UFMT

UNIVERSIDADE FEDERAL DE MATO GROSSO

Faculdade de Ciência e Tecnologia

Departamento de Computação e Tecnologia

Campus de Várzea Grande

PLANO DE AULA

1. IDENTIFICAÇÃO

Professor	Prof. Dr. Willian Garcias de Assunção
Curso	Bacharelado em Ciência da Computação
Disciplina	Introdução à Programação Orientada a Objetos
Título da aula	Programação Orientada a Objetos: classes, objetos, atributos e métodos
Público-alvo	Estudantes de graduação (2.º semestre)
Duração da aula	45 minutos
Data	7 de junho de 2026

2. PRÉ-REQUISITOS

Lógica de programação e fundamentos de programação estruturada na linguagem Java (variáveis, tipos primitivos, estruturas de controle, métodos e arranjos).

3. OBJETIVO GERAL

Compreender os conceitos fundamentais da Programação Orientada a Objetos (**classes**, **objetos**, **atributos** e **métodos**) e aplicá-los na modelagem e na implementação de soluções de software em linguagem Java.

4. OBJETIVOS ESPECÍFICOS

Ao final da aula, espera-se que o estudante seja capaz de:

- distinguir os conceitos de **classe** (molde) e **objeto** (instância);
- identificar **atributos** (estado) e **métodos** (comportamento) na definição de uma classe;
- aplicar **modificadores de acesso** e o princípio do **encapsulamento**;
- implementar uma **classe em Java**, com atributos privados, construtor e métodos de acesso.

5. CONTEÚDO PROGRAMÁTICO

1. **Fundamentos da POO:** o paradigma orientado a objetos; estado e comportamento; os quatro pilares (abstração, encapsulamento, herança e polimorfismo), com foco em **abstração** e **encapsulamento**; da modelagem do mundo real para o software.

2. **Classes:** definição e anatomia de uma classe; sintaxe de declaração em Java; cabeçalho e corpo (atributos, construtor e métodos); **modificadores de acesso** (`private`, *default*, `protected`, `public`).

3. **Objetos, atributos e métodos:** instanciação com o operador `new` e referências (*stack* × *heap*); atributos de instância e de classe (`static`); **encapsulamento**, *getters*/*setters*, construtores e sobrecarga de métodos (*overloading*).

4. **Aplicação e síntese:** estudo de caso integrador (classe `ContaBancaria`) e traço de execução; questão de verificação; síntese conceitual e encaminhamentos.

6. METODOLOGIA

A aula será expositiva e dialogada, apoiada na projeção de slides e no quadro. Inicialmente, o professor contextualiza o tema, motivando a transição da programação estruturada para a orientada a objetos e retomando conhecimentos prévios da turma. Em seguida, desenvolve os conceitos de forma progressiva, do paradigma e da abstração até a definição de classes e à criação de objetos, atributos e métodos, articulando teoria, sintaxe Java e analogias com o mundo real (planta arquitetônica/casas; entidades como Pessoa, Carro e Conta). A consolidação ocorre por meio de um estudo de caso integrador (classe `ContaBancaria`), acompanhado de traço de execução. O encerramento prevê uma questão objetiva de verificação, uma síntese em mapa conceitual e a proposição de exercícios e leitura prévia para a aula seguinte.

7. SEQUÊNCIA DIDÁTICA

Etapa	Atividades didáticas
Abertura e contextualização	Apresentação do tema e dos objetivos; motivação (por que POO; limites da programação estruturada); conexão com conhecimentos prévios.
Fundamentos da POO	Paradigma orientado a objetos; estado e comportamento; os quatro pilares (foco em abstração e encapsulamento); modelagem do mundo real para o software.
Classes	Definição e anatomia de uma classe; sintaxe de declaração em Java; atributos, construtor e métodos; modificadores de acesso.
Objetos, atributos e métodos	Instanciação (operador `new`) e referências (*stack* × *heap*); atributos de instância e de classe; encapsulamento, *getters*/*setters*, construtores e sobrecarga.
Aplicação	Estudo de caso integrador (classe `ContaBancaria`) e traço de execução, articulando todos os conceitos.
Verificação e fechamento	Questão objetiva; síntese em mapa conceitual; encaminhamentos (exercícios e leitura prévia).

8. RECURSOS DIDÁTICOS

Projetor multimídia e notebook para a projeção dos slides; quadro branco e pincel para esquemas e detalhamentos conforme a interação com a turma; material de apoio on-line (ver seção 10).

9. AVALIAÇÃO

A verificação é formativa, contínua e sistemática, considerando:

- a participação e a interação durante as discussões e a construção dos exemplos;
- a resolução, ao final da aula, de uma questão objetiva (ENADE 2017, sobre encapsulamento e modificadores de acesso), para checagem imediata da compreensão;
- exercícios para casa: (i) modelar e implementar em Java a classe `Livro` (título, autor, ano, disponível), com operações de emprestar e devolver; (ii) implementar a classe `Aluno` (matrícula, nome, notas), com cálculo de média e situação; (iii) justificar, em até cinco linhas, por que atributos devem ser privados.

10. ACESSO AO CONTEÚDO DA AULA

Conteúdo da aula

`willian.cc/ufmt`

Aula em vídeo

`youtu.be/2VCgpiahVFQ`

Slides, exemplos de código e exercícios no QR à esquerda; versão em vídeo da aula no QR à direita.

11. BIBLIOGRAFIA

Básica

BARNES, David J.; KÖLLING, Michael. **Programação orientada a objetos com Java** (BlueJ). 4. ed. São Paulo: Pearson Prentice Hall, 2009.

DEITEL, Paul; DEITEL, Harvey. **Java: como programar**. 10. ed. São Paulo: Pearson, 2016.

Complementar

BOOCH, Grady; RUMBAUGH, James; JACOBSON, Ivar. **UML: guia do usuário**. 2. ed. Rio de Janeiro: Elsevier, 2006.

PRESSMAN, Roger S.; MAXIM, Bruce R. **Engenharia de software:** uma abordagem profissional. 9. ed. Porto Alegre: McGraw-Hill/AMGH, 2021.

INEP. **Exame Nacional de Desempenho dos Estudantes (ENADE) 2017:** prova de Ciência da Computação. Brasília: INEP/MEC, 2017.

Cuiabá/MT, 7 de junho de 2026.

Prof. Dr. Willian Garcias de Assunção
Docente responsável