ESTRUTURA DE DADOS E ANÁLISE DE ALGORITMOS

Prof. Calvetti

LISTA DE EXERCÍCIOS 2

Pilhas:

1º) Elaborar um programa, em linguagem Java, capaz de armazenar no máximo 10 caracteres em um vetor, manipulando-os através da estrutura de dados denominada pilha (implementação conceitual). O elemento de índice 0 desse vetor deverá ser o primeiro a ser armazenado e por consequência o último a ser retirado. As operações básicas dessa estrutura de dados deverão obedecer ao seguinte menu:

- Inserir Elemento;
- Consultar Elemento;
- Retirar Elemento;
- Sair.

Optando por Inserir Elemento, o programa deverá pedir ao operador que digite o elemento a ser inserido na pilha; optando por Consultar Elemento, o programa deverá apresentar em tela o último elemento inserido na pilha, sem que o mesmo seja retirado dela e a quantidade de elementos armazenados na pilha; optando por Retirar Elemento, o programa deverá retirar o último elemento da pilha, apresentando-o em tela; E, optando por Sair, o programa deverá ser encerrado.

2º) Idem ao anterior, porém implementando-o através de TADs (Tipos Abstratos de Dados).

3º) Idem ao exercício 1, porém levando-se em conta que o elemento de índice 9 deverá ser o primeiro armazenado e o elemento de índice 0 deverá ser o último armazenado na pilha.

4º) Idem ao anterior, porém implementando-o através de TADs (Tipos Abstratos de Dados).

5º) Utilizando o mesmo princípio das funções do exercício anterior, elaborar um programa que realize as 4 operações matemáticas entre 5 números do tipo *int*, que deverão ser digitados primeiramente em sua totalidade e armazenados em uma pilha e só após isto, conforme as operações matemáticas que forem sendo digitadas, o resultado da conta venha sendo apresentado sempre em relação aos últimos elementos da pilha remanescente.

Exemplo:	Digitados		Apresentado	Observação
	6	↵	6	
	2	↵	2	
	3	↵	3	
	4	↵	4	
	5	↵	5	
	+	↵	9	<5+4>
	-	↵	6	<9-3>
	*	↵	12	<6*2>
	/	↵	2	<12/6>

6º) Idem ao anterior, porém implementando-o através de TADs (Tipos Abstratos de Dados).

7º) Elaborar um programa, em linguagem Java, capaz de armazenar no máximo 10 nomes em um vetor, manipulando-os através de pilha (implementação através de TADs). As operações básicas deverão obedecer ao seguinte menu:

- Inserir Nome;
- Consultar Nome;
- Retirar Nome;
- Sair.

Optando por Inserir Nome, o programa deverá pedir ao operador que digite o nome a ser inserido na pilha; optando por Consultar Nome, o programa deverá apresentar em tela o último nome inserido na pilha, sem que o mesmo seja retirado dela e a quantidade de nomes armazenados na pilha; optando por Retirar Nome, o programa deverá retirar o último nome da pilha, apresentando-o em tela; E, optando por Sair, o programa deverá ser encerrado.

Filas:

8º) Elaborar um programa, em linguagem Java, capaz de armazenar no máximo 10 caracteres em um vetor, manipulando-os através da estrutura de dados denominada fila (implementação conceitual). O elemento de índice 0 desse vetor deverá ser o primeiro a ser armazenado e por consequência o primeiro a ser retirado. As operações básicas dessa estrutura de dados deverão obedecer ao seguinte menu:

- Inserir Elemento;
- Consultar Elemento;
- Retirar Elemento;
- Sair.

Optando por Inserir Elemento, o programa deverá pedir ao operador que digite o elemento a ser inserido na fila; optando por Consultar Elemento, o programa deverá apresentar em tela o primeiro elemento inserido na fila, sem que o mesmo seja retirado dela e a quantidade de elementos armazenados na fila; optando por Retirar Elemento, o programa deverá retirar o primeiro elemento da fila, apresentando-o em tela; E, optando por Sair, o programa deverá ser encerrado.

9º) Idem ao anterior, porém implementando-o através de TADs (Tipos Abstratos de Dados).

10º) Idem ao exercício 8, porém levando-se em conta que o elemento de índice 9 deverá ser o primeiro armazenado e o elemento de índice 0 deverá ser o último armazenado na fila.

11º) Idem ao anterior, porém implementando-o através de TADs (Tipos Abstratos de Dados).

