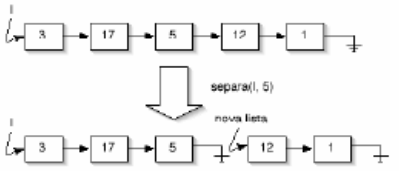
Exercícios de Listas Encadeadas

Observação: Para toda implementação do exercício para lista simples, fazer uma versão para lista circular . Todas as novas funções deverão ser incorporadas às bibliotecas criadas anteriormente (TADs). Os testes deverão ser demonstrados no envio da resolução dos exercícios.

1. Considerando listas de valores inteiros, implemente uma função que receba como parâmetro uma lista encadeada e um valor inteiro n e divida a lista em duas, de tal forma que a segunda lista comece 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:

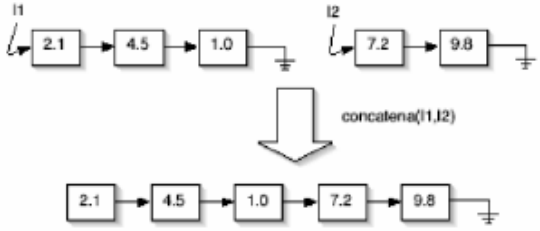
```
Lista* separa (Lista* l, int n);
```

A função deve retornar um ponteiro para a segunda sub-divisão da lista original, enquanto l deve continuar apontando para o primeiro elemento da primeira sub-divisão da lista.

2. Considere estruturas de listas encadeadas que armazenam valores reais. O tipo que representa um nó da lista é dado por:

```
struct lista {
   float info;
   struct lista* prox;
};
typedef struct lista Lista;
```

Implemente uma função que, dadas duas listas encadeadas `l1` e `l2`, concatene a lista `l2` no final da lista `l1`, conforme ilustra a figura abaixo.



A função deve retornar a lista resultante da concatenação, obedecendo ao protótipo:

```
Lista* concatena (Lista* l1, Lista* l2);
```

Observe que `l1` e/ou `l2` podem ser listas vazias.

3. Considere estruturas de listas encadeadas que armazenam valores inteiros. O tipo que representa um nó da lista é dado por:

```
struct lista {
   int info;
   struct lista* prox;
};
typedef struct lista Lista;
```

Implemente uma função que receba um vetor de valores inteiros com `n` elementos e construa uma lista encadeada armazenando os elementos do vetor nos nós da lista. Assim, se for recebido o vetor `v[5] = {3, 8, 1, 7, 2}`, a função deve retornar uma nova lista cujo primeiro nó tem a informação `3`, o segundo a informação `8`, e assim por diante. Se o vetor tiver zero elementos, a função deve ter como valor de retorno uma lista vazia. O protótipo da função é dado por:

```
Lista* constroi (int n, int* v);
```

4. Considere a implementação de uma lista encadeada para armazenar números reais dada pelo tipo abaixo:

```
struct lista {
   float info;
   struct lista* prox;
};
typedef struct lista Lista;
```

Implemente uma função que, dados uma lista encadeada e um número inteiro não negativo `n`, remova da lista seus `n` primeiros nós e retorne a lista resultante. Caso `n` seja maior do que o comprimento da lista, todos os seus elementos devem ser removidos e o resultado da função deve ser uma lista vazia. Essa função deve obedecer o seguinte protótipo:

```
Lista* retira_prefixo (Lista* l, int n);
```