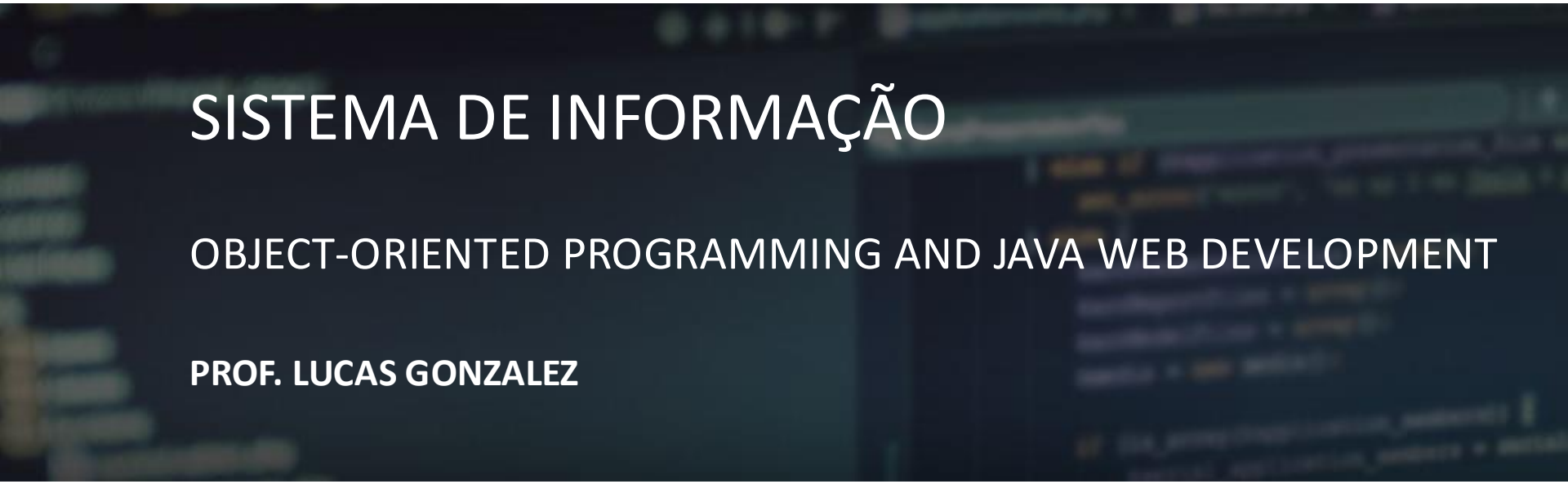
FIAP

FIAP GRADUAÇÃO



SISTEMA DE INFORMAÇÃO

OBJECT-ORIENTED PROGRAMMING AND JAVA WEB DEVELOPMENT

PROF. LUCAS GONZALEZ

CONTEÚDO PROGRAMÁTICO

- Paradigmas de Programação
- Evolução
- Conceitos de Orientação a Objetos
  - Classe
  - Objeto
  - Herança
  - Polimorfismo
  - Interface

Prof. LUCAS GONZALEZ

CONTEÚDO PROGRAMÁTICO

- Estudo e Implementação de Classes e Objetos
  - Codificação de classes
  - Instanciação de classes (criação de objetos)
  - Método construtor
  - Modificadores de acesso (public e private)
  - Objetos como parâmetro de métodos
  - Métodos que retornam objetos

CONTEÚDO PROGRAMÁTICO

- Estudo da Herança no Java
  - Implementação
  - Modificador de acesso protected
- Estudo do Polimorfismo e Interfaces
  - Implementação
  - Classes Abstratas
  - Sobreposição de métodos
- Tratamento de Exceções

CONTEÚDO PROGRAMÁTICO

- Tipos Genéricos em Java (classes e tipos parametrizados)
  - Conceito e aplicações
  - Classes e métodos genéricos
- Interfaces Gráficas com o Usuário
  - Componentes swing
- Acesso a Banco de Dados

BIBLIOGRAFIA BÁSICA

- SCHILDT, H. **Java. A referência completa.** 1ª edição, Editora Alta Books, 2014.

- MANZANO, J. A. N. G.; COSTA, R. A. JR. **Java 8. Programação de Computadores.** 1ª edição, Editora Érica, 2014.

- DEITEL, H. M., DEITEL, P. J. **JAVA como programar**. 10ª edição, São Paulo: Pearson Education do Brasil, 2017.

BIBLIOGRAFIA COMPLEMENTAR

- MEDEIROS, Ernan. **Desenvolvendo Software Com Uml 2.0 Definitivo**. São Paulo, Pearson Makron Books, 2004.

- BARNES, David J.; KOLLING, Michael. **Programação Orientada a Objetos com Java**: uma introdução prática usando o BlueJ - 4ª edição, São Paulo: Pearson, 2009.