12º) Utilizando o mesmo princípio das funções do exercício anterior, elaborar um programa que realize as 4 operações matemáticas entre 5 números do tipo *int*, que deverão ser digitados primeiramente em sua totalidade e armazenados em uma fila e só após isto, conforme as operações matemáticas que forem sendo digitadas, o resultado da conta venha sendo apresentado sempre em relação aos primeiros elementos da fila remanescente.

Exemplo:	Digitados		Apresentado	Observação
	6	↵	6	
	2	↵	2	
	3	↵	3	
	4	↵	4	
	5	↵	5	
	+	↵	8	<6+2>
	-	↵	5	<8-3>
	*	↵	20	<5*4>
	/	↵	4	<20/5>

13º) Idem ao anterior, porém implementando-o através de TADs (Tipos Abstratos de Dados).

14º) Elaborar um programa, em linguagem Java, capaz de armazenar no máximo 10 nomes em um vetor, manipulando-os através de fila (implementação através de TADs). As operações básicas deverão obedecer ao seguinte menu:

- Inserir Nome;
- Consultar Nome;
- Retirar Nome;
- Sair.

Optando por Inserir Nome, o programa deverá pedir ao operador que digite o nome a ser inserido na fila; optando por Consultar Nome, o programa deverá apresentar em tela o primeiro nome inserido na fila, sem que o mesmo seja retirado dela e a quantidade de nomes armazenados na fila; optando por Retirar Nome, o programa deverá retirar o primeiro nome da fila, apresentando-o em tela; E, optando por Sair, o programa deverá ser encerrado.

Filas Circulares:

15º) Elaborar um programa, em linguagem Java, capaz de armazenar no máximo 10 caracteres em um vetor, manipulando-os através da estrutura de dados denominada fila circular (implementação conceitual). O elemento de índice 0 desse vetor deverá ser o primeiro a ser armazenado e por consequência o primeiro a ser retirado. As operações básicas dessa estrutura de dados deverão obedecer ao seguinte menu:

- Inserir Elemento;
- Consultar Elemento;
- Retirar Elemento;
- Sair.

Optando por Inserir Elemento, o programa deverá pedir ao operador que digite o elemento a ser inserido na fila circular; Optando por Consultar Elemento, o programa deverá apresentar em tela o primeiro elemento inserido na fila circular, sem que o mesmo seja retirado dela e a quantidade de elementos armazenados na fila circular; Optando por Retirar Elemento, o programa deverá retirar o primeiro elemento da fila circular, apresentando-o em tela; E, optando por Sair, o programa deverá ser encerrado.

16º) Idem ao anterior, porém implementando-o através de TADs (Tipos Abstratos de Dados).

17º) Idem ao exercício 15, porém levando-se em conta que o elemento de índice 9 deverá ser o primeiro armazenado e o elemento de índice 0 deverá ser o último armazenado na fila circular, e assim seguindo sucessivamente.

18º) Idem ao anterior, porém implementando-o através de TADs (Tipos Abstratos de Dados).

19º) Utilizando o mesmo princípio das funções do exercício anterior, elaborar um programa que realize as 4 operações matemáticas entre 5 números do tipo *int*, que deverão ser digitados primeiramente em sua totalidade e armazenados em uma fila circular e só após isto, conforme as operações matemáticas que forem sendo digitadas, o resultado da conta venha sendo apresentado sempre em relação aos primeiros elementos da fila circular remanescente.

Exemplo:	Digitados	Apresentado	Observação
	6 ↵	6	
	2 ↵	2	
	3 ↵	3	
	4 ↵	4	
	5 ↵	5	
	+ ↵	8	<6+2>
	- ↵	5	<8-3>
	* ↵	20	<5*4>
	/ ↵	4	<20/5>

20º) Idem ao anterior, porém implementando-o através de TADs (Tipos Abstratos de Dados).

21º) Elaborar um programa, em linguagem Java, capaz de armazenar no máximo 10 nomes em um vetor, manipulando-os através de fila circular (implementação através de TADs). As operações básicas deverão obedecer ao seguinte menu:

- Inserir Nome;
- Consultar Nome;
- Retirar Nome;
- Sair.

Optando por Inserir Nome, o programa deverá pedir ao operador que digite o nome a ser inserido na fila circular; optando por Consultar Nome, o programa deverá apresentar em tela o primeiro nome inserido na fila circular, sem que o mesmo seja retirado dela e a quantidade de nomes armazenados na fila circular; optando por Retirar Nome, o programa deverá retirar o primeiro nome da fila circular, apresentando-o em tela; E, optando por Sair, o programa deverá ser encerrado.

