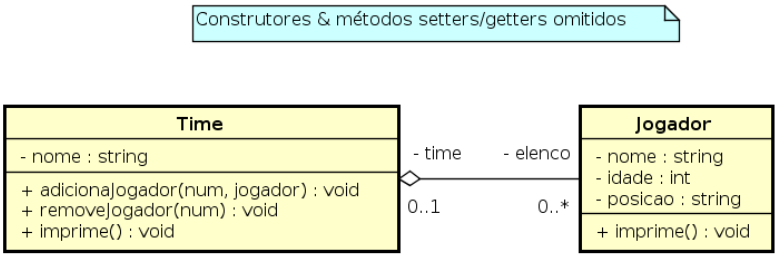
Programação Orientada a Objetos

Prof. Delano M. Beder

Roteiro 09 – Agregação entre classes (Time e Jogador)



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

2. Implementação da abstração Jogador

Nova classe C++ denominada Jogador

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

2.1 Arquivo Jogador.h

```
#ifndef JOGADOR_H
#define JOGADOR_H

#include <string>
using namespace std;

class Time;

class Jogador {
public:
  Jogador(string nome, int idade, string posicao);
  virtual ~Jogador();
  string getNome() const;
  void setNome(string nome);
  int getIdade() const;
  void setIdade(int idade);
  string getPosicao() const;
  void setPosicao(string posicao);
  Time* getTime() const;
  void setTime(Time* time);
  void imprime() const;
private:
  string nome;
  int idade;
  string posicao;
  Time* time;
};

#endif /* JOGADOR_H */
```

2.2 Arquivo Jogador.cpp

```
#include "Jogador.h"
#include "Time.h"

#include <iostream>
using namespace std;

Jogador::Jogador(string nome, int idade, string posicao) :
nome(nome), idade(idade), posicao(posicao) {
  cout << "Criando jogador " << nome << endl;
}

Jogador::~Jogador() {
  cout << "Destruindo jogador " << nome << endl;
}

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

void Jogador::setNome(string nome) {
  this->nome = nome;
}

int Jogador::getIdade() const {
  return idade;
}

void Jogador::setIdade(int idade) {
  this->idade = idade;
}

string Jogador::getPosicao() const {
  return posicao;
}

void Jogador::setPosicao(string posicao) {
  this->posicao = posicao;
}

Time* Jogador::getTime() const {
  return time;
}

void Jogador::setTime(Time* time) {
  this->time = time;
}

void Jogador::imprime() const {
  cout << nome << ", " << posicao << ", " << idade << " anos";
  if (time != NULL) {
    cout << " (Joga no " << time->getNome() << ")";
  } else {
    cout << " (Aposentado/Desempregado)";
  }
  cout << endl;
}
```

3. Implementação da abstração Time

Nova classe C++ denominada Time

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

3.1 Arquivo Time.h

Utilizando vetor de ponteiros para a classe Jogador

```
#ifndef TIME_H
#define TIME_H

#include "Jogador.h"

class Time {
public:
  Time(string nome);
  virtual ~Time();
  string getNome() const;
  void setNome(string nome);
  void adicionaJogador(int numero, Jogador* jogador);
  void removeJogador(int numero);
  void imprime() const;
private:
  string nome;
  Jogador** elenco; // alocação dinâmica (esse array pode mudar de tamanho)
  // Jogador* elenco[22]; // alocação estática (um array de 22 posições fixo)
};

#endif /* TIME_H */
```

3.2 Arquivo Time.cpp

```
#include "Time.h"
#include <iostream>
#include <cstring>
using namespace std;

Time::Time(string nome) :
nome(nome) {
  cout << "Criando time " << nome << endl;
  elenco = new Jogador*[22]; // apenas quando utiliza-se alocacao dinamica
  memset(elenco, 0, 22 * sizeof(Jogador*)); // inicializando com NULL (0) o array
}

Time::~Time() {
  cout << "Destruindo time " << nome << endl;
}

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

void Time::setNome(string nome) {
  this->nome = nome;
}

void Time::adicionaJogador(int numero, Jogador* jogador) {
  elenco[numero - 1] = jogador;
  jogador->setTime(this);
}

void Time::removeJogador(int numero) {
  elenco[numero - 1]->setTime(NULL);
  elenco[numero - 1] = NULL;
}

void Time::imprime() const {
  cout << "Elenco do " << nome << endl;
  for (int i = 0; i < 22; i++) {
    if (elenco[i] != NULL) {
      cout << "(" << (i + 1) << ") ";
      elenco[i]->imprime();
    }
  }
}
```

4. Arquivo main.cpp

```
#include "Time.h"
#include <iostream>
using namespace std;

int main() {

  Jogador *prass = new Jogador("Fernando Prass", 40, "Goleiro");
  Jogador *guerra = new Jogador("Alejando Guerra", 33, "Meia");
  Jogador *borja = new Jogador("Miguel Borja", 25, "Atacante");

  Time *time = new Time("Palmeiras");

  time->adicionaJogador(1, prass);
  time->adicionaJogador(10, guerra);
  time->adicionaJogador(9, borja);

  cout << endl;

  time->imprime();

  cout << endl;

  time->removeJogador(10);

  time->imprime();

  cout << endl;

  guerra->imprime();

  cout << endl;

  delete time;

  delete borja;
  delete guerra;
  delete prass;

  return 0;
}
```

5. Compile e execute (verifique a saída impressa)

6. Fim