DIGITAL
INNOVATION
ONE

Orientação a Objetos

Thiago Leite e Carvalho
Engenheiro de Software, Professor, Escritor

DIGITAL INNOVATION ONE

Mais sobre mim

- Java, backend, docência, livros, artigos e cursos
- Mestre e Serpro
- O que me motiva?
- Pizzas e massas, cerveja e futebol

DIGITAL INNOVATION ONE

Mais sobre mim

- https://www.linkedin.com/in/thiago-leite-e-carvalho-1b337b127/
- https://github.com/thiagoleitecarvalho
- https://github.com/tlcdio

DIGITAL INNOVATION ONE

Objetivo do curso

Possibilitar que o aluno compreenda todos os conceitos relativos à Orientação a Objetos(OO).

DIGITAL INNOVATION ONE

Percurso

Aula 1 Porque usar?

Aula 2 Os fundamentos

Aula 3 A estrutura

Aula 4 As relações

DIGITAL INNOVATION ONE

Percurso

Aula 5 A Organização

Aula 6 Próximos passos

DIGITAL INNOVATION ONE

Requisitos

- ✔ Lógica de Programação
- ✔ Vontade de aprender

DIGITAL INNOVATION ONE

Dúvidas durante o curso?

> Fórum do curso
> Comunidade online (discord)

Aula 1: Porque usar?

Orientação a Objetos

DIGITAL INNOVATION ONE

Objetivos

1. Explicar porque devemos programar orientado a objetos?

DIGITAL INNOVATION ONE

PE vs POO

- Paradigma Estruturado tem uma representação mais simplista
- Paradigma Orientado a Objeto tem uma representação mais realista

DIGITAL INNOVATION ONE

PE vs POO

- Paradigma Estruturado foca em operações(funções) e dados
- Paradigma Orientado a Objetos foca na modelagem de entidades e nas interações entre estas

DIGITAL INNOVATION ONE

PE vs POO

- Programação Estruturada foca mais no "como fazer"
- Programação Orientada a Objetos foca mais no "o que fazer"

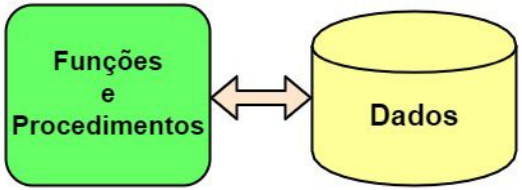
DIGITAL INNOVATION ONE

Vantagens da POO

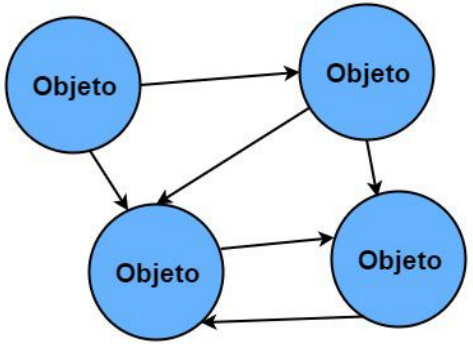
- Melhor coesão
- Melhor acoplamento
- Diminuição do Gap semântico
- Coletor de lixo

DIGITAL INNOVATION ONE

Resumo



Programação Estruturada



Programação Orientada a Objetos

DIGITAL
INNOVATION
ONE

Aula 2: Os fundamentos

Orientação a Objetos

DIGITAL INNOVATION ONE

Objetivos

1. Definir o que é a OO?

2. Entender os pilares no qual o paradigma se sustenta.

DIGITAL INNOVATION ONE

Definição

"A Orientação a Objetos é um paradigma de análise, projeto e programação de sistemas de software baseado na composição e interação entre diversas unidades de software chamadas de objetos."

(https://pt.wikipedia.org/wiki/Orientação_a_objetos)

DIGITAL INNOVATION ONE

Fundamentos

- Abstração

"Processo pelo qual se isolam características de um objeto, considerando os que tenham em comum certos grupos de objetos."

DIGITAL INNOVATION ONE

Fundamentos

- Reuso

"Capacidade de criar novas unidades de código a partir de outras já existentes."

