Universidade Tecnológica Federal do Paraná
UTFPR – Campus Curitiba

Orientação a Objetos
-
Programação em C++

Grupo de Slides 11 – Parte B: Utilização da lista desacoplada

Prof. Dr. Jean Marcelo SIMÃO

Aluno monitor (em 2019) : Luan Carlos Klein

Melhorando a solução apresentada na solução 11...

Qual o problema dessa solução?

```
#ifndef _PRINCIPAL_H_
#define _PRINCIPAL_H_
#include "Lista.h"
#include "Professor.h"
#include "Universidade.h"
#include "Departamento.h"
#include "Disciplina.h"
#include "Aluno.h"
class Principal
{
   private:
            int cont_idDisc;
            int cont_idDepart;
            int cont_idAluno;

            Universidade UTFPR;
            Universidade Princeton;
            Universidade Cambridge;

            Departamento EletronicaUTFPR;
            Departamento MatematicaUTFPR;
            Departamento FisicaUTFPR;
            Departamento MatematicaPrinceton;
            Departamento FisicaPrinceton;
            Departamento MatematicaCambridge;
            Departamento FisicaCambridge;

            Professor Simao;
            Professor Einstein;
            Professor Newton;
            Disciplina Computacao1_2006;
            Disciplina Introd_Alg_2007;
            Disciplina Computacao2_2007;
            Disciplina Metodos2_2007;

            Aluno AAA;
            Aluno BBB;
            Aluno CCC;
            Aluno DDD;
            Aluno EEE;
            int diaAtual;
            int mesAtual;
            int anoAtual;
            Lista <Universidade>   LUniversidades;
            Lista <Departamento> LDepartamentos;
            Lista <Disciplina>       LDisciplinas;
            Lista <Aluno>            LAlunos;
            Lista <Pessoa>           LPessoas;
public:
            Principal ( );
            void Inicializa();
            void InicializaUnivesidades();
            void InicializaDepartamentos();
            void InicializaProfessores();
            void InicializaAlunos();
            void InicializaDisciplinas();
            void Executar();

            void CalcIdadeProfs();
            void UnivOndeProfsTrabalham();
            void DepOndeProfsTrabalham();
            void ListeDiscDeptos ( );
            void ListeAlunosDisc();

            void CadDisciplina();
            void CadDepartamento();
            void CadUniversidade();
            void CadAluno();

            void GravarTudo();
            void GravarUniversidades();
            void GravarDepartamentos();
            void GravarDisciplinas();
            void GravarAlunos();
            void GravarProfessores();
            void MenuCad();
            void MenuExe();
            void Menu();
};
#endif
```

```
#ifndef _PRINCIPAL_H_
#define _PRINCIPAL_H_
#include "Lista.h"
#include "Professor.h"
#include "Universidade.h"
#include "Departamento.h"
#include "Disciplina.h"
#include "Aluno.h"
class Principal
{
   private:
            int cont_idDisc;
            int cont_idDepart;
            int cont_idAluno;

            Universidade UTFPR;
            Universidade Princeton;
            Universidade Cambridge;

            Departamento EletronicaUTFPR;
            Departamento MatematicaUTFPR;
            Departamento FisicaUTFPR;
            Departamento MatematicaPrinceton;
            Departamento FisicaPrinceton;
            Departamento MatematicaCambridge;
            Departamento FisicaCambridge;

            Professor Simao;
            Professor Einstein;
            Professor Newton;
            Disciplina Computacao1_2006;
            Disciplina Introd_Alg_2007;
            Disciplina Computacao2_2007;
            Disciplina Metodos2_2007;

            Aluno AAA;
            Aluno BBB;
            Aluno CCC;
            Aluno DDD;
            Aluno EEE;

            int diaAtual;
            int mesAtual;
            int anoAtual;
            Lista <Universidade>   LUniversidades;
            Lista <Departamento>  LDepartamentos;
            Lista <Disciplina>       LDisciplinas;
            Lista <Aluno>             LAlunos;
            Lista <Pessoa>           LPessoas;
public:
            Principal ( );
            void Inicializa();
            void InicializaUnivesidades();
            void InicializaDepartamentos();
            void InicializaProfessores();
            void InicializaAlunos();
            void InicializaDisciplinas();
            void Executar();

            void CalcIdadeProfs();
            void UnivOndeProfsTrabalham();
            void DepOndeProfsTrabalham();
            void ListeDiscDeptos ( );
            void ListeAlunosDisc();

            void CadDisciplina();
            void CadDepartamento();
            void CadUniversidade();
            void CadAluno();

            void GravarTudo();
            void GravarUniversidades();
            void GravarDepartamentos();
            void GravarDisciplinas();
            void GravarAlunos();
            void GravarProfessores();
            void MenuCad();
            void MenuExe();
            void Menu();
};
#endif
```