Listas Singularmente Ligadas / Listas Simplesmente Encadeadas /

Listas com Encadeamento Simples / Listas Singularmente Encadeadas:

22º) Utilizando-se de alocação dinâmica de memória para armazenamento em listas singularmente encadeadas, elaborar um programa em linguagem Java que realize as seguintes funções:

a) Inserir um novo nó em uma lista, sempre ao final dela, digitando seu conteúdo;
b) Remover sempre o último nó da respectiva lista;
c) Sair.

Cada nó da lista deverá ter um campo para armazenar números inteiros e positivos denominado *elemento* e um campo para armazenar a ligação ao próximo endereço da lista denominado *próximo*, formando a estrutura de dados denominada Lista Singularmente Encadeada.

23º) Utilizando-se de alocação dinâmica de memória para armazenamento em listas singularmente encadeadas, elaborar um programa em linguagem Java que realize as seguintes funções:

a) Inserir um novo nó em uma lista, sempre ao final dela, digitando seu conteúdo;
b) Consultar todos os nós, do primeiro ao último, sem retirá-los da respectiva lista;
c) Remover sempre o último nó da respectiva lista;
d) Sair.

Cada nó da lista deverá ter um campo para armazenar números inteiros e positivos denominado *elemento* e um campo para armazenar a ligação ao próximo endereço da lista denominado *próximo*, formando a estrutura de dados denominada Lista Singularmente Encadeada.

24º) Utilizando-se de alocação dinâmica de memória para armazenamento em listas singularmente encadeadas, elaborar um programa em linguagem Java que realize as seguintes funções:

a) Inserir um novo nó em uma lista, sempre ao final dela, digitando seu conteúdo;
b) Apresentar todos os nós da respectiva lista, sem remove-los, do primeiro ao último e um a um, com seus conteúdos;
c) Remover sempre o último nó da respectiva lista;
d) Sair.

Cada nó da lista deverá ter um campo para armazenar *String* denominado *elemento* e um campo para armazenar a ligação ao próximo endereço da lista denominado *proximo*, formando a estrutura de dados denominada Lista Singularmente Encadeada.

25º) Utilizando-se de alocação dinâmica de memória para armazenamento em listas singularmente encadeadas, elaborar um programa em linguagem Java que apresente ao operador um menu com as seguintes opções, sobre uma lista denominada *lista*:

a) Inicializar a lista fazendo com que o próximo nó a ser apresentado, quando o operador escolher a opção b), seja o primeiro nó. Apresentar a mensagem "Lista Inicializada";
b) Apresentar o conteúdo do próximo nó a cada vez que esta opção for escolhida, percorrendo assim toda a lista através desta opção. Se a lista estiver vazia, apresentar a mensagem "Lista Vazia". Se for o último nó, após a apresentação de seu conteúdo, apresentar a mensagem "Último Nó da Lista". Se não existir o próximo nó, apresentar a mensagem "Lista Encerrada";
c) Inserir um nó na posição corrente da lista (após o último nó apresentado e antes do próximo nó a ser apresentado), sendo que seu conteúdo deverá ser digitado pelo operador. Após a inserção, a lista deverá ser inicializada (opção a) automaticamente;
d) Eliminar da lista o último nó válido que foi apresentado em tela. Apresentar a mensagem "Nó Eliminado" em caso de sucesso. Se a lista foi inicializada imediatamente antes ou se a lista estiver vazia, apresentar a mensagem "Não há nó a ser Eliminado". Após a eliminação, a lista deverá ser inicializada (opção a) automaticamente.

Cada nó da lista deverá ter um campo *String* de tamanho 20 caracteres denominado *nome* para armazenar o nome do aluno, um campo *long* denominado *ra* para armazenar o RA do aluno e um campo para armazenar a ligação ao próximo endereço da lista denominado *proximo*.

26º) Utilizando-se de alocação dinâmica de memória para armazenamento em listas singularmente encadeadas, elaborar um programa em linguagem Java que realize as seguintes funções:

a) Deverá ser digitada uma frase qualquer, de tamanho indeterminado, composta por diversos caracteres;
b) Cada caractere dessa frase deverá ser armazenado em um nó diferente de uma lista denominada *lista*;
c) Deverá ser digitado um caractere qualquer que deverá ser removido da *lista* em todas as posições onde ele se encontrar;
d) Deverá ser apresentada a frase resultante, sem a presença em nenhuma posição do caractere digitado.
e) Fazer todas as consistências necessárias, tais como: caractere digitado não se encontra na lista, lista vazia, etc.

