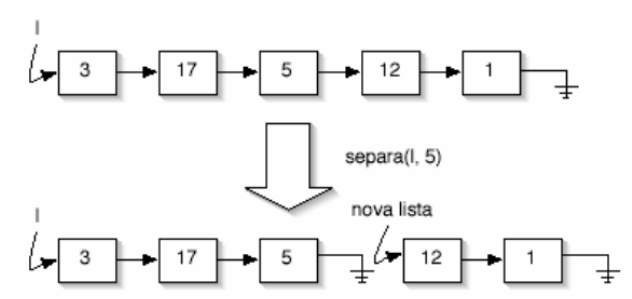
SCC-502 Algoritmos e Estruturas de Dados 1 **M. Cristina/Jorge**

Lista de Exercícios 3 – Listas

Considere a seguinte declaração de um TAD Lista e resolva os exercícios abaixo.

```
typedef int elem;

typedef struct bloco {

        elem info;

        struct bloco *prox;

} no;

typedef struct {

        no *inicio, *fim;

} Lista;
```

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:



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) Escreva uma função que recebe uma lista ordenada e um número inteiro e insere este número de forma ordenada. Siga o protótipo abaixo:

```
void insereOrdenada(Lista *l, int x);
```

3) Escreva uma função que copie um vetor para uma lista encadeada. Considere que a lista l está vazia. l = NULL Siga o protótipo:
void copiaVetorLista (int i[], Lista *l);

4) Escreva uma função que copie uma lista encadeada para um vetor. Siga o protótipo:
void copiaListaVetor (int i[], Lista *l);

5) Escreva uma função que *concatena* duas listas encadeadas (isto é, "amarra" a segunda no fim da primeira) e retorna um ponteiro para a lista concatenada. As listas l1 e l2 não devem ser alteradas. Siga o protótipo:
Lista *listaConcatenada concatena(Lista *l1,Lista *l2);

6) Escreva uma função que verifica se duas listas dadas são *iguais*, ou melhor, se têm o mesmo conteúdo. Retorne 1 se as listas forem iguais e 0 caso contrário. Siga o protótipo:
int iguais(Lista *l1, Lista *l2);

7) Escreva uma função que *desaloca* (função free) todos os nós de uma lista encadeada.
Void desaloca (Lista *l);

8) Escreva uma função que *inverte* a ordem das células de uma lista encadeada (a primeira passa a ser a última, a segunda passa a ser a penúltima etc.). Faça isso sem usar espaço auxiliar; apenas altere os ponteiros. Dê duas soluções: uma iterativa e uma recursiva.
Void inverte (Lista *l);