Como visto no Grupo de Slides 10 B, a melhor solução é especificar uma lista para cada objeto diferente, visando desacoplamento.

```
#ifndef _PRINCIPAL_H_
#define _PRINCIPAL_H_
#include "Lista.h"
#include "Professor.h"
#include "Universidade.h"
#include "Departamento.h"
#include "Disciplina.h"
#include "Aluno.h"
class Principal
{
   private:
            int cont_idDisc;
            int cont_idDepart;
            int cont_idAluno;

            Universidade UTFPR;
            Universidade Princeton;
            Universidade Cambridge;

            Departamento EletronicaUTFPR;
            Departamento MatematicaUTFPR;
            Departamento FisicaUTFPR;
            Departamento MatematicaPrinceton;
            Departamento FisicaPrinceton;
            Departamento MatematicaCambridge;
            Departamento FisicaCambridge;

            Professor Simao;
            Professor Einstein;
            Professor Newton;
            Disciplina Computacao1_2006;
            Disciplina Introd_Alg_2007;
            Disciplina Computacao2_2007;
            Disciplina Metodos2_2007;

            Aluno AAA;
            Aluno BBB;
            Aluno CCC;
            Aluno DDD;
            Aluno EEE;

            int diaAtual;
            int mesAtual;
            int anoAtual;
            Lista <Universidade>   LUniversidades;
            Lista <Departamento>   LDepartamentos;
            Lista <Disciplina>     LDisciplinas;
            Lista <Aluno>          LAlunos;
            Lista <Pessoa>         LPessoas;
public:
            Principal ( );
            void Inicializa();
            void InicializaUnivesidades();
            void InicializaDepartamentos();
            void InicializaProfessores();
            void InicializaAlunos();
            void InicializaDisciplinas();
            void Executar();

            void CalcIdadeProfs();
            void UnivOndeProfsTrabalham();
            void DepOndeProfsTrabalham();
            void ListeDiscDeptos ( );
            void ListeAlunosDisc();

            void CadDisciplina();
            void CadDepartamento();
            void CadUniversidade();
            void CadAluno();

            void GravarTudo();
            void GravarUniversidades();
            void GravarDepartamentos();
            void GravarDisciplinas();
            void GravarAlunos();
            void GravarProfessores();
            void MenuCad();
            void MenuExe();
            void Menu();
};
#endif
```