<u>Listas Duplamente Ligadas / Listas Duplamente Encadeadas</u>:

27º) Utilizando-se de alocação dinâmica de memória para armazenamento em listas duplamente ligadas, elaborar um programa em linguagem Java que realize as seguintes funções:

a) Inserir novo nó sempre ao final da lista denominada *lista*, digitando-se seus conteúdos. Após a inserção, o nó atual da lista sempre deverá passar a ser o nó inicial da mesma;

b) Remover sempre o último nó da lista denominada *lista*. Após a eliminação, o nó atual da lista sempre deverá passar a ser o nó inicial da mesma;

c) Apresentar o conteúdo do nó atual da lista denominada *lista*;

d) Apresentar o conteúdo do próximo nó da lista denominada *lista*. Caso não haja o próximo nó, apresentar a mensagem "Lista Encerrada". Havendo o próximo nó, após sua apresentação, o nó atual da lista passará a ser o próximo nó da mesma, que acabou de ser apresentado.;

e) Apresentar o conteúdo do nó anterior da lista denominada *lista*. Caso não haja o nó anterior, apresentar a mensagem "Lista Encerrada". Havendo o nó anterior, após sua apresentação, o nó atual da lista passará a ser o nó anterior da mesma, que acabou de ser apresentado.

Cada nó da lista deverá ter um campo para armazenar números inteiros denominado *item*, um campo para armazenar a ligação ao endereço anterior da lista denominado *anterior* e um campo para armazenar a ligação ao próximo endereço da lista denominado *proximo*.

28º) Utilizando-se de alocação dinâmica de memória para armazenamento em listas duplamente ligadas, elaborar um programa em linguagem Java que apresente ao operador um menu com as seguintes opções, sobre uma lista denominada lista:

a) Inicializar a lista, fazendo com que o próximo nó a ser apresentado, quando o operador escolher a opção b), seja o primeiro nó. Apresentar a mensagem "Lista Inicializada";

b) Apresentar o conteúdo do próximo nó a cada vez que esta opção for escolhida, percorrendo assim toda a lista através desta opção. Se a lista estiver vazia, apresentar a mensagem "Lista Vazia". Se for o último nó, após a apresentação de seu conteúdo, apresentar a mensagem "Último da Lista". Se não existir o próximo nó, apresentar a mensagem "Lista Encerrada";

c) Apresentar o conteúdo do nó anterior a cada vez que esta opção for escolhida, percorrendo assim toda a lista através desta opção. Se a lista estiver vazia, apresentar a mensagem "Lista Vazia". Se for o primeiro nó, após a apresentação de seu conteúdo, apresentar a mensagem "Primeiro da Lista". Se não existir o nó anterior, apresentar a mensagem "Lista Encerrada";

d) Inserir um nó na posição corrente da lista (após o último nó apresentado e antes do próximo nó a ser apresentado), sendo que seu conteúdo deverá ser digitado pelo operador;

e) Eliminar da lista o último nó válido que foi apresentado em tela. Apresentar a mensagem "Nó Eliminado" em caso de sucesso. Se a lista foi inicializada imediatamente antes ou se a lista estiver vazia, apresentar a mensagem "Não há nó a ser Eliminado". Após a eliminação, o nó atual da lista sempre deverá passar a ser o nó inicial da mesma.

Cada nó da lista deverá ter um campo *String* de tamanho 20 caracteres denominado *nome* para armazenar o nome do aluno, um campo *long* denominado *ra* para armazenar o RA do aluno, um campo para armazenar a ligação ao endereço anterior da lista denominado *anterior* e um campo para armazenar a ligação ao próximo endereço da lista denominado *proximo*.

29º) Utilizando-se de alocação dinâmica de memória para armazenamento em listas duplamente encadeadas, elaborar um programa em linguagem Java que realize as seguintes funções:

a) Deverá ser digitada uma frase qualquer, de tamanho indeterminado, composta por diversos caracteres;

b) Cada caractere dessa frase deverá ser armazenado em um nó diferente de uma lista denominada *lista*;

c) Deverá ser digitado um caractere qualquer que deverá ser removido da *lista* em todas as posições onde ele se encontrar;

d) Deverá ser apresentada a frase resultante, do primeiro nó ao último da lista, sem a presença em nenhuma posição do caractere digitado;

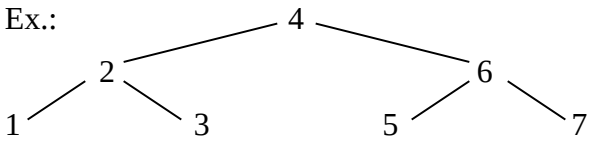
e) Deverá ser apresentada a frase resultante, do último nó ao primeiro da lista, sem a presença em nenhuma posição do caractere digitado;

f) Fazer todas as consistências necessárias, tais como: caractere digitado não se encontra na lista, lista vazia, etc.

