Universidade Federal Fluminense
Instituto de Ciências Exatas
Disciplina: Estrutura de Dados
Prof. Vera Caminha

Lista de Exercícios 1

1) Conte o número de passos e dê a complexidade (pior caso) dos trechos de código abaixo:

(a)
```
int soma= 0;
for(i=1; i<=n; i++ )
        soma++;
```

(b)
```
int soma = 0;
for(  i=0; i<=n-1; i++ )
        for( j=1; j<=n; j++ )
                soma++;
```

(c)
```
int soma = 0;
for(  i=0; i<=n*n; i++ )
        for(  j=0; j<n*n; j++ )
                soma++;
```

(d)
```
int soma = 0;
for( i=0; i<n-1; i++ )
        for( j=1; j<n; j++ )
                soma++;
```

(e) Considere a, b e prod matrizes quadradas n x n.

```
for ( i = 0; i <n; i++)
{
   for ( j=1; j < n; j++)
   {
            prod[i][j] = 0;
            for ( k = 2; k < n; k++)
               prod[i][j] = prod[i][j] + a[i][k] * b[k][j];
   }
}
```

(f)
```
void par_impar(int n)
  {
        int i, j;
        for (i= 0; i <= n, i++)
            if ( i%2 == 0) {
                for (j = 1; j < n; j++)
                        comandoX;
                for (j = 1; j < n; j++)
                        comandoX;
            }

   }
```

(g)
```
void faz_algo1( int n)
    {
      int i, j, k;
      for ( i = 1; i < n; i ++)
            for (j =2; j < =n; j++)
               for (k = 3; k <= n ; k++)
                        comandoX;
    }
```

(h)
```
void conta( int n)
    {
       int i, j;
       for ( i = 0; i <= n; i ++)
       {
            for (j = 0; j <= n; j++)
                 comandoX;
             for (j = 0; j <= i; j++)
                comandoX;
      }
   }
```

(i)
```
for (i = 0; i < n; i++) {

    j = 0;

    while (j <= i)

          j = j + 1;
  }
```

(j)
```
for (i = 0; i < n; i++) {

    j = i;

    while (j < n)

        j = j + 1;

  }
```

2) Faça um programa, usando a estrutura de dados pilha, que converta um número decimal para binário.

Ex:

Operação	Resto	Quociente
19/2	**1**	9
9/2	**1**	4
4/2	**0**	2
2/2	**0**	**1**

Resultado: $(19)_{10} = (10011)_2$

3) Escreva um programa que leia 20 números reais e insira-os em uma pilha p1. Logo após, crie uma cópia de p1 em p3 usando uma pilha p2 como estrutura auxiliar.

4) Escreva um programa que utilizando uma pilha, determine se uma string lida é um palíndromo ou não, isto é, se a string pode ser lida da mesma maneira para frente ou para trás. Ex: ovo, arara, osso.

5) Escreva um programa que leia 20 números inteiros e empilhe numa pilha p1. Imprima a pilha p1. Logo após, inverta o conteúdo de p1 nela mesma utilizando uma pilha auxiliar. Imprima a pilha p1 invertida.

6) Escreva um programa, utilizando pilha, que leia uma expressão e verifique se o número de abre parênteses é igual ao número de fecha parênteses.

7) Escreva um programa que empilhe uma sequência de números inteiros positivos até o momento em que for digitado o valor zero. Neste momento, o conteúdo da pilha deverá ser distribuído em outras duas pilhas. Uma delas conterá apenas os valores ímpares e a outra conterá apenas os valores pares. Imprima o conteúdo das pilhas.

8) Escreva um programa que utilize a estrutura pilha para conversão de expressões da notação tradicional (infixa), completamente parentizadas, para a e notação polonesa reversa (pós-fixada).

9) Escreva um programa que leia um número indeterminado de valores inteiros. O valor 0 (zero) finaliza a entrada de dados. Para cada valor lido, determinar se ele é um número par ou ímpar. Se o número for par, então incluí-lo na FILA_PAR; caso contrário, incluí-lo na FILA_IMPAR. Após o término da entrada de dados, retirar um elemento de cada fila alternadamente (iniciando-se pela FILA_IMPAR) até que ambas as filas estejam vazias. Se o elemento retirado de uma das filas for um valor positivo, então incluí-lo em uma PILHA; caso contrário, remover o elemento da FILA. Finalmente, escrever o conteúdo da pilha. Considere que as filas e a pilha são sequenciais e que possuem no máximo 50 elementos.

10) Escreva um programa que leia um número indeterminado de valores inteiros e insira-os numa pilha. O valor 0 (zero) finaliza a entrada de dados. Em seguida, leia um número indeterminado de valores inteiros e insira-os numa fila. O valor 0 (zero) finaliza a entrada de dados. Logo após, forneça o maior, o menor e a média aritmética dos elementos na pilha e na fila, respectivamente. O tamanho de cada estrutura deve ser no máximo 50.

11) Desenvolva um programa leia duas filas f1 e f2 de inteiros. Em seguida, chame uma função para testar se a fila f1 tem mais elementos que a fila f2.

12) Desenvolva um programa que leia duas pilhas com n elementos inteiros. Em seguida, chame uma função para testar se as duas pilhas P1 e P2 são iguais. Duas pilhas são iguais se possuem os mesmos elementos, na mesma ordem. Você pode utilizar pilhas auxiliares também, se necessário.

