OO – Engenharia Eletrônica

Orientação a Objetos
-
Programação em C++

Slides 7: Arquivos e Persistência de Objetos

Prof. Dr. Jean Marcelo SIMÃO – DAELN / UTFPR

Persistência de Objetos

Arquivos via fluxo

ofstream,ifstream, …

Capítulo 14 do Deitel
(na edição antiga).

Classe Aluno

Lembrando...

```
#ifndef _ALUNO_H_
#define _ALUNO_H_

#include “Pessoa.h“

#include "Departamento.h"

class Aluno : public Pessoa
{
  private:
    int                         RA;

  public:

   . . .

    Aluno      ( int i = -1 );
    ~Aluno     ( );

    void setRA ( int ra );
    int getRA ( );

};

#endif
```

```
#include "stdafx.h"
#include "Aluno.h"

   . . .

Aluno::Aluno ( int i )
{
    id   = i;
    RA = 0;
}

Aluno::~Aluno ( )
{

}

void Aluno::setRA ( int ra )
{
    RA = ra;
}

int Aluno::getRA ( )
{
     return RA;
}
```

```
#ifndef _ALUNO_H_
#define _ALUNO_H_

#include “Pessoa.h“

#include "Departamento.h"

class Aluno : public Pessoa
{
  private:
    int                         RA;

  public:

   . . .

    Aluno      ( int i = -1);
    ~Aluno     ( );

    void setRA ( int ra );
    int getRA ( );

};

#endif
```

```
#include "stdafx.h"
#include "Aluno.h"

   . . .

Aluno::Aluno ( int i ) :
Pessoa ( i )
{
    RA = 0;
}

Aluno::~Aluno ( )
{

}

void Aluno::setRA ( int ra )
{
    RA = ra;
}

int Aluno::getRA ( )
{
     return RA;
}
```

```
#ifndef _ALUNO_H_
#define _ALUNO_H_

#include “Pessoa.h“

#include "Departamento.h"

class Aluno : public Pessoa
{
  private:
    int                         RA;

  public:

   . . .

    Aluno      ( int i = -1);
    ~Aluno     ( );

    void setRA ( int ra );
    int getRA ( );

};

#endif
```

```
#include "stdafx.h"
#include "Aluno.h"

   . . .

Aluno::Aluno ( int i ) :
Pessoa ( i ),
RA ( 0 )
{

}

Aluno::~Aluno ( )
{

}

void Aluno::setRA ( int ra )
{
     RA = ra;
}

int Aluno::getRA ( )
{
      return RA;
}
```