- LEE, V.; SCHNEIDER, H.; SCHELL, R. **Aplicações Móveis**, Pearson, 2014.

- PUGA, Sandra, GOMES, RISSETTI, Gerson. **Lógica de Programação e Estrutura de Dados com Aplicações em Java**. 2ª edição, São Paulo: Pearson Education do Brasil, 2013.

- ELMASRI, Ramez; NAVATHE, Shamkant B. **Sistemas de Bancos de Dados**. 6ª ed. Addison Wesley, 2010.

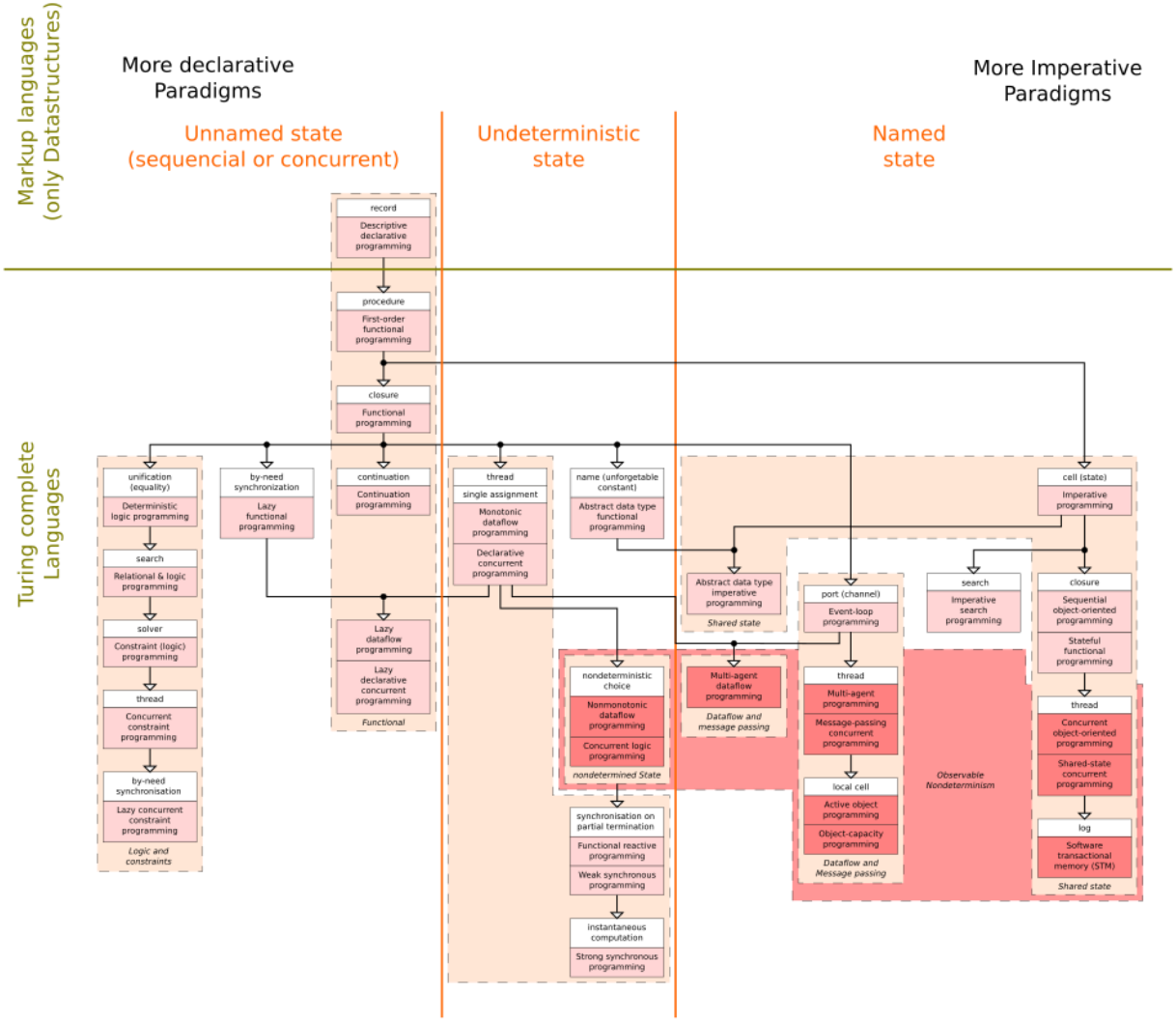
DATAS

- Checkpoint 1: 01/09/2025
- Checkpoint 2: 29/09/2025
- Checkpoint 3: 27/10/2025

PARADIGMAS DE PROGRAMAÇÃO

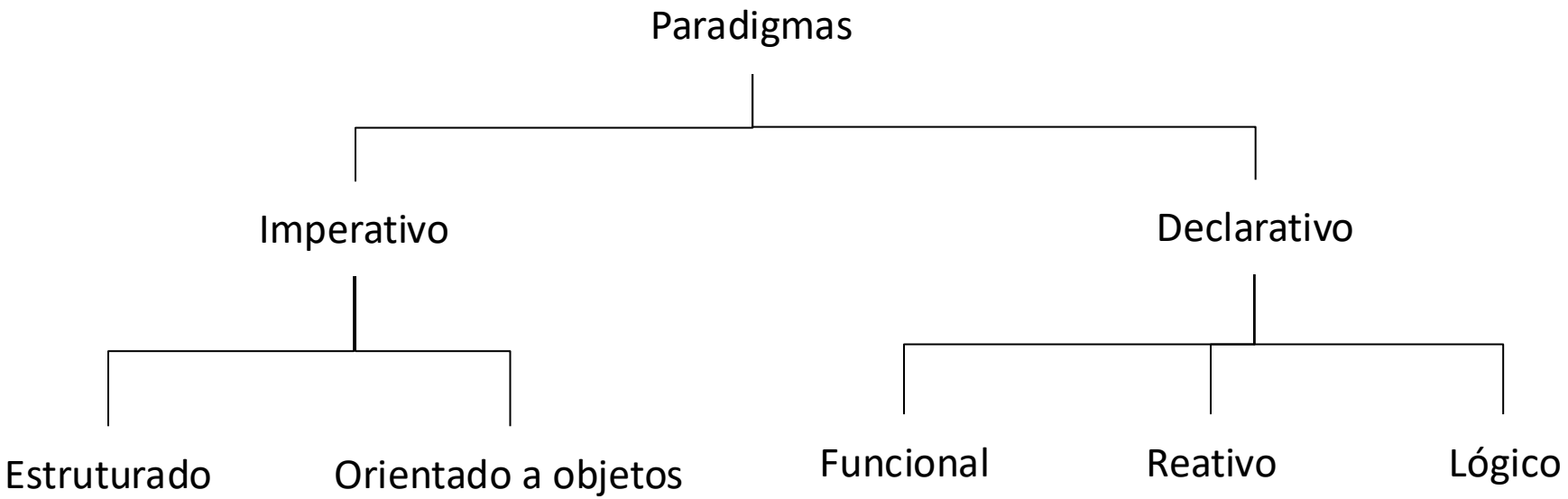
- Paradigma vem do grego *paradeigma*, que significa “padrão, exemplo, modelo”;
- Um paradigma fornece e determina a visão que o programador possui sobre a estruturação e execução do programa;

PARADIGMAS DE PROGRAMAÇÃO



https://webperso.info.ucl.ac.be/~pvr/VanRoyChapter.pdf ROY, P. V. Programming Paradigms for Dummies: What Every Programmer Should know, 2012

PARADIGMAS DE PROGRAMAÇÃO



PARADIGMA IMPERATIVO

- Fundamentado na ideia de computação como um processo que realiza mudanças de estados;
- Um estado representa uma configuração da memória do computador;
- LPs incluídas nesse paradigma especificam como uma computação é realizada por uma sequência de alterações no estado da memória do computador;

PARADIGMA IMPERATIVO