DIGITAL INNOVATION ONE

Fundamentos

- Encapsulamento

"Capacidade de esconder complexidades e proteger dados."

DIGITAL INNOVATION ONE

Exercitando

Levando em consideração uma loja on-line de livros, modele uma entidade livro.

DIGITAL INNOVATION ONE

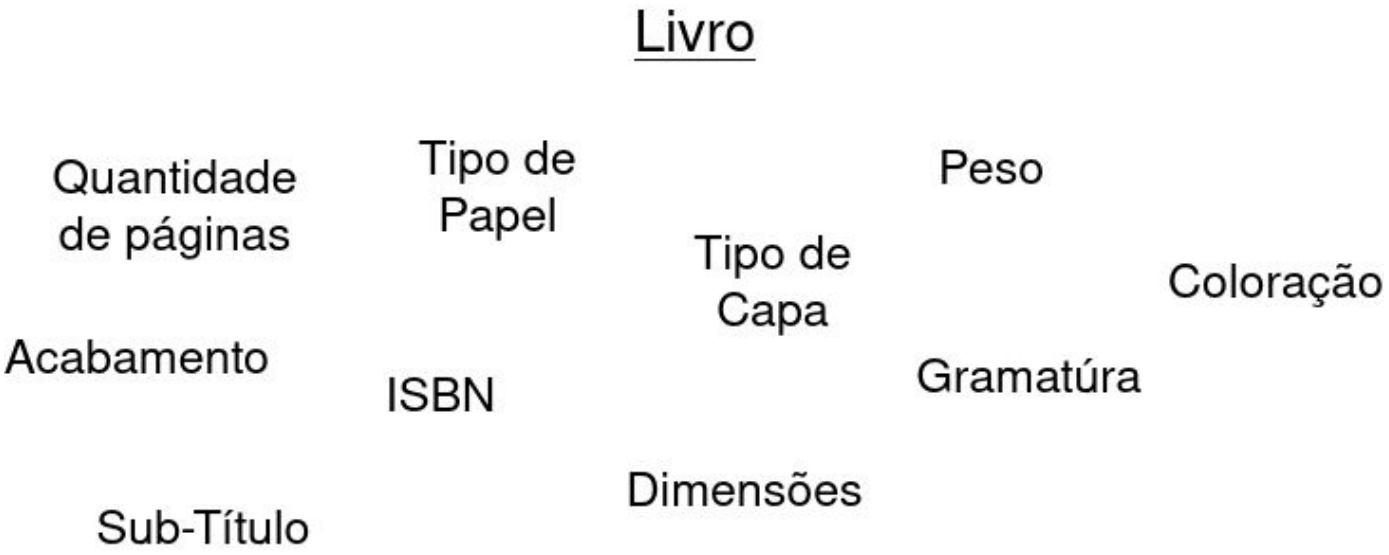
Exercitando

DIGITAL INNOVATION ONE

Exercitando

DIGITAL INNOVATION ONE

Exercitando



DIGITAL INNOVATION ONE

Exercitando

DIGITAL
INNOVATION
ONE

Aula 3: A estrutura

Orientação a Objetos

DIGITAL INNOVATION ONE

Objetivos

1. Apresentar os conceitos que criam as estruturas básicas da OO:

- Classe
- Atributo
- Método
- Objeto
- Mensagem

DIGITAL INNOVATION ONE

Classe

"É uma estrutura que abstrai um conjunto de objetos com características similares. Uma classe define o comportamento de seus objetos através de métodos e os estados possíveis destes objetos através de atributos. Em outros termos, uma classe descreve os serviços providos por seus objetos e quais informações eles podem armazenar."

(https://pt.wikipedia.org/wiki/Classe_(programação))

DIGITAL INNOVATION ONE

Classe

- Exemplos:

Bola

Carro

Viagem

Computador

Venda

Comprador

DIGITAL INNOVATION ONE

Classe

- Dicas:

- Substantivos
- Nome significativos
- Contexto deve ser considerado

DIGITAL INNOVATION ONE

Classe

- Códigos:

Java

```
class Carro {

}
```

C#

```
class Carro {

}
```

Python

```
class Carro:
    pass
```

DIGITAL INNOVATION ONE

Exercitando 1

Crie a classe "Carro".

