



Programação Orientada a Objetos

Introdução à P.O.O.: paradigma, abstração, objetos, classes e instanciação

Ely – ely.miranda@ifpi.edu.br

1

Paradigma de programação

- *É um modelo, padrão ou estilo de programação suportado por linguagens que agrupam certas características comuns.*

2

2

Paradigma de programação

- Foco em uma forma de pensar e resolver problemas.
- Ex:
 - Paradigma imperativo: os principais elementos são funções e módulos;
 - Paradigma orientado a objetos: os principais elementos são objetos.

ely.miranda@ifpi.edu.br

3

3

Motivação

- Por que estudar um paradigma novo:
 - Desenvolver a habilidade de aprender novas linguagens;
 - Aprofundar conceitos de algumas linguagens;
 - Compreender que linguagens são mais adequadas para alguns problemas.

4

4

Motivação

- Por que estudar um paradigma novo:
 - Habilidade maior de projetar novas linguagens;
 - Abstrair melhor problemas para o ambiente computacional.

5

5

Abstração

- Definir apenas os elementos essenciais para um sistema;
- Definir níveis de complexidade dos elementos;
- Ignorar aspectos irrelevantes relacionados à resolução do problema.



6

6

Abstração

Example for Java Abstraction



Owner
<ul style="list-style-type: none"> • Car Description • Service History • Petrol Mileage History

Registration
<ul style="list-style-type: none"> • Vehicle Identification Number • License plate • Current Owner • Tax due, date

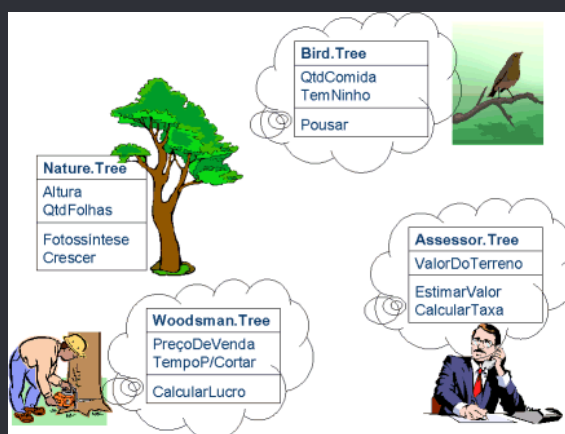
Garage
<ul style="list-style-type: none"> • License plate • Work Description • Billing Info • Owner

ely.miranda@ifpi.edu.br

7

7

Abstração



ely.miranda@ifpi.edu.br

8

8

Abstração

- Dada uma instituição bancária tradicional:
 - A instituição Banco possui:
 - Estrutura física;
 - Localização;
 - Agências, funcionários...;



9

9

Abstração

- Dada uma instituição bancária tradicional:
 - Pessoas com contas em um banco possuem:
 - Nome;
 - Altura;
 - cor dos olhos;
 - CPF...



10

10

Abstração

- Se precisássemos criar um sistema bancário, precisaríamos abstrair detalhes;
- Abstrair é simplificar;

11

11

Abstração

- Abstrair é transformar uma situação em algo computacional;
- Poucos dados dos slides anterior seriam úteis se quiséssemos controlar operações financeiras.

12

12

Abstração

- Uma possível abstração/simplificação para uma agência bancária:
 - Um “array”/conjunto de contas cada uma com:
 - Numero: inteiro;
 - CPF do titular: número;
 - Saldo: real;
 - Operações/funções de crédito e débito;
 - Em C, tal array teria “structs” com os dados e haveria funções implementando as operações.

13

13

Paradigmas e abstração

- Um paradigma de programação também ajuda a abstrair problemas:
 - Simplifica elementos do mundo real;
 - Transforma-os em analogias/peças aplicáveis a algoritmos e soluções.

14

14

Paradigmas e abstração

- Um paradigma de programação também ajuda a abstrair problemas:
 - Transforma-os em analogias/peças aplicáveis a algoritmos e soluções:
 - Em análise: modelos visuais, fluxogramas, diagramas e etc;
 - Em programação: variáveis, ciclos, condicionais, expressões (valor, tipo), entrada e saída.



