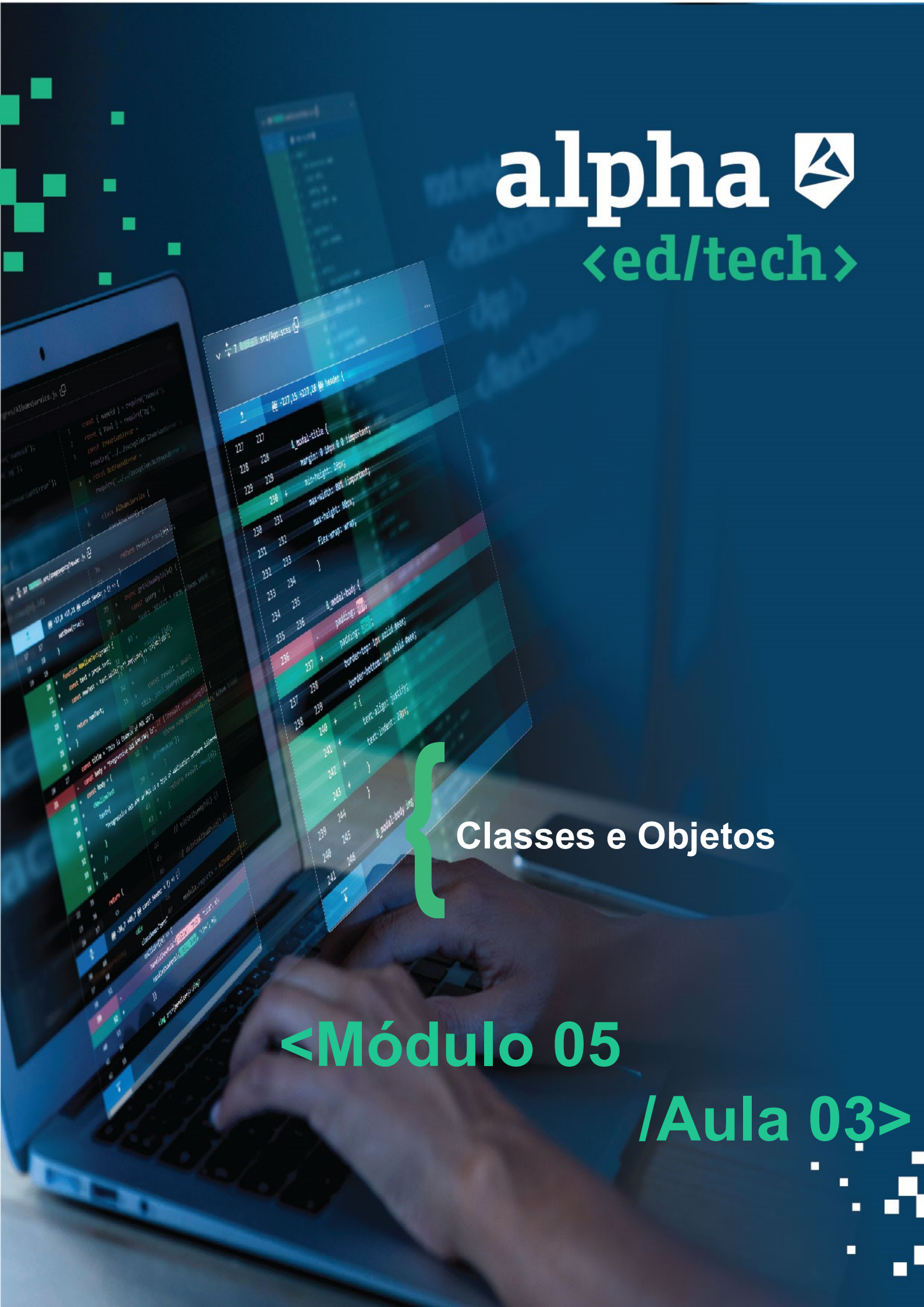
alpha
<ed/tech>
Classes e Objetos
<Módulo 05
/Aula 03>

Classes e Objetos

Python - Programação Orientada a Objetos

Na programação orientada a objetos (POO), uma classe é uma estrutura fundamental que representa um modelo para criar objetos (**instanciar um Objeto a partir da Classe**). Ela funciona como um plano ou projeto que define **atributos (características)** e **métodos (comportamentos)** que os objetos desse tipo terão.

Um objeto possui um **estado** e uma coleção de **métodos** que ele pode executar. **Atributos** são as **propriedades de um objeto**. **O estado de um objeto representa as coisas que o objeto sabe sobre si mesmo**.

Métodos são as ações que um objeto pode realizar. Os **métodos** determinam o comportamento dos objetos de uma classe e são **análogos às funções ou procedimentos da programação estruturada**.

A troca de mensagens entre objetos, ou seja, chamadas dos métodos dos objetos instanciados é que fazem o sistema aparecer. Por exemplo, um objeto que representa um Funcionário `taniguchi` envia o nome desse Funcionário (`nome.taniguchi`), conteúdo desse atributo ("Kenji Taniguchi") para um objeto telaCadastro que foi instanciada no monitor do seu computador, dentro do aplicativo que está executando, neste exemplo o método poderia ser getNome().taniguchi (método que pega o nome e envia uma string para quem chamou).