Obs: use a linguagem que gostar e siga as dicas sobre como criar classes

DIGITAL INNOVATION ONE

Atributo

"É o elemento de uma classe responsável por definir sua estrutura de dados. O conjunto destes será responsável por representar suas características e fará parte dos objetos criados a partir da classe."

DIGITAL INNOVATION ONE

Atributo

- Exemplos:

Bola
- diametro

Carro
- cor

Viagem
- distancia

Computador
- memoria

Venda
- valor

Comprador
- nome

DIGITAL INNOVATION ONE

Atributo

Atributo x Variável

A: O que é próprio e peculiar a alguém ou a alguma coisa.
V: Sujeito a variações ou mudanças; que pode variar; inconstante, instável

DIGITAL INNOVATION ONE

Atributo

- Dicas:

- Substantivos e adjetivos
- Nome significativos
- Contexto deve ser considerado
- Abstração
- Tipos adequados

DIGITAL INNOVATION ONE

Atributo

- Códigos:

Java

```java
class Carro {
  int portas;
}
```

C#

```csharp
class Carro {
    int portas;
}
```

Python

```python
class Carro:
    portas = 0
```

DIGITAL INNOVATION ONE

Exercitando 2

Evolua o exercicio 1 e defina 3 atributos para sua classe "Carro": cor, modelo e capacidade do tanque.

Obs: use a linguagem que gostar e siga as dicas sobre como criar atributos

DIGITAL INNOVATION ONE

Método

"É uma porção de código (sub-rotina) que é disponibilizada pela classe. Este é executado quando é feita uma requisição a ele. Um método serve para identificar quais serviços, ações, que a classe oferece. Eles são responsáveis por definir e realizar um determinado comportamento"

DIGITAL INNOVATION ONE

Método

- Exemplos:

Carro

- ligar

Computador

- desligar

Venda

- calcular total

Comprador

- realizar troca

DIGITAL INNOVATION ONE

Método

- Criação:

Java e C#	Python
- Visibilidade	- `def`
- Retorno	- Nome
- Nome	- Parâmetros
- Parâmetros	

DIGITAL INNOVATION ONE

Método

- Dicas:

- Verbos
- Nome significativos
- Contexto deve ser considerado

DIGITAL INNOVATION ONE

Método

- Códigos:

Java

```
class Carro {
 void frear(){
   ...
  }
}
```

C#

```
class Carro {
 void frear(){
   ...
  }
}
```

Python

```
class Carro:
    def frear()
        ...
```

DIGITAL INNOVATION ONE

Método

- Dois métodos especiais:
  - Construtor
  - Destrutor

Método

- Dois métodos especiáis:
- ✔ Construtor

Java

```
class Carro {
 Carro(){
   ...
 }
}
```

C#

```
class Carro {
 Carro(){
   ...
 }
}
```

Python

```
class Carro:
    def __init__(self):
        ...
```

Método

- Dois métodos especiáis:
- ✔ Destrutor

Java

```
class Carro {
 void finalize(){
   ...
  }
}
```

C#

```
class Carro {
 ~Carro(){
   ...
  }
}
```

Python

```
class Carro:
    def __del__(self):
        ...
```

Método

- Sobrecarga:
  - Mudar a assinatura de acordo com a necessidade

    - Assinatura: nome + parâmetros

Java

```
m1()
m1(int i)
m1(float f)
m1(String s, long l)
m1(long l, String s)
```

C#

```
M1()
M1(int i)
M1(float f)
M1(String s, long l)
M1(long l, String s)
```

Python

  - Porque usar?

DIGITAL INNOVATION ONE

Exercitando 3

Evolua o conceito do exercício 2 e defina 1 método para calcular o valor total para encher o tanque. Este deve receber como parâmetro o valor da gasolina. Faça também duas sobrecargas do construtor.

Obs: use a linguagem que gostar e siga as dicas sobre como criar métodos
Obs: Crie métodos específicos para fornecer e obter os valores dos atributos(set/get), caso aplicável.