Classe Principal

Mudanças...

```
#ifndef _PRINCIPAL_H_
#define _PRINCIPAL_H_
#include “Professor.h"
#include "ListaUniversidades.h"
#include "ListaDepartamentos.h"
#include "ListaDisciplinas.h"
#include "Aluno.h"
class Principal
{  private:
   // A T E N Ç Ã O !!!
   // A ORDEM DA DECLARAÇÃO DOS OBJETOS/VARIÁVEIS
   // (AGREGADOS) NUMA CLASSE AFETA
   // A ORDEM DA CHAMADA(OU DA EXECUÇÃO) DE SEUS
   // CONTRUTORES A PARTIR DO CONSTRUTOR
   // DESTA CLASSE AGREGADORA. NESTE CASO,
   // A CLASSE QUE OS AGREGA É A PRINCIPAL!

   // Contadores para identificadores;
   int cont_idDisc;
   int cont_idDepart;
   int cont_idAluno;
   // Universidades
   Universidade UTFPR;
   Universidade Princeton;
   Universidade Cambridge;
   // Departamentos
   Departamento EletronicaUTFPR;
   Departamento MatematicaUTFPR;
   Departamento FisicaUTFPR;
   Departamento MatematicaPrinceton;
   Departamento FisicaPrinceton;
   Departamento MatematicaCambridge;
   Departamento FisicaCambridge;
   . . .
   // Professores
   Professor      Simao;
   Professor      Einstein;
   Professor      Newton;
   // Disciplinas
   Disciplina     Computacao1_2006;
   Disciplina     Introd_Alg_2007;
   Disciplina     Computacao2_2007;
   Disciplina     Metodos2_2007;

   int            diaAtual, mesAtual, anoAtual;
   // Alunos
   Aluno                  AAA;
   Aluno                  BBB;
   Aluno                  CCC;
   Aluno                  DDD;
   Aluno                  EEE;

   ListaUniversidades     LUniversidades;
   ListaDepartamentos     LDepartamentos;
   ListaDisciplinas       LDisciplinas;
   ListaAlunos            LAlunos;
 public:
   Principal ( );
    void Inicializa();
   . . .
   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 RecuperarTudo ();
   void RecuperarUniversidades ();
   void RecuperarDepartamentos ();
   void RecuperarDisciplinas ();
   void RecuperarAlunos ();
   void RecuperarProfessores ();
   void MenuCad ();
   void MenuExe ();
   void MenuGravar ();
   void MenuRecuperar ();
   void Menu ();
};
#endif
```

```
#include "stdafx.h"
#include "Principal.h"
// A T E N Ç Ã O
// A ordem de chamada dos construtores dos objetos (agredados) nesta classe
// é definida pela sua ordem de declaração na classe Principal e NÃO (“estra-
// nhamente”) pela sua ordem de chamada aqui (a partir do construtor da Principal).
Principal::Principal ( ) :
  // "geradores" de identificação
  cont_idAluno              ( 0 ),
  cont_idDisc               ( 0 ),
  cont_idDepart             ( 0 ),
  // Contrutores dos objetos da Classe Disciplina
  Computacao1_2006          ( cont_idDisc++ ),
  Introd_Alg_2007           ( cont_idDisc++ ),
  Computacao2_2007          ( cont_idDisc++ ),
  Metodos2_2007             ( cont_idDisc++ ),
  // Contrutores dos objetos da Classe Aluno
  AAA                       ( cont_idAluno++ ),
  BBB                       ( cont_idAluno++ ),
  CCC                       ( cont_idAluno++ ),
  DDD                       ( cont_idAluno++ ),
  EEE                       ( cont_idAluno++ ),
  //Contrutores dos Objetos da Classe Departamento
  EletronicaUTFPR           ( cont_idDepart++ ),
  MatematicaUTFPR           ( cont_idDepart++ ),
  FisicaUTFPR               ( cont_idDepart++ ),
  MatematicaPrinceton       ( cont_idDepart++ ),
  FisicaPrinceton           ( cont_idDepart ++),
  MatematicaCambridge       ( cont_idDepart++ ),
  FisicaCambridge           ( cont_idDepart++ )
{
    . . .
    Inicializa ( );
}
```