13) Escreva um programa que lê uma lista de dados de 40 alunos (nome e nota) e forneça as seguintes opções:

a) inserir um aluno; (ler nome e nota)
b) excluir um aluno;
c) calcular média da turma;
d) imprimir alunos acima da média;
e) ordenar a lista;
f) imprimir lista;
g) sair.

Na implementação da opção **e**, deve-se utilizar o algoritmo inserctionSort.

14) Dada uma estrutura com nome, conta corrente e saldo. Criar um programa, usando lista sequencial ordenada, com opções para inserir, buscar e remover uma conta corrente. Na busca mostrar o nome e o saldo.

15) Considere a lista de clientes da livraria, em que os elementos já estão ordenados. Escreva um programa que contenha funções para:

a) Ler uma identificação e buscar o cliente (usar busca binária);
b) Inserir um novo cliente;
c) Retirar um cliente.

OBS: Todas as funções deverão retornar sucesso(1) ou falha (0).

16) Escreva um programa que leia um vetor de inteiros, ordene o vetor na ordem decrescente e chame uma função que faça a busca de um elemento num vetor de valores inteiros ordenados em ordem decrescente. A função deve retornar 1 se o elemento estiver presente no vetor e 0, caso contrário.

17) Dada uma lista sequencial ordenada, escreva um programa que contenha uma função que inverta a ordem dos elementos na lista, utilizando para isso uma pilha como estrutura auxiliar.

18) Escreva um programa que contenha as seguintes operações adicionais para uma Lista Sequencial:

- Verificar se uma Lista L1 está ordenada (em ordem crescente ou decrescente);
- Copiar uma Lista L1 para uma outra L2. Use necessariamente a função insere().;
- Copiar uma Lista L1 para uma Lista L2 eliminando elementos repetidos. Use a função insere();
- Inverter uma lista L1;
- Intercalar L1 com L2 gerando L3 sem elementos repetidos (considere L1 e L2 ordenadas);
- A partir de L1 e L2, gere L4 considerando os elementos repetidos em ambas as listas (considere L1 e L2 ordenadas).

19) Faça um programa em C que exiba o seguinte menu principal:

- - - - - -Editor de Listas - - - - - - - -
1- Exibir lista
2- Inserir na lista
3- Remover da lista
4- Exibir elemento da lista
5- Exibir posição de elemento na lista
6- Esvaziar lista
7- Sair

O usuário deve digitar a opção de sua preferência para executar uma operação. implemente a lista de números inteiros. Cada uma das operações do menu é descrita a seguir:

a) Na opção 1, devem ser exibidos o tamanho corrente da lista e todos os seus elementos;

b) Na opção 2, devem ser lidos não somente o elemento a ser inserido, mas também a posição na lista onde o usuário deseja realizar a inserção;

c) Na opção 3, deve ser lida a posição na lista do elemento a ser removido;

d) Na opção 4, deve ser lida a posição do elemento;

e) Na opção 5, deve ser lido o valor do elemento cuja posição se quer exibir;

f) Na opção 6, deve ser exibida uma mensagem de confirmação para o usuário antes da lista ser totalmente esvaziada;

g) Após a execução de cada operação, o programa deve retornar ao menu principal para que o usuário possa executar outras opções ou encerre o programa;

h) Lembre-se: em cada uma das operações, identifique possíveis situações de erros do usuário e exiba mensagens para ele nestas situações. (Ex. o programa deve exibir mensagens no caso do usuário tentar remover um item ou esvaziar uma lista que está vazia, inserir elemento numa lista que já está cheia, inserir posições inválidas, etc.).

20) Reescreva o algoritmo de ordenação bubblesort que permute os elementos de um vetor inteiro `v[0 . . .n-1]` de modo que eles fiquem em ordem decrescente.

21) Reescreva o algoritmo de ordenação seleção que permute os elementos de um vetor inteiro `v[0 ... n-1]`de modo que eles fiquem em ordem decrescente.

22) Reescreva o algoritmo de ordenação inserção que permute os elementos de um vetor inteiro v[0. . . n-1] de modo que eles fiquem em ordem decrescente.

23) Um DEQUE é um caso especial de listas, é uma estrutura de dados, onde as retiradas e as inserções podem ser realizadas nas duas extremidades. Utilize as variáveis direita e esquerda que referenciam cada uma das extremidades e faça as funções abaixo:

a) Inicializar a deque
b) Testar deque vazia
c) Testar deque cheia
d) Inserir a direita
e) Inserir a esquerda
f) Remover a direita
g) Remover a esquerda

DESAFIO 1:

Invente um algoritmo de ordenação que seja *mais rápido* que os algoritmos de inserção e o de seleção vistos em sala de aula.

DESAFIO 2: Faça um editor gráfico, inspirado no PowerPoint ou Paint, que permita ao usuário editar um gráfico composto pelas seguintes primitivas gráficas: ponto, reta, círculo, retângulo ou texto. Cada primitiva possui seus próprios parâmetros: ponto (x, y, cor), reta (xini, yini, xfim, yfim, cor), círculo (x, y, raio, cor), retângulo ((xini, yini, xfim, yfim, cor), texto (x, y, mensagem_de_texto). A partir da definição desta primitivas, faça um programa que permita que seja fornecido pelo usuário uma coleção de primitivas a serem armazenadas em uma estrutura de dados em memória: o programa deve consistir de um laço onde é perguntado ao usuário o tipo da primitiva (p=ponto, r=reta, c=círculo, b=box, t=texto) e seus parâmetros, e depois perguntando se deseja inserir mais primitivas ou sair do programa. Uma vez terminada a inserção das primitivas, você deve salvar em um arquivo texto todos os dados digitados.