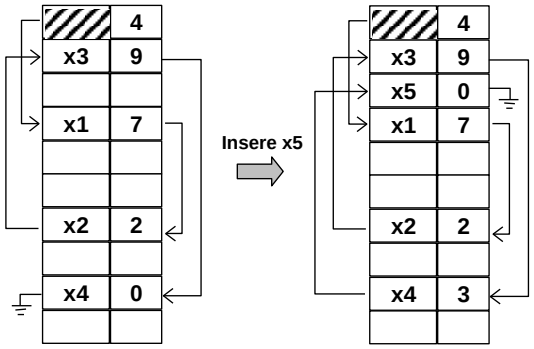
5- Escreva o código das seguintes operações adicionais para o TAD Lista Linear (supondo a implementação feita através de um vetor):

- Verificar se uma Lista L_1 está ordenada (em ordem crescente ou decrescente)
- Copiar uma Lista L_1 para uma outra L_2. Use necessariamente a função *insere*.
- Copiar uma Lista L_1 para uma Lista L_2 eliminando elementos repetidos. Use a função *insere*.
- Inverter uma lista L_1
- Intercalar L_1 com L_2 gerando L_3 sem elementos repetidos (considere L1 e L_2 ordenadas)
- A partir de L_1 e L_2, gere L_4 considerando os elementos repetidos em ambas as listas (considere L1 e L_2 ordenadas)

6- Construa uma função que recebe como parâmetros uma Lista L_1 e um valor x e que retire os primeiros x da lista L_1, inserindo-os no fim de L_1. Use as funções de inserção e remoção já implementados. Suponha que a lista está inicialmente preenchida. É proibido o uso de uma lista auxiliar.

7- Uma lista com alocação encadeada pode ser implementada de maneira estática utilizando-se um vetor como mostrado abaixo. A grande vantagem deste tipo de implementação é que ela evita movimentos durante inserção e remoção de elementos da lista. Implemente as operações de remoção e inserção para esta estrutura de lista.

```
const int N=100; {no máximo de elementos}
const int nil=0; {valor 0 indica fim da
lista}
typedef struct {
               int chave;
               int prox;
} Telem;
Typedef struct {
                Telem A[N];
                int Primeiro,Ultimo;
                int Disponivel;
} Lista;
```



PARTE 2 – Listas Encadeadas com Alocação Dinâmica

8- Escreva uma função que receba como parâmetro uma lista encadeada contendo apenas valores 0 e 1 e que retorne as posições inicial e final da maior seqüência S de elementos 0 dentro da lista.
Ex: Lista={0,1,1,0,0,0,1,0} pini = 4 e pfim = 6 S={0,0,0}
Obs: No caso de empate em tamanho, a primeira ocorrência de S é a que deve ser retornada.

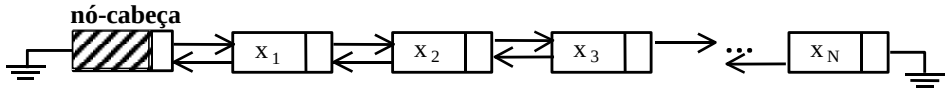
9- Construa um algoritmo que retira um elemento da posição pos_1 e coloca na pos_2.

10- Considere uma lista simplesmente encadeada L_1 representando uma seqüência de caracteres. Construa uma função para imprimir a seqüência de caracteres da lista L_1 na ordem inversa (não é permitido o uso de listas auxiliares).

Ex: Para a lista L_1={A,E,I,O,U}, a função deve imprimir "UOIEA".

11- Repita a questão 4 considerando agora que a lista linear foi implementada através de vetores.

12- Para a lista duplamente encadeada representada pela figura a seguir, escreva:



(a) A estrutura de dados para implementar esta lista.

(b) A função `TipoItem *Locate(TipoLista L, int P)` que retorna o apontador para o item na P-ésima posição de L. Retorne NULL se a lista tiver menos que P elementos.

(c) A função `void InsereP(TipoLista *L, TipoItem E, int P)` que insere o item E na posição P da lista L. Use a função Locate(L,P) neste procedimento. Retorne erro se a lista tiver menos que P−1 itens.

(d) A função `TipoItem RemovaP(TipoLista *L, int P)` que remove e retorne o item na P-ésima posição da lista L. Use a função Locate(L,P) neste procedimento. Retorne erro se a lista tiver menos que P itens.

