

## Exercício 27

Defina uma classe **Instituição** na qual trabalham **peessoas**. Defina as operações de **adição** e **remoção** de um **Aluno** e **Professor** à instituição.

*Melhoramento da solução, com a implementação do construtor de cópia, destrutor e operador afectação, necessários à correcta manipulação da coleção híbrida em casos de duplicação e desalocação da mesma.*

ColecaoHibrida.h

...

## classe Pessoa

```
class Pessoa{
    string nome;
public:
    Pessoa(const string &n):nome(n){}

    virtual void Print(){ cout << nome << endl; }

    virtual bool operator<(const Pessoa &p) const{
        return nome < p.nome;
    }

    virtual Pessoa *Clone(){
        Pessoa *p=new Pessoa(*this);
        return p;
    }
};
```

## classe Aluno

```
class Aluno: public Pessoa{
    int no_mec;
public:
    Aluno(const string &n, int no): Pessoa(n) {no_mec=no;}

    void Print(){
        Pessoa::Print();
        cout << no_mec << endl;
    }

    Pessoa *Clone(){
        Pessoa *p=new Aluno(*this);
        return p;
    }
};
```

## classe Professor

```
class Professor: public Pessoa{
    string categoria;
public:
    Professor(const string &n, const string &c)
        :Pessoa(n), categoria(c){}

    void Print(){
        Pessoa::Print();
        cout << categoria << endl;
    }

    Pessoa *Clone(){
        Pessoa *p=new Professor(*this);
        return p;
    }
};
```

## classe Instituicao

```
#include "ColecaoHibrida.h"
```

```
class Instituicao{
    ColecaoHibrida<Pessoa*> pessoas;
public:
    Instituicao(){}
    Instituicao(const Instituicao &outra){
        copiarPessoasDe(outra);
    }
    Instituicao &operator=(const Instituicao &outra){
        if(this!=&outra){removerPessoas();
                        copiarPessoasDe(outra);
        }
        return(*this);
    }
    ~Instituicao(){removerPessoas();}
    ...
}
```

## classe Instituicao (continuação)

```
void removerPessoas(){
    ColecaoHibrida<Pessoa*>::iterator i;
    for(i=pessoas.begin(); i!=pessoas.end(); i++) delete *i;
    pessoas.clear();
}

void copiarPessoasDe(const Instituicao &outra){
    ColecaoHibrida<Pessoa*>::iterator i;
    for(i=outra.pessoas.begin(); i!=outra.pessoas.end(); i++){
        Pessoa *p=(*i)->Clone();
        pessoas.insert(p);
    }
}

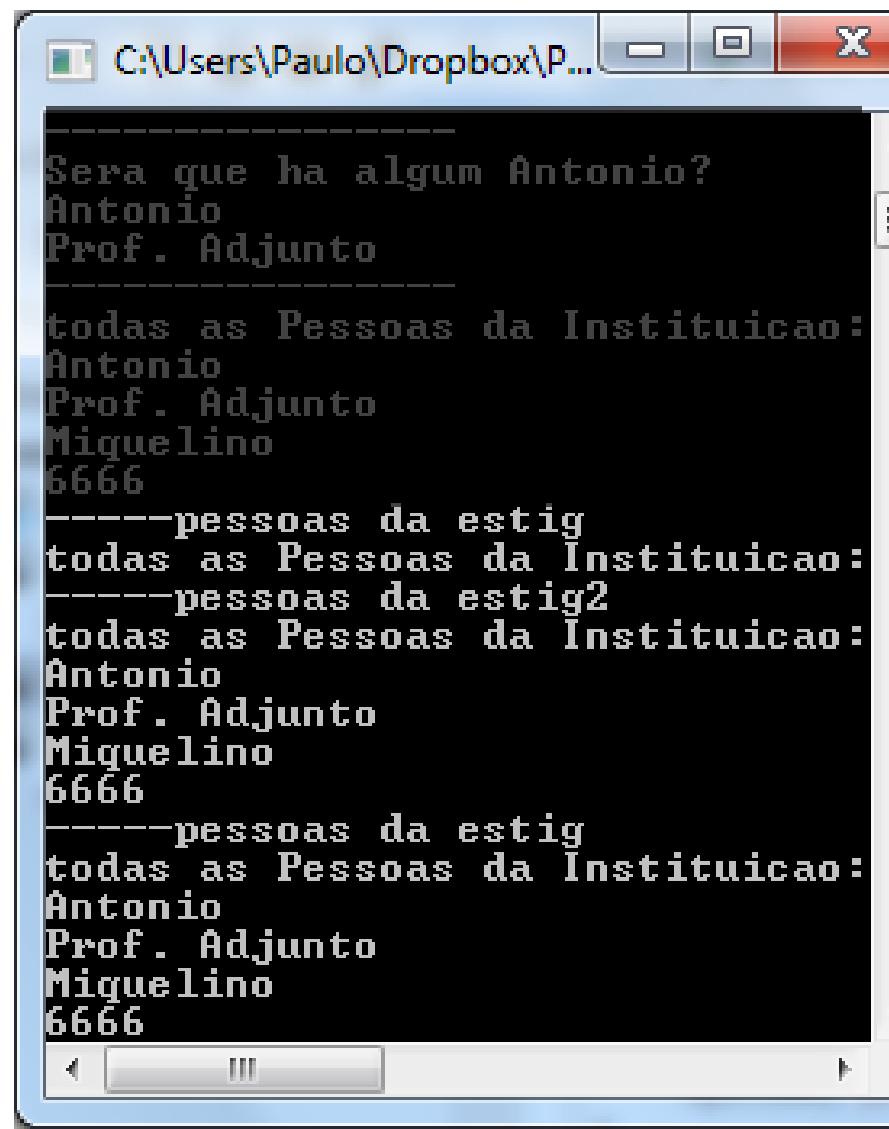
...
};
```

## main

```
#include "ColecaoHibrida.h"

void main(){
    Instituicao estig;
    estig.addAluno("Luis", 5555);
    ...

    Instituicao estig2(estig);
    estig.removerPessoas();
    cout<<"-----pessoas da estig\n";
    estig.PrintPessoas();
    cout<<"-----pessoas da estig2\n";
    estig2.PrintPessoas();
    estig=estig2;
    cout<<"-----pessoas da estig\n";
    estig.PrintPessoas();
}
```



```
-----
Sera que ha algum Antonio?
Antonio
Prof. Adjunto
-----
todas as Pessoas da Instituicao:
Antonio
Prof. Adjunto
Miquelino
6666
-----pessoas da estig
todas as Pessoas da Instituicao:
-----pessoas da estig2
todas as Pessoas da Instituicao:
Antonio
Prof. Adjunto
Miquelino
6666
-----pessoas da estig
todas as Pessoas da Instituicao:
Antonio
Prof. Adjunto
Miquelino
6666
```