Python é uma linguagem Orientada a Objetos e as variáveis são objetos. Em Python, uma classe é uma estrutura que permite organizar e estruturar dados e funcionalidades relacionadas. Classe serve como um modelo para criar objetos, que são instâncias dessa classe.

1. Definindo uma Classe:

```python
class Carro:
    def __init__(self, modelo, cor):
        self.modelo = modelo
        self.cor = cor
```

Aqui, criamos a classe `Carro` com atributos `modelo` e `cor`. O método `__init__` é um construtor que inicializa os atributos quando um objeto é criado.

2. Criando Objetos (Instâncias) a partir de uma Classe:

```python
meu_carro = Carro("Sedan", "Prata")
```

Agora, `meu_carro` é uma instância da classe `Carro` com modelo "Sedan" e cor "Prata".

Classes e Objetos

3. Atributos e Métodos:

```
class Carro:
      def __init__(self, modelo, cor):
          self.modelo = modelo
          self.cor = cor
      def acelerar(self):
          print(f"O carro {self.modelo} está acelerando!")
   meu_carro = Carro("Sedan", "Prata")
   meu_carro.acelerar()
```

`acelerar` é um método da classe que pode ser chamado em instâncias. No exemplo, imprime uma mensagem.

Classes em Python oferecem uma estrutura poderosa para organizar código, promovendo a reutilização e facilitando a manutenção.
Em **Python**, você pode **usar uma classe como atributo em outra classe** para modelar relacionamentos mais complexos e organizar melhor o código. Isso é conhecido como composição. Vamos analisar um exemplo detalhado de como uma classe pode ser usada como atributo em outra classe:

```
```python
# Definindo a classe Endereco
class Endereco:
    def __init__(self, rua, cidade, estado):
        self.rua = rua
        self.cidade = cidade
        self.estado = estado
    def __str__(self):
        return f"{self.rua}, {self.cidade}, {self.estado}“
# Definindo a classe Pessoa que usa a classe Endereco como atributo
class Pessoa:
    def __init__(self, nome, idade, endereco):
        self.nome = nome
        self.idade = idade
        self.endereco = endereco
  # Usando a classe Endereco como atributo
    def __str__(self):
        return f"{self.nome}, {self.idade} anos, morando em {self.endereco}“
# Criando instâncias da classe Endereco
endereco1 = Endereco("Rua A", "Cidade A", "Estado A")
endereco2 = Endereco("Rua B", "Cidade B", "Estado B")
# Criando instâncias da classe Pessoa com diferentes endereços
pessoa1 = Pessoa("João", 25, endereco1)
pessoa2 = Pessoa("Maria", 30, endereco2)
# Exibindo informações das instâncias
print(pessoa1)
print(pessoa2)
```
```

Neste exemplo, a classe `Pessoa` tem um atributo `endereco`, que é uma instância da classe `Endereco`. Dessa forma, você pode agrupar dados relacionados e organizar o código de maneira mais modular. Ao utilizar classes como atributos, você cria uma relação de composição, onde um objeto é composto por outros objetos. Isso promove a reutilização de código e torna o design mais flexível e expansível.

Classes e Objetos

Atributos de Classe e Atributos de Objetos

Variáveis de Instância (do Objeto):

Definição: São variáveis associadas a instâncias específicas de uma classe.
Acesso: São acessadas usando a notação `self.variavel` dentro dos métodos da classe.
Uso: Armazenam **dados específicos de cada objeto** criado a partir da classe.

Exemplo:

```
class Exemplo:
    def __init__(self, nome):
        self.nome = nome

obj1 = Exemplo('Objeto1')
print(obj1.nome) # Saída: Objeto1
```

Variáveis de Classe:

Definição: São **variáveis compartilhadas por todas as instâncias de uma classe**.
Acesso: São acessadas usando a notação `NomeDaClasse.variavel` ou `self.__class__.variavel`.
Uso: Armazenam dados que são comuns a todas as instâncias da classe.

Exemplo:

```
```python
# Definindo a classe Endereco
class Exemplo:
    variavel_compartilhada = 'Compartilhada'

obj1 = Exemplo()
obj2 = Exemplo()
print(obj1.variavel_compartilhada)  # Saída: Compartilhada
print(obj2.variavel_compartilhada)  # Saída: Compartilhada
```
```

Diferenças e Uso

1. **Escopo:**
   - Instância: Pertence a uma instância específica da classe.
   - Classe: Pertence à classe como um todo.

2. **Acesso:**
   - Instância: Acessada através da instância (`self`).
   - Classe: Acessada através do nome da classe ou da instância (`self.__class__`).

Classes e Objetos

3. **Modificação:**
   - Instância: Modificada separadamente para cada objeto.
   - Classe: Alterada para todas as instâncias quando modificada em qualquer uma delas.

4. **Uso:**
   - Instância: Armazenar dados específicos a cada objeto.
   - Classe: Armazenar dados compartilhados por todas as instâncias.

Escolha entre variáveis de instância e de classe depende do que você deseja realizar.

Se precisar armazenar dados únicos para cada objeto, use variáveis de instância. Se quiser compartilhar dados entre todas as instâncias, use variáveis de classe. Em muitos casos, ambos são usados em conjunto para atender às necessidades específicas do programa.