13- Escreva os algoritmos de inserção e remoção de um elemento para uma lista duplamente encadeada. Use uma função `TipoItem *Busca_Dup_Ord(TipoLista L, TipoChave x)` que retorna um ponteiro para o nó com chave igual a x na lista L.

14- Considere duas listas simplesmente encadeada de números inteiros L_1 e L_2

(a) Crie uma lista L_3 que seja a união sem repetição de L_1 e L_2

(b) Crie uma lista L_4 que seja a intersecção dos elementos que estão em L_1

PARTE 3 – Pilhas e Filas

15- Escreva um programa para verificar se uma expressão matemática tem os parênteses agrupados de forma correta, isto é: **(1)** se o número de parênteses à esquerda e à direita são iguais e; **(2)** se todo parêntese

aberto é seguido posteriormente por um fechamento de parêntese. (**ver Capítulo 2 do Livro do Tenembaum**).

- Ex1: As expressões ((A+B) ou A+B(violam a condição 1
- Ex2: As expressões)A+B(– C ou (A+B)) – (C + D violam a condição 2

16- Escreva um algoritmo para determinar se uma string de caracteres de entrada é da forma *xCy*, onde *x* é uma string consistindo das letras A e B e *y* é o inverso de *x*, isto é se *x* = ABABBA (**use uma pilha para resolver o problema**).

17- Mostre como uma pilha pode ser implementada utilizando duas filas. Analise o tempo das operações de empilhar (PUSH) e desempilhar (POP).

18- Deque (ou fila de duas pontas) é uma estrutura de dados que consiste de uma lista na qual as seguintes operaçõoes são permitidas:

(a) Empilha(A) Insere o elemento A no início da deque.
(b) Desempilha() Remove o elemento que está no início da deque.
(c) Inject(A) Insere o elemento A no final da deque.
(d) Eject() Remove o elemento que está no final da deque.

Crie uma estrutura de dados adequada e implemente as operações acima.

19- Considere o seguinte conjunto de números 1234 e as seguintes operações:

(a) inserir o no. 1 em uma pilha;
(b) inserir o 2 na pilha;
(c) retirar o 2 da pilha;
(d) inserir o 3 na pilha;
(e) inserir o 4 na pilha;
(f) retirar o 4 da pilha;.
(g) retirar o 3 da pilha;
(h) retirar o 1 da pilha.

A seqüência de saídas do procedimento acima é 2431. Considere agora a seqüência 123456.

a) Podemos obter as seqüências 325641 e 154623 utilizando um processo semelhante ao do exemplo anterior?

b) Se I e R representam respectivamente inserção e remoção da pilha, o exemplo acima pode ser descrito como IIRIIRRR. Se possível descreva as seqüências do item (a) em termos de I e R.

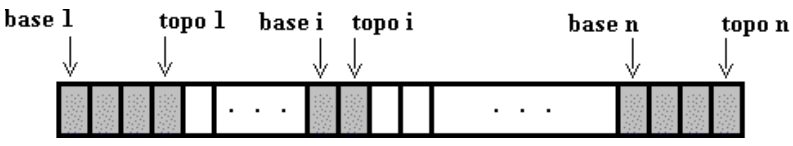
c) Qual seria uma regra simples para anlisar se uma sequ^encia de I's e R's é valida?

d) Interprete a seguinte proposição: *Existe uma permutação p_1p_2 ... p_n dos números 1234...n usando uma pilha se e somente se não existem índices i < j < k tal que $p_j < p_k < p_i$.*

20- Duas pilhas podem ser implementadas em um único *array* A da seguinte forma: A primeira pilha cresce a partir do início do array para direita e a segunda cresce a partir do final do *array* para a esquerda, ou seja, as pilhas crescem uma em direção a outra. Escreva os seguintes procedimentos:

(a) Empilha(A,i)→ insere o elemento A na pilha i = 1,2.
(b) Desempilha(i) → retorna o elemento que está no topo da pilha i = 1,2.

Existe um problema com este tipo de implementação. Qual é?



Referências:

1- SZWARCFITER, J.L.; MARKENZON, L. *Estruturas de Dados e Seus Algoritmos*. Editora LTC, 1998.
2- TENENBAUM, A.M.; LANGSAM Y.; AUGENSTEIN,M.A. - Estruturas de Dados em C - Prentice-Hall, 1993.
3- Lista do Prof Luis Gustavo Nonato (ICMC-USP, São Carlos), SCE5763, 2000.