15

15

Diversidade de linguagens

- C, C++, Basic, COBOL, Lisp, Haskell, Modula-2, Oberon, Prolog, Java, C#, Pascal, PL/1, Ada, Smalltalk, Simula, Algol, Eiffel, Fortran, ASM, Scheme, CLOS, Maude, Python, Glass, Ruby, JavaScript, TypeScript, Perl, Lua, Groovy, SQL, Erlang e etc*

16

16

Diversidade de linguagens

- Por que tantas?
 - Propósitos Diferentes;
 - Avanços Tecnológicos;
 - Interesses comerciais;
 - Cultura e background científico;
 - Ciência da computação relativamente nova.

17

17

Alguns paradigmas

- Imperativo/procedural: C, Pascal, COBOL...
- Orientado a Objetos: C++, Java, Ruby, TypeScript...
- Funcional: Haskell, F#
- Lógico: Prolog, LISP...

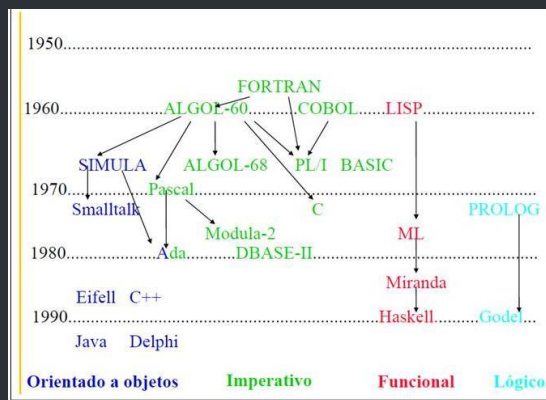
Tendência: linguagens com recursos de mais de um paradigma.

18

18

Alguns paradigmas

- Uma linha do tempo resumida:



19

19

Alguns paradigmas

- Links:
 - <https://dev.to/viebel/data-oriented-programming-a-link-in-the-chain-of-programming-paradigms-3aa7>
 - <https://www.csd.uoc.gr/~hy252/html/Lectures2012/CS252Intro12.pdf>
 - <https://blog.acolyer.org/2019/01/25/programming-paradigms-for-dummies-what-every-programmer-should-know/>
 - <https://digitalfellows.commons.gc.cuny.edu/2018/03/12/an-introduction-to-programming-paradigms/>

ely.miranda@ifpi.edu.br

20

20

Orientação a Objetos

- Paradigma que envolve várias etapas de desenvolvimento de software, dentre elas:
 - Projeto;
 - Análise;
 - Programação; ←
- Possui relevante aceitação comercial e acadêmica.

21

21

Orientação a Objetos

- Principais elementos:
 - Classes e Objetos;
 - Herança;
 - Encapsulamento;
 - Polimorfismo.

22

22

Orientação a Objetos

- Parte do princípio que o mundo é formado por elementos (objetos) que interagem entre si;
- Tudo relacionado ao paradigma são objetos:

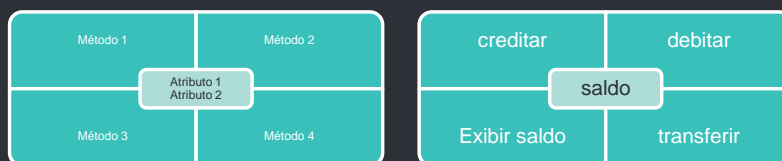


23

23

Orientação a Objetos

- Tudo relacionado ao paradigma são objetos:
 - Possuem características/estado: variáveis conhecidas como atributos;
 - Possuem comportamentos: funções denominadas métodos;
 - Os métodos alteram os valores dos atributos.



24

24

Objetos

- São analogias a elementos do mundo real:
 - Pessoa, Motor, Lâmpada, Cartão de crédito, Processo, Conta, Produto, Venda...
- Possuem **características** e **comportamentos**:
 - Pessoa: **cor**, **altura**, **CPE**, **contas**, **comprar**, **vender**, **postar**, **comentar**;
 - Blog: **postagens**, **nome**, **URL**, **exibir postagens**, **excluir comentários**, **pesquisar**;

25

25

Objetos

- São analogias a elementos do mundo real:
 - Pessoa, Motor, Lâmpada, Cartão de crédito, Processo, Conta, Produto, Venda...
- Possuem **características** e **comportamentos**:
 - Conta bancária: **saldo**, **titular**, **creditar**, **debitar**, **transferir**;
 - Jogador: **número**, **nome**, **velocidade**, **nível de cansaço**, **chutar**, **correr**, **lançar**, **efetuar passe**.

26

26

Definindo Objetos

- Objetos: Substantivos
 - Atributos:
 - Características relacionadas aos substantivos;
 - Métodos:
 - Verbos que representam comportamentos e ações dos objetos.