<u>Árvores Binárias</u>:

30º) Elaborar um programa em linguagem Java capaz de armazenar itens numéricos e inteiros em NÓS de uma estrutura de dados do tipo ÁRVORE BINÁRIA. O operador deverá digitar os valores para os itens seguindo a regra de Acesso PRÉ-ORDENADO e apresentá-los seguindo o Acesso ORDENADO.

Ex.:

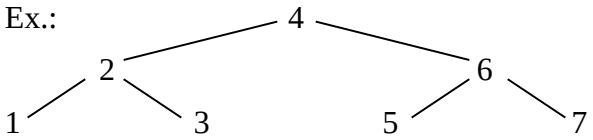


Digitação dos nós da árvore (pré-ordenado): 4 2 1 3 6 5 7

Apresentação dos nós da árvore (ordenado): 1 2 3 4 5 6 7

31º) Elaborar um programa em linguagem Java capaz de armazenar itens numéricos e inteiros em NÓS de uma estrutura de dados do tipo ÁRVORE BINÁRIA. O operador deverá digitar os valores para os itens seguindo a regra de Acesso PRÉ-ORDENADO e apresentá-los seguindo o Acesso PRÉ-ORDENADO.

Ex.:

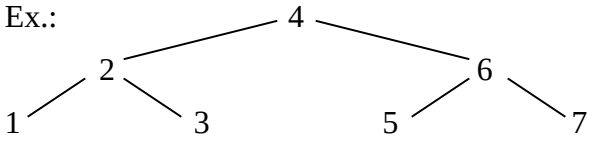


Digitação dos nós da árvore (pré-ordenado): 4 2 1 3 6 5 7

Apresentação dos nós da árvore (ordenado): 4 2 1 3 6 5 7

32º) Elaborar um programa em linguagem Java capaz de armazenar itens numéricos e inteiros em NÓS de uma estrutura de dados do tipo ÁRVORE BINÁRIA. O operador deverá digitar os valores para os itens seguindo a regra de Acesso PRÉ-ORDENADO e apresentá-los seguindo o Acesso PÓS-ORDENADO.

Ex.:

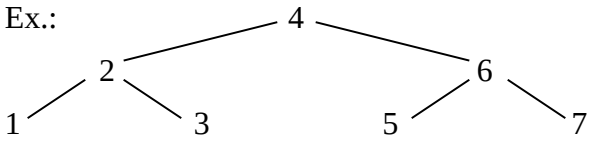


Digitação dos nós da árvore (pré-ordenado): 4 2 1 3 6 5 7

Apresentação dos nós da árvore (ordenado): 1 3 2 5 7 6 4

33º) Elaborar um programa em linguagem Java capaz de armazenar itens numéricos e inteiros em NÓS de uma estrutura de dados do tipo ÁRVORE BINÁRIA. O operador deverá digitar os valores para os itens seguindo a regra de Acesso PRÉ-ORDENADO e apresentá-los seguindo o exemplo abaixo.

Ex.:



Digitação dos nós da árvore (pré-ordenado): 4 2 1 3 6 5 7

Apresentação dos nós da árvore:

7
6
5
4
3
2
1

34º) Elaborar um programa em linguagem Java que apresente ao operador o menu abaixo e execute as funções descritas nele:

A- Inserir Nó na Árvore (Acesso Pré-Ordenado);

B- Apresentar Árvore;

C- Buscar Nó na Árvore;

D- Sair.

Obs.: Os itens A e B são semelhantes ao exercício anterior. O item C deve informar se o nó solicitado está ou não na árvore binária.

35º) Elaborar um programa em linguagem Java que apresente ao operador o menu abaixo e execute as funções descritas nele:

A- Inserir Nó na Árvore (Acesso Pré-Ordenado);

A- Apresentar Árvore;

B- Eliminar Nó da Árvore (Digitado o Conteúdo do Nó);

C- Sair.

Obs.: Os itens A e B são semelhantes ao exercício anterior. O item C deve solicitar a digitação de um Nó, que se existir na árvore, será ser eliminado. Informar todas as possibilidades em tela. Quando eliminar um nó pai que tenha apenas um dos dois nós filhos, esse nó filho assumirá o lugar do nó pai. Quando eliminar um nó pai que tenha os dois nós filhos, o nó filho da direita assumirá o lugar do nó pai e o nó filho da esquerda deverá ser alocado corretamente ao novo nó pai.