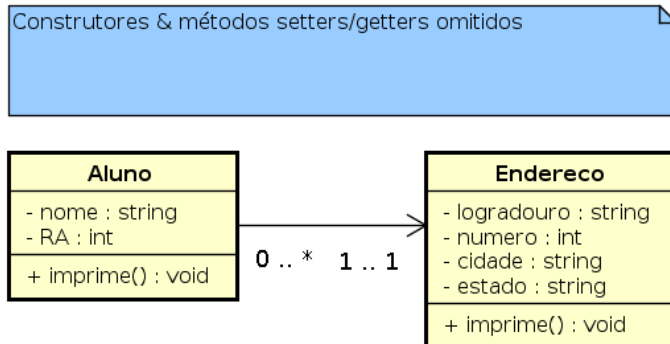


Programação Orientada a Objetos

Prof. Delano M. Beder

Roteiro 08b – Associação entre classes (Aluno e Endereco)

Endereço é obrigatório



1. Crie um projeto (C++) denominado AlunoEndereco

2. Implementação da abstração Endereco

Nova classe C++ denominada Endereco

Dois arquivos: Endereco.h (Cabeçalho) e Endereco.cpp (Código-fonte)

2.1 Arquivo Endereco.h

```
#ifndef ENDERECO_H
#define ENDERECO_H

#include <string>

using namespace std;

class Endereco {
public:
    Endereco(string logradouro, int numero, string cidade, string estado);
    virtual ~Endereco();
    string getLogradouro() const;
    void setLogradouro(string val);
    int getNumero() const;
    void setNumero(int val);
    string getCidade() const;
    void setCidade(string val);
    string getEstado() const;
    void setEstado(string val);
    void imprime() const;
private:
    string logradouro;
    int numero;
    string cidade;
    string estado;
};

#endif // ENDERECO_H
```

2.2 Arquivo `Endereco.cpp`

```
#include "Endereco.h"
#include <iostream>

using namespace std;

// construtor com 4 parâmetros
Endereco::Endereco(string logradouro, int numero, string cidade, string estado) {

    this->logradouro = logradouro;
    this->numero = numero;
    this->cidade = cidade;
    this->estado = estado;
}

// destrutor da classe
Endereco::~Endereco() {
    cout << "Destrutor Endereco (" << logradouro << ", " << numero << ")" << endl;
}

string Endereco::getLogradouro() const {
    return logradouro;
}

void Endereco::setLogradouro(string val) {
    logradouro = val;
}

int Endereco::getNumero() const {
    return numero;
}

void Endereco::setNumero(int val) {
    numero = val;
}

string Endereco::getCidade() const {
    return cidade;
}

void Endereco::setCidade(string val) {
    cidade = val;
}

string Endereco::getEstado() const {
    return estado;
}

void Endereco::setEstado(string val) {
    estado = val;
}

void Endereco::imprime() const {
    cout << "Logradouro: " << this->getLogradouro() << endl;
    cout << "Numero: " << this->getNumero() << endl;
    cout << "Cidade: " << this->getCidade() << endl;
    cout << "Estado: " << this->getEstado() << endl;
}
```

3. Implementação da abstração Aluno

Nova classe C++ denominada Aluno

Dois arquivos: Aluno.h (Cabeçalho) e Aluno.cpp (Código-fonte)

3.1 Arquivo Aluno.h

Utilizando referência para a classe Endereco

```
#ifndef ALUNO_H
#define ALUNO_H

#include "Endereco.h"
#include <string>

using namespace std;

class Aluno {
public:
    Aluno(string nome, int RA);
    Aluno(string nome, int RA, Endereco& endereco);
    virtual ~Aluno();
    string getNome() const;
    void setNome(string val);
    int getRA() const;
    void setRA(int val);
    Endereco& getEndereco() const;
    void setEndereco(Endereco& val);
    void imprime() const;
private:
    string nome;
    int RA;
    Endereco& endereco;
};

#endif // ALUNO_H
```

3.2 Arquivo Aluno.cpp

```
#include "Aluno.h"
#include <iostream>

using namespace std;

// construtor com 3 parâmetros
Aluno::Aluno(string nome, int RA, Endereco& endereco) : nome(nome), RA(RA),
endereco(endereco) {
}

// destrutor da classe
Aluno::~Aluno() {
    cout << "Destrutor Aluno (" << nome << ", " << RA << ")" << endl;
}

string Aluno::getNome() const {
    return nome;
}

void Aluno::setNome(string val) {
    nome = val;
}

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

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

Endereco& Aluno::getEndereco() const {
    return endereco;
}

void Aluno::setEndereco(Endereco& val) {
    endereco = val;
}
```

```

void Aluno::imprime() const {
    cout << "-----" << endl;
    cout << "Nome: " << this->getNome() << endl;
    cout << "RA: " << this->getRA() << endl;
    cout << "<Endereco>" << endl;
    this->getEndereco().imprime();
}

```

4. Arquivo main.cpp

```

#include "Aluno.h"
#include <iostream>

using namespace std;

int main() {

    // Ponteiro para objeto (instância de classe)
    Endereco* end1;
    // Construtor que seta os valores de todos os atributos
    end1 = new Endereco("Rua da Saudade", 10, "São Carlos", "São Paulo");

    // Criação de objeto estático.
    // Construtor que seta os valores de todos os atributos
    Endereco end2("Rua das Bandeiras", 1000, "São Carlos", "São Paulo");

    // Construtor com 3 parâmetros
    Aluno aluno1("Fulano", 12345, *end1);

    // Ponteiro para objeto (instância de classe)
    // Construtor que seta os valores de todos os atributos
    Aluno* aluno2 = new Aluno("Sincrano", 54321, end2);

    // imprime as informações do aluno1
    aluno1.imprime();

    // imprime as informações do aluno2
    aluno2->imprime();

    cout << endl;

    cout << "(Destrutor) - Explicitos (delete)" << endl;

    delete end1;
    delete aluno2;

    cout << "(Destrutor) - Implicitos (fim de função)" << endl;
}

```

5. Compile e execute (verifique a saída impressa)

6. Fim