Lista de exercícios 2

Disciplina: Estruturas de Dados I
Professora: Juliana Pinheiro Campos Pirovani

1) Implemente as seguintes funções para a estrutura de dados lista encadeada vista em sala de aula:

a) Uma função que retorna um ponteiro para o último elemento da lista, com o protótipo:

NoLista* ultimo(NoLista **l);

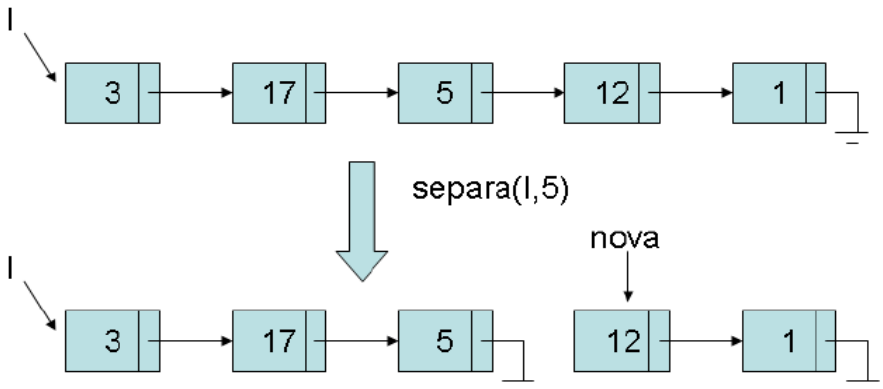
b) Uma função que retorna o número de nós da lista que possuem o campo info com valores maiores do que n, com protótipo:

int maiores(NoLista** l, int n);

c) Função que concatena duas listas encadeadas l1 e l2. Essa função deve retornar a lista l1 contendo todos os seus elementos (na mesma ordem) e, em seguida, todos os elementos de l2. A função deve seguir o protótipo abaixo:

NoLista* concatena(NoLista** l1, NoLista **l2);

d) Função que receba como parâmetro uma lista encadeada e um valor inteiro n e divida a lista em duas, de forma à segunda lista começar no primeiro nó logo após a primeira ocorrência de n na lista original. A figura a seguir ilustra essa separação.



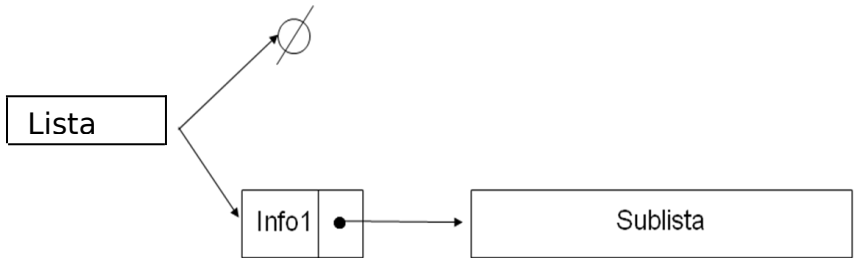
Essa função deve obedecer ao protótipo:

NoLista* separa(NoLista** l, int n);

A função deve retornar um ponteiro para a nova lista, enquanto l deve continuar apontando

para o primeiro elemento da primeira subdivisão da lista.

2) Uma lista encadeada pode ser definida de maneira recursiva da seguinte forma: ou ela é uma lista vazia, ou contém um nó que aponta para uma sublista (demais elementos da lista) conforme figura abaixo.



Implemente as seguintes funções **recursivas** para listas encadeadas:

a) Função que imprime as informações da lista:

```
void imprimeRecursiva(NoLista** l) ;
```

b) Reimplemente a função anterior para que as informações da lista sejam impressas na ordem inversa. OBS: não é para inverter a lista, somente imprimir as informações na ordem inversa.

c) Função para liberar a lista:

```
void liberarLista(NoLista** l) ;
```

d) Função que busca um valor v em uma lista encadeada l, segundo protótipo abaixo. A função deve retornar um ponteiro para o nó cuja informação é igual a v. Caso o valor v não seja encontrado na lista, a função deve retornar NULL.

```
NoLista* buscaRecursiva(NoLista** l, int v);
```

e) Função para remover um elemento da lista. Essa função só remove o elemento se ele for o primeiro da lista. Se não, chama a função recursivamente. Para alguma chamada recursiva, v será o primeiro.

```
void removerElemento(NoLista** l, int v);
```