- Podemos relacionar com o comportamento imperativo das linguagens naturais que expressam ordens;
- O foco dos programas é especificar como um processamento deve ser feito no computador;
- Conceitos fundamentais: variável, valor e atribuição;
- Esse paradigma é subdividido em: estruturado (ou procedural), orientado a objetos

PARADIGMA IMPERATIVO: ESTRUTURADO

- Programas divididos em subprogramas e em blocos aninhados;
- Enfocam o controle de execução dos programas;
- Não utilizam desvios incondicionais (goto);
- Exemplos de LPs: Pascal, C, etc;

EXEMPLO DE PROGRAMA ESTRUTURADO

```
#include <stdio.h>
#include <math.h>
int lerNum();
int raiz(int z);
int main(void) {
      int x, r;
      x = lerNum();
      r = raiz(x);
}
int lerNum() {
      int y;
      scanf("%d", &y);
      return y;
}
int raiz(int z) {
      return sqrt(z);
}
```

PARADIGMA IMPERATIVO: ORIENTADO A OBJETOS

- O programa é entendido como um conjunto de objetos que interagem entre si;
- Objetiva tornar mais rápido e confiável o desenvolvimento de sistemas, focando os dados como elementos básicos;
- Os conceitos importantes são: classes, objetos, herança e polimorfismo;
- O paradigma orientado a objetos pode ser considerado uma evolução do paradigma estruturado;
- Exemplo de LPs: C++, Java, C#;