Chamada aos construtores dos objetos agregados explicitamente na classe Principal... Isto é, dos atributos da Principal que são objetos de alguma outra classe criada no sistema.

```
void Principal::Menu ( )
{
   int op = -1;
   while ( op != 5 )
   {
      system ( "cls" );
      cout   <<        "  Informe sua opção:    "        << endl;
      cout   <<        "  1 - Cadastrar.        "        << endl;
      cout   <<        "  2 - Executar.         "        << endl;
      cout   <<        "  3 - Gravar.           "        << endl;
      cout   <<        "  4 - Recuperar.        "        << endl;
      cout   <<        "  5 – Sair.             "        << endl;
      cin    >>        op;

      switch ( op )
      {
            case 1:    {          MenuCad ();                    }
                       break;
            case 2:    {          MenuExe ();                    }
                       break;
            case 3:    {          MenuGravar ();                 }
                       break;
            case 4:    {          MenuRecuperar ();              }
                       break;
            case 5:    {          cout << " FIM " << endl;       }
                       break;
            default:   {
                                  cout <<     "Opção Inválida - Pressione uma tecla."  << endl;
                                  getchar();
                       }
      }
   }
}
```

```
void Principal::MenuCad ( )
{
   int op = -1;
   while ( op != 5 )
   {
      system ( "cls" );
      cout   << "  Informe sua opção:                "          << endl;
      cout   << "  1 - Cadastrar Disciplina.         "          << endl;
      cout   << "  2 - Cadastrar Departamentos.      "          << endl;
      cout   << "  3 - Cadastrar Universidade.       "          << endl;
      cout   << "  4 - Cadastrar Aluno.              "          << endl;
      cout   << "  5 - Sair.                         "          << endl;
      cin    >> op;

      switch ( op )
      {
            case 1 :    { CadDisciplina ( );                    }
                        break;
            case 2:     { CadDepartamento ( );                  }
                        break;
            case 3:     { CadUniversidade ( );                  }
                        break;
            case 4:     { CadAluno ( );                         }
                        break;
            case 5:     { cout << " FIM " << endl;              }
                        break;
            default:    {
                          cout << " Opção Inválida - Pressione uma tecla. " << endl;
                          getchar ( );
                        }
      }
   }
}
```

```
void Principal::MenuExe ( )
{
   int op = -1;
   while ( op != 5 )
   {
      system ( "cls" );
      cout << "  Informe sua opção:       "              << endl;
      cout << "  1 - Listar Disciplinas.  "              << endl;
      cout << "  2 - Listar Departamentos."              << endl;
      cout << "  3 - Listar Universidade. "              << endl;
      cout << "  4 - Listar Alunos.       "              << endl;
      cout << "  5 – Sair.                "              << endl;
      cin >> op;

      switch ( op )
      {
            case 1:  {  LDisciplinas.listeDisciplinas ();          fflush ( stdin );  getchar(); }
                     break;

            case 2:  { LDepartamentos.listeDepartamentos ();       fflush(stdin);    getchar(); }
                     break;

            case 3:  { LUniversidades.listeUniversidades ();       fflush(stdin);   getchar();  }
                     break;

            case 4:  { LAlunos.listeAlunos ();                     fflush(stdin);   getchar(); }
                     break;

            case 5:  {     cout << " FIM " << endl;   }
                     break;

            default: {      cout << "Opção Inválida - Pressione uma tecla." << endl;
                            getchar();    }
      }
   }
}
```

```
void Principal::MenuGravar ( )
{   int op = -1;
    while (op != 6)
    {
       system ( "cls" );
       cout    << "  Informe sua opção:                "    << endl;
       cout    << "  0 - Gravar Tudo.                  "    << endl;
       cout    << "  1 - Gravar Universidades.         "    << endl;
       cout    << "  2 - Gravar Departamentos.         "    << endl;
       cout    << "  3 - Gravar Disciplinas.           "    << endl;
       cout    << "  4 - Gravar Alunos.                "    << endl;
       cout    << "  5 - Gravar Professores.           "    << endl;
       cout    << "  6 – Sair.                         "    << endl;
       cin     >> op;

       switch ( op )
       {
              case 0:    {    GravarTudo ( );              }
                         break;
              case 1:    {    GravarUniversidades ( );     }
                         break;
              case 2:    {    GravarDepartamentos ( );     }
                         break;
              case 3:    {    GravarDisciplinas ( );       }
                         break;
              case 4:    {    GravarAlunos ( );            }
                         break;
              case 5:    {    GravarProfessores ( );       }
                         break;
              case 6:    {    cout << " FIM " << endl;     }
                         break;
              default:   {     cout << "Opção Inválida - Pressione uma tecla." << endl;
                               getchar(); }
       }
    }
}
```

```
void Principal::MenuRecuperar ( )
{   int op = -1;
    while ( op != 6 )
    {
        system ( "cls" );
        cout    << "  Informe sua opção:                "    << endl;
        cout    << "  0 - Recuperar Tudo.               "    << endl;
        cout    << "  1 - Recuperar Universidades.      "    << endl;
        cout    << "  2 - Recuperar Departamentos.      "    << endl;
        cout    << "  3 - Recuperar Disciplinas.        "    << endl;
        cout    << "  4 - Recuperar Alunos.             "    << endl;
        cout    << "  5 - Recuperar Professores.        "    << endl;
        cout    << "  6 – Sair.                         "    << endl;
        cin     >> op;

        switch ( op )
        {
                case 0:     {       RecuperarTudo ( );                  }
                            break;
                case 1:     {       RecuperarUniversidades ( );         }
                            break;
                case 2:     {       RecuperarDepartamentos ( );         }
                            break;
                case 3:     {       RecuperarDisciplinas ( );           }
                            break;
                case 4:     {       RecuperarAlunos ( );                }
                            break;
                case 5:     {       RecuperarProfessores ( );           }
                            break;
                case 6:     {       cout << " FIM " << endl;            }
                            break;
                default:    { cout      << "Opção Inválida - Pressione uma tecla." << endl;
                              getchar(); }
        }
    }
}
```

```
void Principal::RecuperarAlunos ( )
{
    LAlunos.recupereAlunos ( );
}
```

```
void Principal::GravarAlunos ( )
{
    LAlunos.graveAlunos ( );
}
```

```
void Principal::CadAluno ( )
{
   char     nomeAluno [150];
   int      ra;
   Aluno*   pal;

   cout     <<      "Qual o nome do aluno. "      << endl;
   cin      >>      nomeAluno;

   cout     <<      "Qual o RA do aluno.“         << endl;
   cin      >>      ra;

   pal      = new Aluno ( cont_idAluno );
   cont_idAluno++;

   pal->setNome ( nomeAluno );

   pal->setRA ( ra );

   LAlunos.incluaAluno ( pal );
}
```

```
void Principal::RecuperarAlunos ( )
{
    LAlunos.recupereAlunos ( );
}
```

```
void Principal::GravarAlunos ( )
{
    LAlunos.graveAlunos ( );
}
```

```
void Principal::CadAluno ( )
{
   char     nomeAluno [150];
   int      ra;
   Aluno*   pal;

   cout     <<      "Qual o nome do aluno. "      << endl;
   cin      >>      nomeAluno;

   cout     <<      "Qual o RA do aluno.“         << endl;
   cin      >>      ra;

   pal      = new Aluno ( cont_idAluno++ );

   pal->setNome ( nomeAluno );

   pal->setRA (ra);

   LAlunos.incluaAluno ( pal );
}
```