DIGITAL INNOVATION ONE

Objeto

"Um objeto é a representação de um conceito/entidade do mundo real, que pode ser física (bola, carro, árvore etc.) ou conceitual (viagem, estoque, compra etc.) e possui um significado bem definido para um determinado software. Para esse conceito/entidade, deve ser definida inicialmente uma classe a partir da qual posteriormente serão instanciados objetos distintos."

DIGITAL INNOVATION ONE

Objeto

- Criação:

Java

```
Carro carro = new Carro();
```

Python

```
carro = Carro()
```

C#

```
Carro carro = new Carro();
```

DIGITAL INNOVATION ONE

Mensagem

"É o processo de ativação de um método de um objeto. Isto ocorre quando uma requisição (chamada) a esse método é realizada, assim disparando a execução de seu comportamento descrito por sua classe. Pode também ser direcionada diretamente à classe, caso a requisição seja a um método estático."

DIGITAL INNOVATION ONE

Mensagem

Java

```
Carro carro = new Carro();
carro.<método>;

Carro.<método>;
```

Python

```
carro = Carro()
carro.<método>

Carro.<método>
```

C#

```
Carro carro = new Carro();
carro.<método>;

Carro.<método>;
```

DIGITAL INNOVATION ONE

Siga em frente...

- Instância x Estático: atributos e métodos
- Estado de um Objeto
- Identidade de um Objeto
- Representação numérica de um objeto
- Representação padrão de um objeto

DIGITAL INNOVATION ONE

Exercitando 4

Evolua o conceito do exercício 3 criando objetos da classe "Carro". Use os métodos set/get, quando aplicáveis, para definir os valores dos atributos e exibir estes valores "get". Passe também uma mensagem para o cálculo do total pra encher o tanque.

Obs: use a linguagem que gostar e siga as dicas
sobre como criar métodos, atributos, etc.
Obs: use System.out, Console.WriteLine ou print

DIGITAL
INNOVATION
ONE

Aula 4: As relações

Orientação a Objetos

DIGITAL INNOVATION ONE

Objetivos

1. Apresentar os conceitos que ajudam a criar entidades a partir de outras entidades:

- Herança
- Associação
- Interface

DIGITAL INNOVATION ONE

Herança

"É o relacionamento entre classes em que uma classe chamada de subclasse (classe filha, classe derivada) é uma extensão, um subtipo, de outra classe chamada de superclasse (classe pai, classe mãe, classe base). Devido a isto, a subclasse consegue reaproveitar os atributos e métodos dela. Além dos que venham a ser herdados, a subclasse pode definir seus próprios membros."

DIGITAL INNOVATION ONE

Herança

- Códigos:

Java

```
class A extends B {
   ...
}
```

C#

```
class A : B {
    ...
}
```

Python

```
class A(B):
    ...
```

DIGITAL INNOVATION ONE

Exercitando 1

Crie a classe "Veiculo","Carro", "Moto" e "Caminhao".

Obs: use a linguagem que gostar e siga as dicas sobre como criar classes. Faça a relação de herança que julgue adequada.

