# 1 Estruturas de dados do tipo Pilha

A estrutura denominada **pilha** é considerada do tipo *FILO (first in last out)*, ou seja, o primeiro elemento inserido será o último a ser removido. Nessa estrutura, cada elemento armazena um ou vários dados (estrutura homogênea ou heterogênea, respectivamente) e um ponteiro para o próximo elemento, permitindo o encadeamento e mantendo a estrutura linear. Nesse tipo de estrutura serão abordadas as seguintes operações: inserir na pilha, consultar toda a pilha, remover e esvaziá-la. Qualquer estrutura desse tipo possui um ponteiro denominado **topo**, no qual todas as operações de inserção e remoção acontecem. Assim, as operações ocorrem sempre na mesma extremidade da estrutura.

## 1.1 Implementação da Estrutura de Dados do tipo Pilha

### 1.1.1 Pilha.h

```
#include <iostream>
class Pilha{
    private:
        std::string Nome, Telefone;
        Pilha *elo;
    public:
        Pilha* InserirPilha(Pilha*, std::string, std::string);
        void PercorrerPilha(Pilha*);
        Pilha* RemoverPilha(Pilha*);
        Pilha* EsvaziarPilha(Pilha*);
};
```

### 1.1.2 Pilha.cpp

```
#include "Pilha.h"

Pilha* Pilha::InserirPilha(Pilha *T, std::string N, std::string Tel){
    Pilha *aux = new Pilha();
    aux->Nome = N;
    aux->Telefone = Tel;
    aux->elo = T;
    T = aux;
    return T;
};
void Pilha::PercorrerPilha(Pilha *T){
    Pilha *aux = T;
    if(aux == NULL){
        std::cout << "\nPilha vazia!\n";
    }else{
        std::cout << "\nRegistros Cadastrados\n";
        while(aux != NULL){
            std::cout << aux->Nome << " - " << aux->Telefone << std::endl;
            aux = aux->elo;
        }
    }
};
Pilha* Pilha::RemoverPilha(Pilha *T){
    Pilha *aux = T;
    T = T->elo;
    delete(aux);
    return T;
};
Pilha* Pilha::EsvaziarPilha(Pilha *T){
    Pilha *aux = T;
    while(aux != NULL){
```

```
32          T = T->elo;
33          delete(aux);
34          aux = T;
35      }
36      return T;
37  };
```

### 1.1.3 main.cpp

```
1   #include "Pilha.h"
2
3   using namespace std;
4
5   int main()
6   {
7       Pilha P, *topo = NULL;
8       int op;
9       string n, t;
10      do{
11          system("clear");
12          cout << "1 - Inserir dados\n";
13          cout << "2 - Exibir dados\n";
14          cout << "3 - Apagar registro\n";
15          cout << "4 - Esvaziar pilha de dados\n";
16          cout << "5 - Finalizar programa\n";
17          cout << "Informe sua opção: ";
18          cin >> op;
19          switch(op){
20              case 1:
21                  cout << "\nDigite o nome: ";
22                  cin.ignore(); // limpa o buffer
23                  getline(cin,n); // armazena até digitar o enter
24                  cout << "\nDigite o telefone: ";
25                  getline(cin,t);
26                  topo = P.InserirPilha(topo, n, t);
27                  cout << "\nRegistro incluido com sucesso!!\n";
28                  break;
29              case 2:
30                  P.PercorrerPilha(topo);
31                  break;
32              case 3:
33                  if(topo == NULL){
34                      cout << "\nSem registros para deletar!\n";
35                  }else{
36                      topo = P.RemoverPilha(topo);
37                      cout << "\nRegistro deletado!\n";
38                  }
39                  break;
40              case 4:
41                  if(topo == NULL){
42                      cout << "\nSem registros para deletar!\n";
43                  }else{
44                      topo = P.EsvaziarPilha(topo);
45                      cout << "\nPilha vazia!\n";
46                  }
47                  break;
48              case 5:
49                  cout << "\nTchau!!\n";
50                  break;
51              default:
```

```
52                  cout << "\nOpção inválida!\n";
53          }
54          cout << "\nPressione Enter para continuar!! ";
55          cin.ignore().get();
56      }while(op != 5);
57
58      return 0;
59  }
```