Persistência dos Objetos da Classe Aluno a partir da LAlunos existentes na Classe Principal

Utilização de Arquivos na
'Orientação' a Fluxos
-
Arquivos Sequenciais
ou de Texto.

Gravando em arquivo

```
void ListaAlunos::graveAlunos ( )
{
   ofstream GravadorAlunos ( "alunos.dat", ios::out );

   if ( !GravadorAlunos )
   {
         cerr << " Arquivo não pode ser aberto " << endl;
         fflush ( stdin );
         getchar( );
         return;
   }

   ElAluno* pauxElAluno;
   pauxElAluno = pElAlunoPrim;

   while ( pauxElAluno != NULL)
   {
         Aluno * pauxAluno;

         pauxAluno = pauxElAluno->getAluno();

         GravadorAlunos << pauxAluno->getId ( ) << ' '
                        << pauxAluno->getRA ( ) << ' '
                        << pauxAluno->getNome ( ) << endl;
          pauxElAluno = pauxElAluno->pProx;
   }

  GravadorAlunos.close ( );
}
```

Recuperando de arquivo

```
void ListaAlunos::recupereAlunos ( )
{
    ifstream RecuperadorAlunos ( "alunos.dat", ios::in );
    if ( !RecuperadorAlunos )
    {
          cerr << " Arquivo não pode ser aberto " << endl;
          fflush ( stdin );
          getchar( );
    }
    limpaLista ( );

    while ( !RecuperadorAlunos.eof ( ) )
    {
          Aluno *        pauxAluno;
          int            id;
          int            RA;
          char           nome [ 150 ] ;

          RecuperadorAlunos  >> id >> RA >> nome;

           if ( 0 != strcmp ( nome, "" ) )
           {
                 pauxAluno = new Aluno ( -1 );
                 pauxAluno->setId ( id );
                 pauxAluno->setRA ( RA );
                 pauxAluno->setNome ( nome );

                 incluaAluno ( pauxAluno );
          }
   }
   RecuperadorAlunos.close ( );
}
```