EXEMPLO DE PROGRAMA ORIENTADO A OBJETOS

```
public class Pessoa {
    String nome;
    int idade;

    Pessoa(String nome, int idade) {
        this.nome = nome;
        this.idade = idade;
    }
}
```

```
public class Aluno extends Pessoa {
    int ra;

    Aluno(String nome, int idade, int ra)
    {
        super(nome, idade);
        this.ra = ra;
    }
}
```

PARADIGMA DECLARATIVO

- Neste paradigma os programas são especificações de como é a tarefa a ser realizada;
- O programador não precisa saber como o computador é implementado, nem sobre a maneira pelo qual ele é melhor utilizado;
- Os programas são especificações de relações e funções;
- Não existem atribuições a variáveis;
- As variáveis são incógnitas e não representam posições de memória;

PARADIGMA DECLARATIVO: FUNCIONAL

- LPs funcionais operam apenas com funções, tratando a computação como uma avaliação de funções matemáticas;
- As funções recebem listas de valores e retornam um valor como resposta do problema (objetivo da programação);
- Um programa funcional é uma chamada de função que chama outras funções;
- As principais operações são a definição de funções e suas chamadas recursivas;
- Exemplo de LPs: Lisp, Haskell, ML e Elixir;

EXEMPLO DE PROGRAMA FUNCIONAL

```
(defun factorial (n)
    (if (<= n1)
          1
    (* n (factorial (- n1)))))
```

PARADIGMA DECLARATIVO: LÓGICO

- São baseadas em cálculo de predicados;
- Um predicado define uma relação entre constantes ou variáveis;
- A execução dos programas corresponde a um processo de dedução automático;
- Exemplo de LP: Prolog;

EXEMPLO DE PROGRAMA EM LÓGICA

```
pai(jose, ana);
pai(pedro, jose);
avo(x, z) :- pai(x, y), pai(y, z);
?-avo(x, ana).
```

PARADIGMA DECLARATIVO: REATIVO

- Baseado em mudanças em fontes de dados ou ambiente;
- Responde a eventos, como cliques ou nos dados;
- Utiliza fluxo de dados assíncrono;
- Pilares:
  - Responsividade
  - Resiliência
  - Elasticidade
  - Guiado por Mensagens
- Exemplo de Framework: RxJS (JavaScript) e ReactiveX;

PROGRAMAÇÃO CONCORRENTE

- Ocorre quando vários processos executam simultaneamente e concorrem por recursos;
- Podem utilizar uma única unidade de processamento ou várias unidades em paralelo;
- As unidades podem estar em um mesmo computador ou distribuída em vários;
- Exemplo de LPs: Ada, Java, etc;

LINHA DO TEMPO - LINGAGENS DE PROGRAMAÇÃO

- 1843 – Ada Lovelace
- 1936 – Alan Turing
- 1940 – Plankalkul
- 1949 – Assembly e Shortcode
- 1952 – Autocode
- 1957 – Fortran
- 1958 – ALGOL e LISP

https://www.computer.org/publications/tech-news/insider-membership-news/timeline-of-programming-languages

LINHA DO TEMPO - LINGAGENS DE PROGRAMAÇÃO

- 1959 – COBOL
- 1964 – BASIC
- 1970 – PASCAL
- 1972 – Smalltalk, C e SQL
- 1980 – ADA
- 1983 – C++ e Objective C
- 1987 – Perl
- 1990 – Haskell

https://www.computer.org/publications/tech-news/insider-membership-news/timeline-of-programming-languages

LINHA DO TEMPO - LINGAGENS DE PROGRAMAÇÃO

- 1991 – Python e Visual Basic
- 1993 – Ruby
- 1995 – Java, JavaScript e PHP
- 2000 – C#
- 2003 – Scala e Groovy
- 2009 – Go
- 2014 - Swift

https://www.computer.org/publications/tech-news/insider-membership-news/timeline-of-programming-languages

BIBLIOGRAFIA

- SEBESTA, R. W. **Concepts of Programming Languages**, 4ª edição, Addison-Wesley Logman, Inc, 1999.

- VAREJÃO, F. **Linguagens de Programação**: Conceitos e Técnicas. 1ª edição, Editora Campus, Rio de Janeiro, 2004.

- VAN ROY, P. Programming Paradigms for Dummies: What Every Programmer Should know, 2012.

- VAN ROY, P.; HARIDI, S. **Concepts, Techniques, and Models of Computer Programming**. Cambridge, MA: MIT Press, 2004. 936 p. ISBN 978-0-262-22069-9

Próxima aula estudaremos

- Conceitos de Orientação a Objetos.

CONTATO

proflucas.gonzalez@fiap.com.br