27

27

Atributos

- Características e propriedades que os objetos possuem;
- Exemplos:
 - Pessoa: cor, altura, CPF, contar;
 - Blog: postagens, nome, URL;
 - Conta bancária: número, saldo, titular;
 - Jogador: número, nome, velocidade, nível de cansaço;
- Preferencialmente devem ser alterados e lidos apenas por métodos.

28

28

Atributos

- Definem ainda o estado dos objetos;
 - São definidos em linguagens de programação como tipos inteiros, arrays, reais, strings...

29

29

Métodos

- Comportamentos de um objeto ou ações que um objeto pode realizar;
- Exemplos:
 - Pessoa: comprar, vender, postar, comentar;
 - Blog: exibir postagens, excluir comentários, pesquisar;
- São implementados através conjuntos de instruções (semelhantes às funções).

30

30

Métodos

- Comportamentos de um objeto ou ações que um objeto pode realizar;
- Exemplos:
 - Conta **bancária**: **creditar**, **debitar**, **transferir**;
 - Jogador: **chutar**, **correr**, **lançar**, **efetuar passe**;
- São implementados através conjuntos de instruções (semelhantes às funções).

31

31

Métodos

- Métodos podem executar ações sobre outros objetos;
- Devem ser os responsáveis pela alteração dos atributos.

32

32

Vantagens

- Objetos são elementos padronizados e podem ser reutilizados;
 - Analogia com o mundo industrial:
 - Produção em larga escala de peças;
 - Peças padrões são mais baratas;
 - São mais testadas e consequentemente mais confiáveis;
 - Possuem facilidades de manutenção e substituição.

33

33

Vantagens

- Modularidade: cada objeto é um módulo;
- Bibliotecas e frameworks: Agrupamento de objetos relacionados resolvendo problemas maiores;
- Reutilização: objetos usados em diferentes contextos/sistemas.

34

34

Vantagens

- Extensão: objetos simples podem ser estendidos através de herança;
- Confiabilidade: possibilidade de criar testes simples e complexos.

Resumindo: menores custos de desenvolvimento.

35

35

Desvantagens

- Novos conceitos;
- Maior curva de aprendizado;
- Programadores “antigos” possuem vícios de outras linguagens.

36

36

Desvantagens

- Aumento da complexidade de software:
 - Necessidade de ferramentas que deem mais produtividade;
 - Ambientes de desenvolvimento não acompanham tão rápido as “novidades”.

37

37

Classes

- Enquadramento dos objetos em categorias conforme atributos e métodos;
- Diz-se que uma classe é uma “matriz/origem” de objetos;
- São modelos a partir dos quais os objetos são criados (instanciados).

38

38

Classes

- Analogias:
 - Planta (classe) e casa construída (objeto);
 - Projeto (classe) e execução (objeto);
 - Receita (classe) e bolo (objeto).

39

39

Classes x Objetos

- Classes são modelos, objetos são classes em execução/memória (instanciadas);
- Uma classe está para um objeto, assim como:
 - Uma receita está para uma torta;
 - Uma planta está para uma casa.

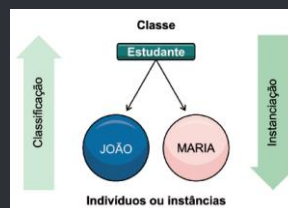
Classe Pessoa	Objeto Pessoa
Nome: Texto; Data de Nascimento: Data; Altura: Número;	Nome: Cláudio; Data de Nascimento: 20/05/1978; Altura: 1.6

40

40

Classificação x Instanciação

- Classificar:
 - Definir classes, ou seja, agrupar objetos com atributos e métodos semelhantes;
- Instanciar:
 - Criar objetos a partir de classes.