DIGITAL INNOVATION ONE

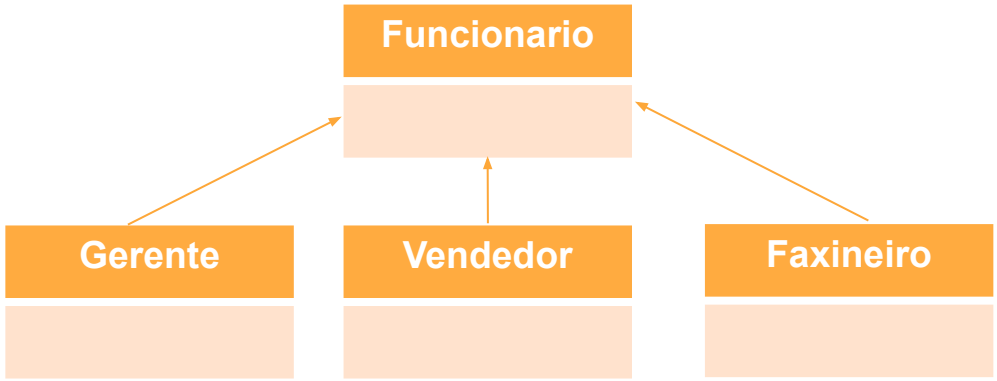
Herança

- Tipos de herança:
  - Simples
  - Múltipla

Herança

- Tipos de herança:
  - Simples

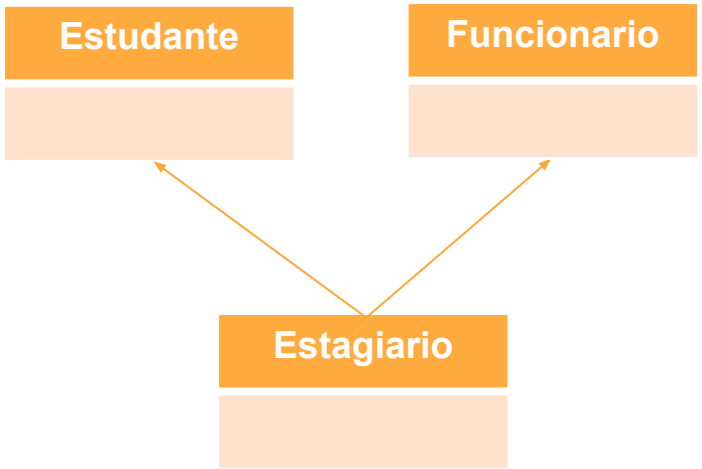
- A classe filha tem só uma classe mãe



Herança

- Tipos de herança:
  - Mútipla

- A classe filha tem uma ou mais classes mães



Herança

- Tipos de herança:
  - Mútipla

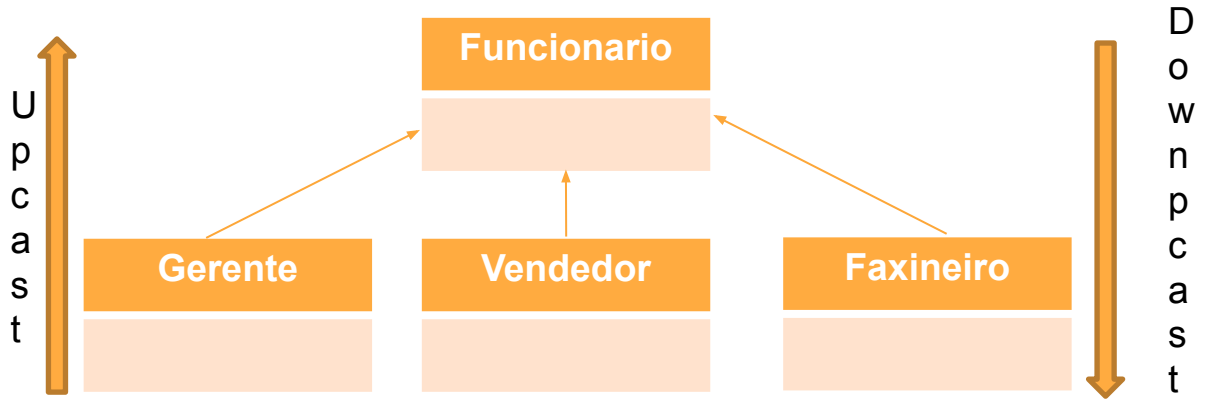
- Java
- C#
- Python
- C++

Python

```
class A(B,C):
    pass
```

Herança

- Upcast e Downcast:



DIGITAL INNOVATION ONE

Herança

- Upcast:

Java

```
A a = new B();
```

C#

```
A a = new B();
```

Python

DIGITAL INNOVATION ONE

Herança

- Downcast:

Java

```
B b = (B)new A();
```

C#

```
B b = (B)new A();
```

Python

DIGITAL INNOVATION ONE

Herança

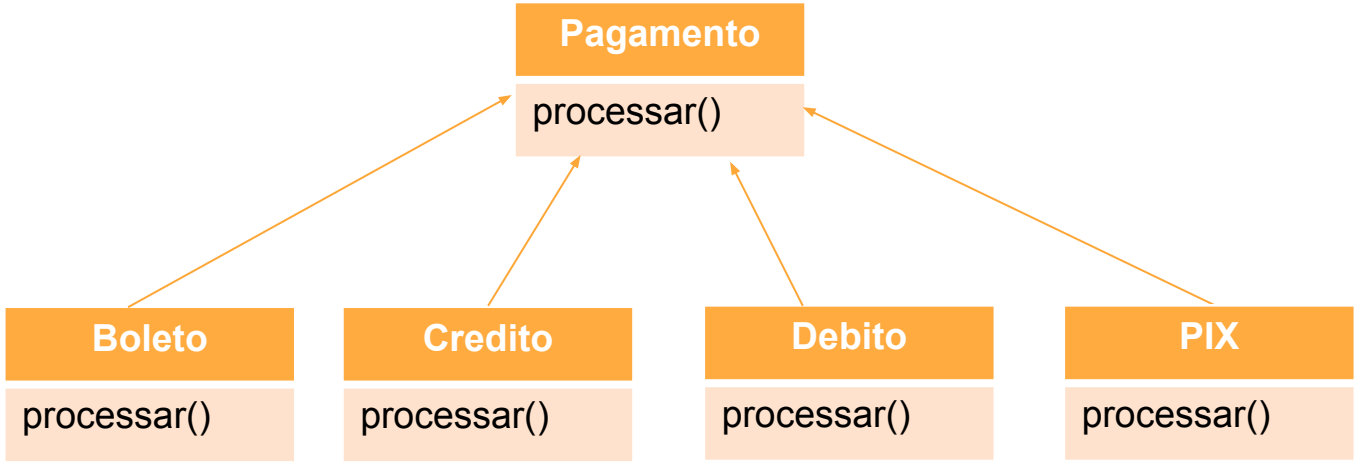
- Polimorfismo

"A mesma ação, se comportando diferente."

DIGITAL INNOVATION ONE

Herança

- Polimorfismo



DIGITAL INNOVATION ONE

Herança

- Polimorfismo
  - Códigos:
    - Ver exemplo

DIGITAL INNOVATION ONE

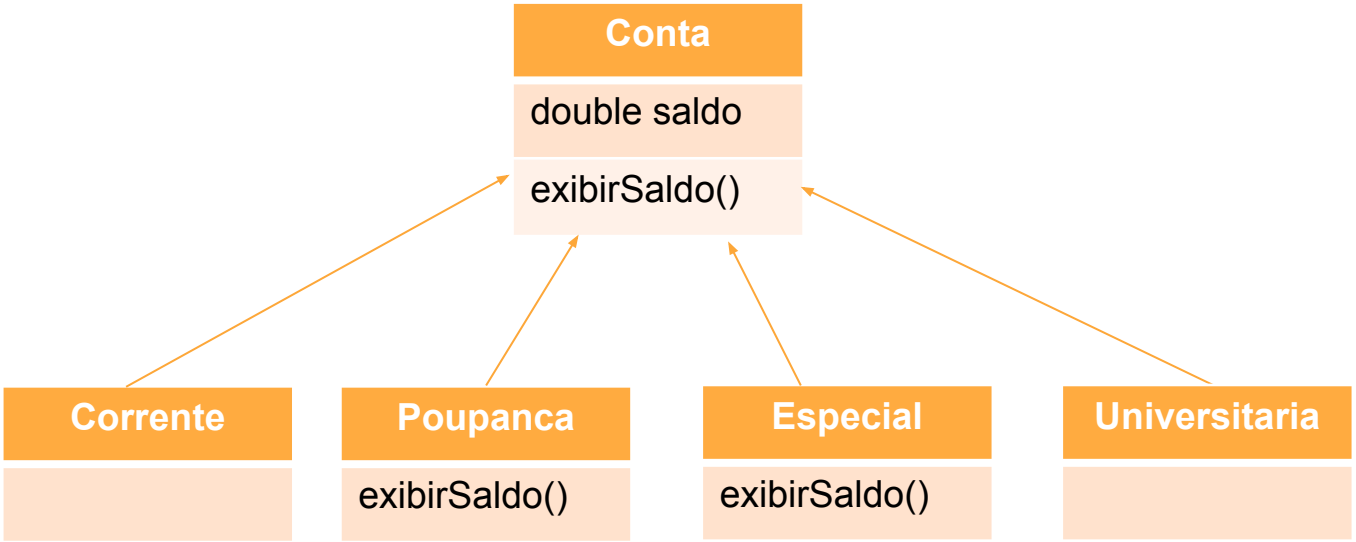
Herança

- Sobrescrita

"A mesma ação, podendo se comportar diferente. "