Apenas destacando o método...

```
void ListaAlunos::graveAlunos ( )
{
   ofstream GravadorAlunos ( "alunos.dat", ios::out );

   if ( !GravadorAlunos )
   {
         cerr << "Arquivo não pode ser aberto" << endl;
         fflush ( stdin );
         getchar ( );
         return;
   }

   ElAluno*          pauxElAluno = NULL;
   Aluno *           pauxAluno = NULL;;

   pauxElAluno =     pElAlunoPrim;

   while ( pauxElAluno != NULL )
   {
        pauxAluno = pauxElAluno->getAluno ( );

          GravadorAlunos << pauxAluno->getId ( )      << ' '
                         << pauxAluno->getRA ( )      << ' '
                         << pauxAluno->getNome ( )    << endl;
         pauxElAluno = pauxElAluno->pProx;
   }
  GravadorAlunos.close ( );
}
```

Apenas destacando o método...

```
void ListaAlunos::recupereAlunos ( )
{
    ifstream RecuperadorAlunos ( "alunos.dat", ios::in );
    if ( ! RecuperadorAlunos )
    {
        cerr << "Arquivo não pode ser aberto" << endl;
        fflush ( stdin );    getchar ( );
        return;
    }
    limpaLista ( );

    while ( !RecuperadorAlunos.eof ( ) )
    {
        Aluno * pauxAluno = NULL;
        int             id;
        int             RA;
        char            nome [ 150 ] ;

         RecuperadorAlunos >> id >> RA >> nome;
         if ( 0 != strcmp ( nome, "" ) )
         {
             pauxAluno = new Aluno ( -1 );
             pauxAluno->setId ( id );
             pauxAluno->setRA ( RA );
             pauxAluno->setNome ( nome );

             incluaAluno ( pauxAluno );
         }
   }
   RecuperadorAlunos.close ();
}
```

A função “eof” pode executar uma vez mais dependendo da formatação do arquivo de texto utilizado ou do sistema operacional, por isso uma solução mais genérica seria:

```
void ListaAlunos::recupereAlunos ( )
{
    . . .
    limpaLista ( );
    Aluno * pauxAluno = NULL;
    int  id;
    int  RA;
    char  nome [ 150 ] ;
    while (RecuperadorAlunos >> id >> RA >> nome; )
    {

        if ( 0 != strcmp ( nome, "" ) )
        {
            pauxAluno = new Aluno ( -1 );
            pauxAluno->setId ( id );
            pauxAluno->setRA ( RA );
            pauxAluno->setNome ( nome );

            incluaAluno ( pauxAluno );
        }
   }
   RecuperadorAlunos.close ();
}
```

O operador de fluxo sobrecarregado ‘ >> ‘ retornará false caso não consiga extrair dados do arquivo, quebrando o loop quando chegar ao fim do arquivo

Exercícios

- Permitir a persistência (i.e. gravação/recuperação) dos dados dos objetos da classe *Disciplina*, *Departamento* etc, tal qual sugere os métodos *menuGravar* e *menuRecuperar* da classe *Principal*. Fazer isto primeiramente sem considerar a persistência das relações entre os objetos e usando arquivos sequencias (i.e. arquivos de texto).

- Permitir, além da persistência dos objetos, a persistência de suas relações também. Por exemplo, ao gravar um objeto disciplina em arquivo, faz-se necessário também outro arquivo para armazenar o identificador de cada objeto *Aluno* relacionado. No momento da recuperação dos objetos, os seus relacionamentos podem então ser recuperados por meio destas informações neste arquivo adicional...

- *Estudar mais sobre arquivos em C++ para, além de arquivos de texto, ver também arquivos binários. Como fonte de estudo, por exemplo, há o capítulo pertinente do livro dos Deteil. Depois deste estudo, poderia-se elaborar uma solução para a persistência dos objetos no sistema estudados por meio de arquivos binários (também chamados de arquivos 'aleatórios' por assim dizer).*