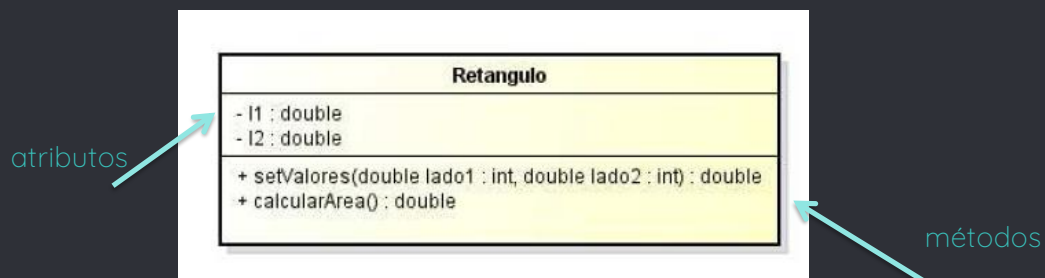


41

41

Classes em UML

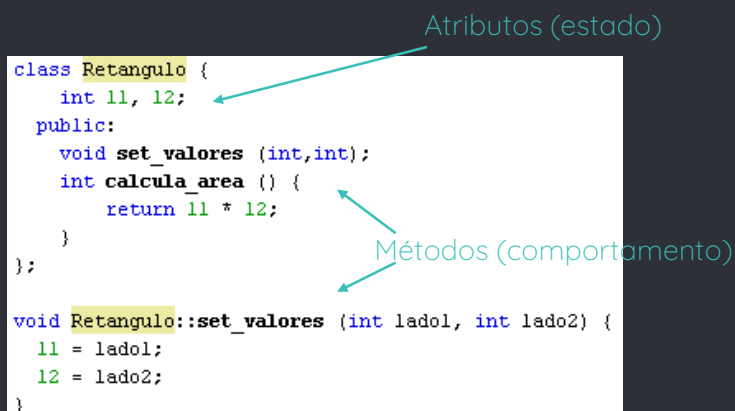
- Unified Modeling Language.



42

42

Exemplo de Classe em C++



Atributos (estado)

```
class Retangulo {
    int l1, l2;
public:
    void set_valores (int,int);
    int calcula_area () {
        return l1 * l2;
    }
};

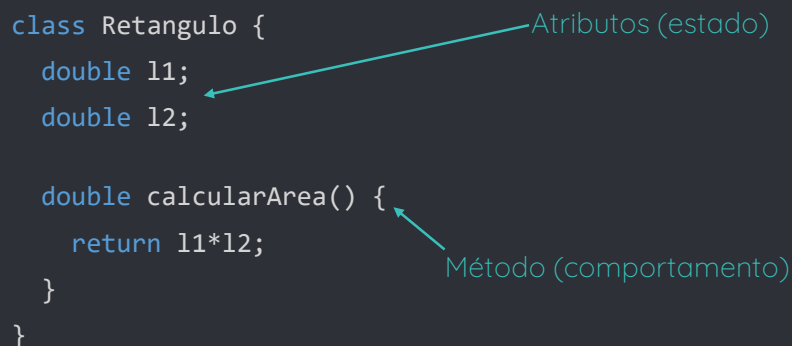
void Retangulo::set_valores (int lado1, int lado2) {
    l1 = lado1;
    l2 = lado2;
}
```

Métodos (comportamento)

43

43

Exemplo de Classe em Java



Atributos (estado)

```
class Retangulo {
    double l1;
    double l2;

    double calcularArea() {
        return l1*l2;
    }
}
```

Método (comportamento)

44

44

Exemplo de Classe em TypeScript

```
class Retangulo {  
  l1: number = 0;  
  l2: number = 0;  
  
  calcularArea(): number {  
    return this.l1 * this.l2;  
  }  
}
```

Atributos (estado)

Método (comportamento)

45

45

Instanciação de objetos

- Para criarmos um objeto, devemos realizar uma instanciação;
- Instanciar um objeto é o equivalente a:
 - alocar uma área de memória;
 - atribuímos a uma variável o endereço dessa área.

46

46

Instanciação de objetos

- Dizemos que um variável é uma referência para um objeto;
- Instanciamos um objeto a partir do nome de sua classe e usando o operador new.

47

47

Instanciação de objetos TypeScript

- Ex1:

```
let retangulo : Retangulo;  
retangulo = new Retangulo();
```

- Ex2:

```
retangulo : Retangulo = new Retangulo();
```

objeto

classe

operador

construtor

48

48

Instanciação de objetos TypeScript

- Utilizando a classe Retangulo:

```
let retangulo : Retangulo;  
retangulo = new Retangulo();  
retangulo.l1 = 10;  
retangulo.l2 = 20;  
  
console.log(retangulo.calcularArea());
```

Atributos e métodos de uma classe são acessados através de um ponto "."

ely.miranda@ifpi.edu.br

49

49



Programação Orientada a Objetos

Introdução à P.O.O.: paradigma, abstração, objetos, classes e instanciação

Ely – ely.miranda@ifpi.edu.br

50