Herança

- Sobrescrita



DIGITAL INNOVATION ONE

Herança

- Sobrescrita
  - Códigos:
    - Ver exemplo

DIGITAL INNOVATION ONE

Curiosidade

Polimorfismo x Sobrescrita

DIGITAL INNOVATION ONE

Exercitando 2

Crie as classes "Funcionario", "Gerente", "Vendedor" e "Faxineiro". Realize upcasts e downcasts.

Obs: use a linguagem que gostar e siga as dicas sobre como criar classes. Faça a relação de herança de acordo com o slide.

DIGITAL INNOVATION ONE

Exercitando 3

Analise do comportamento de Polimorfismo e Sobrescrita.

DIGITAL INNOVATION ONE

Associação

"Possibilita um relacionamento entre classes/objetos, no qual estes possam pedir ajuda a outras classes/objetos e representar de forma completa o conceito ao qual se destinam. Neste tipo de relacionamento, as classes e os objetos interagem entre si para atingir seus objetivos."

DIGITAL INNOVATION ONE

Associação

- Tipos:
  - Estrutural
    - Composição
    - Agregação
  - Comportamental
    - Dependência

Associação

- Tipos:
  - Estrutural
    - Composição: "Com Parte Todo"
    - Ex: Pessoa e Endereço

DIGITAL INNOVATION ONE

Associação

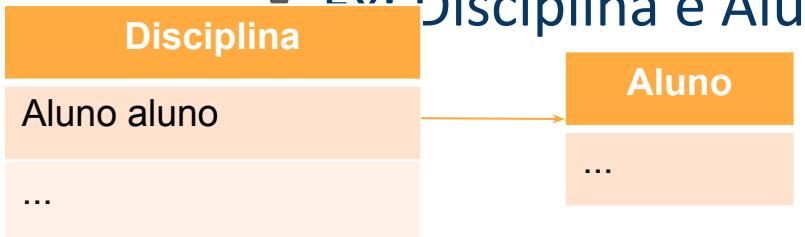
- Tipos:
  - Estrutural
    - Composição: "Com Parte Todo"
    - Ex: Pessoa e Endereço

Java

```
class Pessoa {
  Endereco endereco;
}
```

Associação

- Tipos:
  - Estrutural
    - Agregação: "Sem Parte Todo"
    - Ex: Disciplina e Aluno



DIGITAL INNOVATION ONE

Associação

- Tipos:
  - Estrutural
    - Composição: "Sem Parte Todo"
    - Ex: Pessoa e Endereço

Java

```
class Disciplina {
  Aluno aluno;
}
```

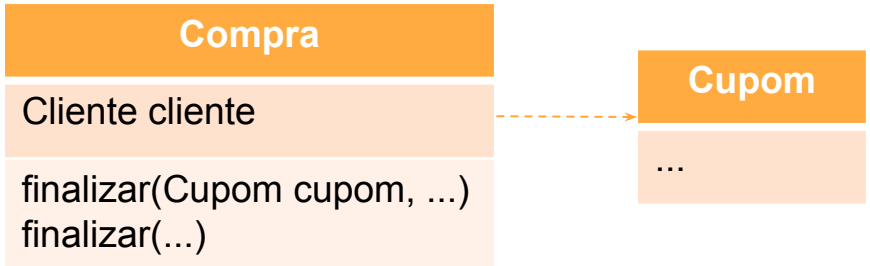
DIGITAL INNOVATION ONE

Curiosidade

Agregação x Composição

Associação

- Tipos:
  - Comportamental
    - Dependência: "Depende de"



DIGITAL INNOVATION ONE

Associação

- Tipos:
  - Comportamental
    - Dependência: "Depende de"

Java

```
class Compra {
   ...
   finalizar(Cupom cupom, ...) {
     ...
   }
}
```

DIGITAL INNOVATION ONE

Curiosidade

Herança x Associação

DIGITAL INNOVATION ONE

Exercitando 4

Apenas para praticar, codifique os exemplos dos slides anteriores sobre as associações.