Como visto no Grupo de Slides 10 B, a melhor solução é especificar uma lista para cada objeto diferente, visando desacoplamento.

```
#ifndef _PRINCIPAL_H_
#define _PRINCIPAL_H_
#include "Lista.h"
#include "Professor.h"
#include "Universidade.h"
#include "Departamento.h"
#include "Disciplina.h"
#include "Aluno.h"
class Principal
{
   private:
            int cont_idDisc;
            int cont_idDepart;
            int cont_idAluno;

            Universidade UTFPR;
            Universidade Princeton;
            Universidade Cambridge;

            Departamento EletronicaUTFPR;
            Departamento MatematicaUTFPR;
            Departamento FisicaUTFPR;
            Departamento MatematicaPrinceton;
            Departamento FisicaPrinceton;
            Departamento MatematicaCambridge;
            Departamento FisicaCambridge;

            Professor Simao;
            Professor Einstein;
            Professor Newton;
            Disciplina Computacao1_2006;
            Disciplina Introd_Alg_2007;
            Disciplina Computacao2_2007;
            Disciplina Metodos2_2007;

            Aluno AAA;
            Aluno BBB;
            Aluno CCC;
            Aluno DDD;
            Aluno EEE;
            int diaAtual;
            int mesAtual;
            int anoAtual;
            ListaUniversidade   LUniversidades;
            ListaDepartamento LDepartamentos;
            ListaDisciplina      LDisciplinas;
            ListaAluno           LAlunos;
            ListaPessoa          LPessoas;
public:
            Principal ( );
            void Inicializa();
            void InicializaUnivesidades();
            void InicializaDepartamentos();
            void InicializaProfessores();
            void InicializaAlunos();
            void InicializaDisciplinas();
            void Executar();

            void CalcIdadeProfs();
            void UnivOndeProfsTrabalham();
            void DepOndeProfsTrabalham();
            void ListeDiscDeptos ( );
            void ListeAlunosDisc();

            void CadDisciplina();
            void CadDepartamento();
            void CadUniversidade();
            void CadAluno();

            void GravarTudo();
            void GravarUniversidades();
            void GravarDepartamentos();
            void GravarDisciplinas();
            void GravarAlunos();
            void GravarProfessores();
            void MenuCad();
            void MenuExe();
            void Menu();
};
#endif
```

Dessa maneira, cada lista, trata de maneira distinta seus objetos, de acordo com a especificação e funcionalidade de tal.

```
#include "Elemento.h"
#include "Departamento.h"

class ListaDepartamento
{
   public:
      ListaDepartamento(int nd = 1000, const char* n = "");
      virtual ~ListaDepartamento();
      void setNome (char* n);
      void incluaDepartamento(Departamento* pd);
      void listeDepartamentos();
      void listeDepartamentos2();

   private:

      int cont_dep;
      int numero_dep;
      char nome[150];
      Lista<Departamento> LTDepartamento;
};
```

ListaDepartamento.h

Uma lista específica, que utiliza uma lista template, e molda os métodos de acordo com a necessidade dessa lista.

```
#include "ListaDepartamento.h"

ListaDepartamento::ListaDepartamento ( int nd, const char* n )
{
    numero_dep = nd;
    cont_dep      = 0;
    strcpy (nome, n );
}

ListaDepartamento::~ListaDepartamento()  {  // dtor  }

void ListaDepartamento::setNome (char* n) { strcpy (nome, n); }

void ListaDepartamento::incluaDepartamento ( Departamento* pd )
{
    if (  ( ( cont_dep < numero_dep ) && ( pd != NULL ) ) ||
          ( ( numero_dep == -1 ) && ( pd != NULL ) )  )
    {
       LTDepartamento.incluaInfo(pd, pd->getNome());
    }
    else
    {
          cout << "Departamento não incluído. "
                << "Quantia de deps já lotada em " << numero_dep
                << " departamentos." << endl;
    }
}
```

ListaDepartamento.cpp

```
void ListaDepartamento::listeDepartamentos()
{
    Elemento<Departamento>* pElAux = NULL;

    Departamento* pDeAux = NULL;

    pElAux = LTDepartamento.getPrimeiro();

    while (pElAux != NULL)
    {
        pDeAux = pElAux->getInfo();

        cout << " Departamento "
                << pDeAux->getNome()
                << " da universidade "
                << nome << "." << endl;

        pElAux = pElAux->getProximo();
    }
}

void ListaDepartamento::listeDepartamentos2()
{
     \\ A implementar
}
```

Exercícios Propostos

- Utilizando a solução apresentada, modifique as demais listas, fazendo com que cada grupo de objetos pertinentes tenha sua própria lista (para tratar seus apontamentos), a partir de classes com seus métodos de acordo para tal. Particularmente, faça isto inclusive para lista de apontamentos pessoas.

- Além da classe principal, faça com que as outras classes que têm listas (como departamento que tem uma lista de disciplinas), também utilizem essa solução.