UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
Bacharelado em Ciência da Computação / Engenharia da Computação
INF 01203 – Estruturas de Dados

PRIMEIRA AVALIAÇÃO

IDENTIFICAÇÃO
Nome: _____ **14/04/2009**

01. ***(valor 1,0)*** Descreva a saída resultante da seguinte série de operações de pilha: push(5), push(3), pop(), push(2), push(8), pop(), pop(), push(9), push(1), pop(),push(7), push(6), pop(), pop(),push(4), pop(), pop(). Responda, também, como fica a pilha ao final da execução?

02. ***(valor 1.0)*** Quais abordagens você utilizaria para a especificação do **modelo lógico** (*lista, pilha, fila, deque*) e do **modelo físico** (*lista com array, lista circular com array, lista simplesmente encadaeada, lista simplemente encadeada circular, lista duplamente encadeada, lista duplamente encadeada circular*), se o objetivo fosse implementar uma estrutura de dados que otmizasse os exemplo descrito a seguir. Justifique sua resposta.

a) Uma aplicação na qual a verificação de agrupamento é importante é na validação de documentos XML. HTML é um formato padrão para hiperdocumentos na Internet. Em um documento HTML, porções de texto são delimitadas por ***tags HTML***. Uma *tag* de abertura simples tem a forma **"<nome>"** e a *tag* de fechamento tem a forma **"</nome>"**. No caso ideal, todas as tags de um documento HTML devem casar. Qual seria o modelo lógico e físico adequados se quiséssemos implementar uma função para verificar a correção das tags de um docuemento HTML.

03. ***(valor 2,5)*** Explique, sucintamente, o que faz o trecho de código apresentado a seguir (Função **OqueSera**).

Estrutura de Dados

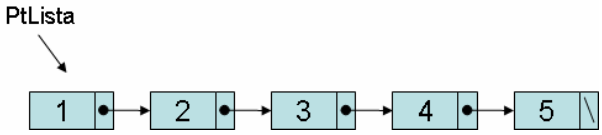
```
typedef int TipoInfoNo;

typedef struct Nodo
        {
           TipoInfoNo info;
           struct Nodo *elo;
        }  TipoNo;

typedef TipoNo *TipoPtNo;
```

Função **OQueSerá**

```
void OQueSera(TipoPtNo *PtLista){
TipoPtNo ptk,ptant,ptaux;
  ptk=(*PtLista);
  while (ptk->elo!=NULL) ptk=ptk->elo;
  ptaux=ptk;
  do{
    ptk=(*PtLista);
    while (ptk->elo!=NULL){
      ptant=ptk;
      ptk=ptk->elo;
    }
```

```
    ptk->elo=ptant;
    ptant->elo=NULL;
    } while (ptk!=(*PtLista));
    (*PtLista)=ptaux;
}
```

03.b) Demonstre a execução da função (teste de mesa) para a lista apresentada a seguir. Qual a saída da função?



04. ***(valor 2,5)*** Especificar uma função em C para testar se duas pilhas são iguais (conteúdo e estrutura). A função deve retornar o valor 1 (um) se as duas pilhas forem iguais e 0 (zero) caso contrário. Ao final da execução, as duas pilhas devem estar no mesmo estado em que estavam no início.

Atenção: A implementação da função deve utilizar o TAD Pilha apresentado a seguir. O acesso a pilha deve ser feito somente através das funções apresentadas nos TADs abaixo.

```
typedef int TipoInfo;

struct TPtPilha{
      TipoInfo dado;
      struct TPtPilha *elo;
};

typedef struct TPtPilha TipoPilha;

TipoPilha* InicializaPilha (TipoPilha *Topo);
int Vazia (TipoPilha *Topo);
TipoPilha* PushPilha (TipoPilha *Topo, TipoInfo Dado);
int PopPilha (TipoPilha **Topo, TipoInfo *Dado);
TipoInfo ConsultaPilha (TipoPilha *Topo);
TipoPilha* DestroiPilha (TipoPilha *Topo);
```

05. ***(valor 3,0)*** Seja **L** uma lista duplamente encadeada não circular, composta dos números $\mathbf{l_1, l_2, l_3, ..., l_n}$, respectivamente, segundo a ordem de armazenamento. Escreva uma função em C que, percorrendo **L** uma única vez, construa uma outra lista **L'**, formada dos seguintes elementos: $\mathbf{l_2, l_3, ..., l_n, l_1}$.

A resposta da questão deve conter:

- as estruturas de dados necessárias para a resolução do problema;
- a função solicitada, escrita em C. Todas as operações sobre a lista devem ser implementadas.