UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
Bacharelado em Ciência da Computação e Engenharia da Computação

INF 01203 – Estruturas de Dados

Laboratório 05 – Pilhas (encadeamento)

ATENÇÃO. O trabalho pode ser feito em **duplas**. Somente **UM dos integrantes da dupla deve submeter** o trabalho no *moodle*. Os nomes dos dois integrantes da dupla devem estar como comentário no início de um dos arquivos do código.

Problema para o laboratório. Um sistema tipo GPS armazena o caminho entre dois pontos como uma lista simplesmente encadeada de dados do tipo Rua (descrito a seguir). O campo nome contém o nome da rua e o campo prox aponta para o próximo elemento da lista.

```
struct rua
{
   char nome[51];
   struct rua *prox;
};
typedef struct rua Endereco;
```

`TipoPilha* InicializaPilha (TipoPilha *Topo);`	Retorna o ponteiro para uma nova pilha alocada dinamicamente (pilha vazia).
`int Vazia (TipoPilha *Topo);`	Testa se uma pilha está vazia. Retorna 1 para pilha vazia e zero; caso contrário.
`TipoInfo ConsultaPilha (TipoPilha *Topo);`	Retorno o conteúdo que está no topo da pilha
`int PopPilha (TipoPilha **Topo, TipoInfo *Dado);`	Exclui o elemento do topo da pilha. Armazena o valor excluído na variável Dado, caso a aplicação precise trabalhar esse valor. Retorna 1 se excluiu e zero se não excluiu.
`Pilha* pilha_push (Pilha* p, char* x);`	Insere um novo nó no topo da pilha. O ponteiro da pilha e o elemento a ser inserido são passados como parâmetros. Retorna um ponteiro para a pilha modificada.
`TipoPilha* DestroiPilha (TipoPilha *Topo);`	Esvazia e libera a memória alocada para uma pilha. O ponteiro da pilha é passado como parâmetro.

Crie uma função em C para comparar dois caminhos e verificar se um é o inverso do outro, ou seja, comparar as duas pilhas que armazenam caminhos e verificar se o conteúdo de uma pilha (sequência de nomes de ruas armazenados em cada nó) equivale ao da outra pilha na ordem inversa. Essa função deve usar SOMENTE o tipo abstrato `Pilha` para auxiliar nessa comparação. Os parâmetros da função são os ponteiros `E1` e `E2` para as duas listas. O retorno da função deve ser 1 se uma lista for o inverso da outra e `0`, caso contrário. A função tem o seguinte protótipo:

```
int Compara_pilhas(Endereco* E1, Endereco* E2);
```

A sua função DEVE chamar as funções prontas da TAD_Pilha. Faça as alterações necessárias na TAD_Pilha para a estrutura de dados da aplicação.

ATENÇÃO.: Aproveite o que já temos pronto do código da aula de Pilha. Entretanto, é necessário adaptar o código para estrutura de dados endereço apresentada no enunciado.

ATENÇÃO AOS CRITÉRIOS DE ENTREGA DO TRABALHO.

1. Submeta o seu trabalho no *moodle* (**arquivo zip** contendo: o projeto, o protótipo, a implementação do protótipo e o programa principal).
2. Não esqueça de colocar um comentário no `arquivo.h` com os nomes dos dois integrantes da dupla. Apenas um dos integrantes deve submeter o código no *moodle*.