Obs: use a linguagem que gostar e siga as dicas sobre como criar classes.

DIGITAL INNOVATION ONE

Interface

"Define um contrato que deve ser seguido pela classe que a implementa. Quando uma classe implementa uma interface, ela se compromete a realizar todos os comportamentos que a interface disponibiliza."

DIGITAL INNOVATION ONE

Interface

- Códigos:

Java

```
interface A {
   ...
}

class B implements A {
   ...
}
```

C#

```
interface A {
    ...
}

class B : A {
   ...
}
```

Python

DIGITAL INNOVATION ONE

Exercitando 5

Apenas para praticar, crie uma interface chamada "OperacaoMatematica". Crie também 4 métodos das operações básicas: soma, subtração, multiplicação e divisão.

Obs: use a linguagem que gostar e siga as dicas sobre como criar classes. Tente não implementar algum dos métodos e veja o que acontece.

DIGITAL INNOVATION ONE

Siga em frente...

- Tipos de classe: Abstrata e Concreta
- Métodos abstratos
- Caracteríscas das associações
- Palavras coringas: super, base e super()
- Relações entre classes e interface: extends e implements

DIGITAL INNOVATION ONE

Aula 5: A organização

Orientação a Objetos

DIGITAL INNOVATION ONE

Objetivos

1. Apresentar os conceitos que organização a OO:

- Pacotes
- Visibilidades

DIGITAL INNOVATION ONE

Pacotes

"São uma organização física ou lógica criada para separar classes com responsabilidades distintas. Com isso, espera-se que a aplicação fique mais organizada e seja possível separar classes de finalidades e representatividades diferentes."

DIGITAL INNOVATION ONE

Pacotes

- Códigos:

Java	C#	Python
`package ...;`	`namespace {...}`	`__init__.py(2.x)`
`import ...;`	`using ...;`	`from ...` `import ...`

DIGITAL INNOVATION ONE

Exercitando

Ver projeto de exemplo para entender funcionamento.

DIGITAL INNOVATION ONE

Visibilidades

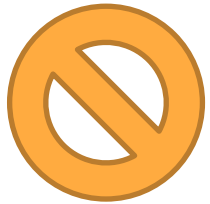
"Um modificador de acesso tem como finalidade determinar até que ponto uma classe, atributo ou método pode ser usado. A utilização de modificadores de acesso é fundamental para o uso efetivo da Orientação a Objetos. Algumas boas práticas e conceitos só são atingidos com o uso corretos deles."

DIGITAL INNOVATION ONE

Visibilidades

- Tipos:
  - Private
  - Protected
  - Public

Python



DIGITAL INNOVATION ONE

Visibilidades

- Tipos:
  - Private: Só dentro a classe

Java

```
private

private int i;
private void do();
```

C#

```
private

private int i;
private void Do();
```

Visibilidades

- Tipos:
  - Protected: Dentro da classe, mesmo pacote e subclasses

Java

```
protected

protected int i;
protected void do();
```

C#

```
protected

protected int i;
protected void Do();
```

DIGITAL INNOVATION ONE

Visibilidades

- Tipos:
  - Public: Em qualquer lugar

Java

```
public

public int i;
public void do();
```

C#

```
public

public int i;
public void Do();
```

DIGITAL INNOVATION ONE

Curiosidade

Qua usar?l

DIGITAL INNOVATION ONE

Exercitando

Ver projeto de exemplo para entender funcionamento.

DIGITAL INNOVATION ONE

Aula 6: Próximos passos

Orientação a Objetos

- Padrões de Projeto(Desing Patterns)
- Boa práticas: SOLID, código, tecnicas de programação, etc.
- Refatoração
- UML
- Frameworks
- MUITA prática e estudo!

DIGITAL INNOVATION ONE

Dúvidas?

> Fórum do curso
> Comunidade online (discord)

DIGITAL INNOVATION ONE

Para saber mais

- https://www.casadocodigo.com.br/products/livro-oo-conceitos