BCC - Estruturas de Dados
Lab 03 - Listas Simplesmente Encadeadas - Parte 2
Versão ANSI C

Prof. Dr. Paulo César Rodacki Gomes
IFC - Instituto Federal Catarinense

15 de Março de 2023

1 Objetivo

O objetivo desta atividade prática em laboratório é realizar a segunda etapa de implementação de listas simplesmente encadeadas. Por motivo de simplicidade, vamos implementar listas encadeadas para armazenar valores inteiros.

A atividade consiste em implementar em linguagem C as operações a seguir, complementando a implementação feita na lista de exercícios lab 2.

Funções a serem criadas no arquivo `lista_simples.c`:

1. `NoLista *sllRetira(NoLista *head, int v)`: remove da lista o primeiro nó que contiver o valor igual a v. Se nenhum nó com esse valor for encontrado, a função não retira nenhum nó da lista.
   - O valor de retorno da função é o endereço do primeiro nó da lista (lembre-se que, se o nó a ser retirado é o primeiro, a função vai retornar o endereço do "novo"primeiro nó, ou null se a lista ficar vazia;
   - não se esqueça de liberar memória ocupada pelo nó que será retirado da lista;
2. `void sllLibera(NoLista *head)`: libera a lista, ou seja, apaga todo o conteúdo da lista, liberando a memória ocupada por cada um dos seus nós;

3. `NoLista *sllInsereFim(NoLista *head, int v)`: insere um novo nó no final da lista. A função recebe o endereço da cabeça da lista (i.e. o primeiro nó) e o valor a ser inserido.
   - O valor de retorno é o endereço da nova cabeça (este endereço vai mudar quando estivermos inserindo no final de uma lista vazia);
   - Você pode usar a função `ultimo` implementada no exercício anterior para facilitar a inserção no final da lista.
4. `int sllIgual(NoLista *lista1, NoLista *lista2)`: verifica se as duas listas passadas como parâmetros são iguais (neste caso retorna 1) ou diferentes (retorna 0). Para as duas listas serem iguais, elas devem armazenar valores iguais e na mesma ordem;
5. `void sllImprimeRecursivo(NoLista *head)`: versão recursiva da função para impressão dos valores da lista;
6. `NoLista *sllRetiraRecursivo(NoLista *head, int v)`: versão recursiva da função `sllRetira`;
7. `int sllComprimentoRecursivo(NoLista *head)`: versão recursiva da função `sllComprimento`;
8. `NoLista *sllIgualRecursivo()`: versão recursiva da função `sllIgual`.

Observação 1: após implementar a lista, implemente um programa principal para testar e demonstrar o funcionamento da lista implementada.

Observação 2: entregue apenas os arquivos fonte `*.c` e `*.h` separados individualmente (sem "zipar").