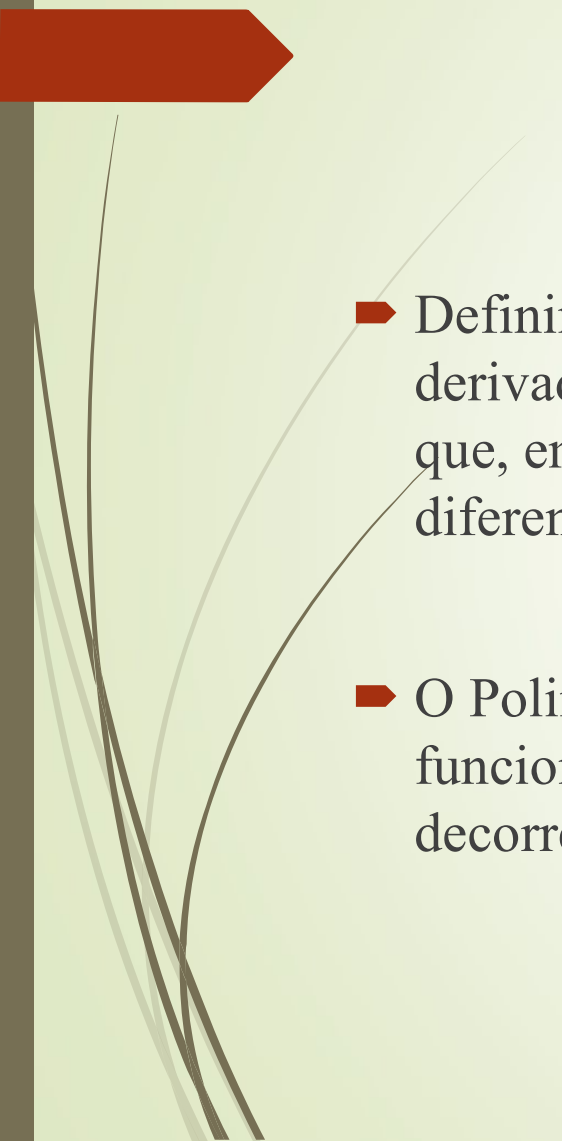


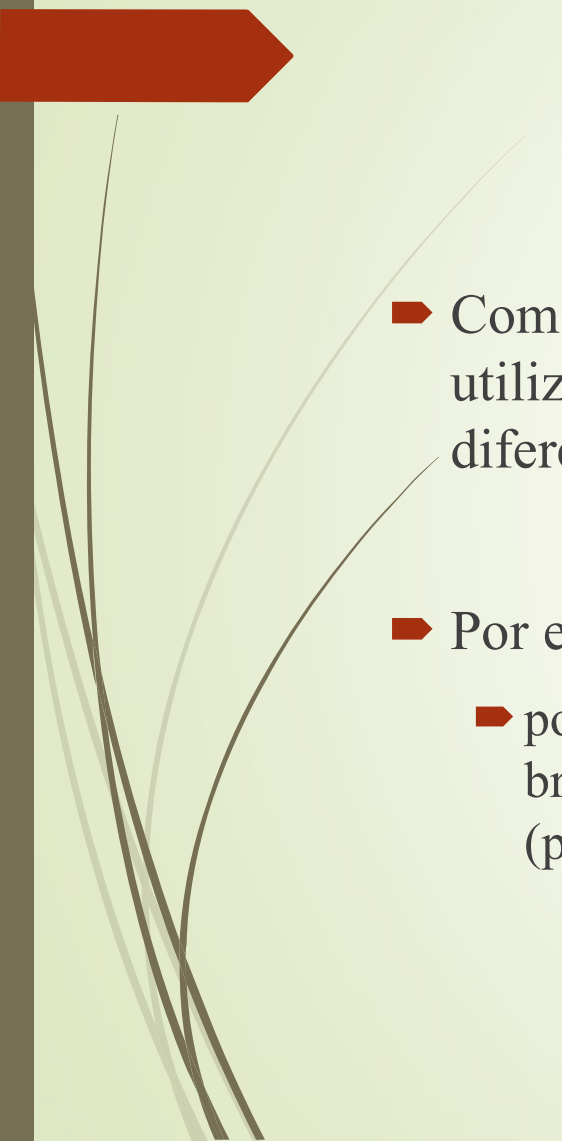


POO em Python

Polimorfismo

INE 5402 – Programa Orientada a Objetos I
Prof Luciana Rech

- 
- Definimos Polimorfismo como um princípio a partir do qual as classes derivadas de uma única classe base são capazes de invocar os métodos que, embora apresentem a mesma assinatura, comportam-se de maneira diferente para cada uma das classes derivadas.
 - O Polimorfismo é um mecanismo por meio do qual selecionamos as funcionalidades utilizadas de forma dinâmica por um programa no decorrer de sua execução.

- 
- Com o **Polimorfismo**, os mesmos atributos e objetos podem ser utilizados em objetos distintos, porém, com implementações lógicas diferentes.
 - Por exemplo:
 - podemos assumir que uma bola de futebol e uma camisa da seleção brasileira são artigos esportivos, mas o cálculo deles em uma venda (precificação) é efetuado de forma diferente.



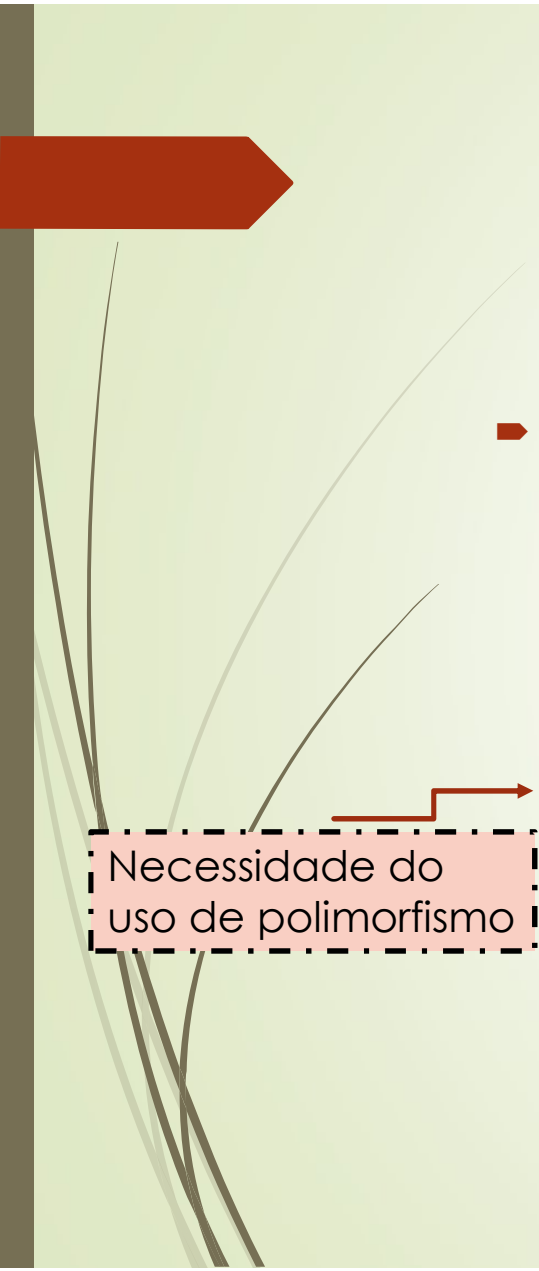
Polimorfismo no Python

- Polimorfismo, em Python, **é a capacidade que uma subclasse tem de ter métodos com o mesmo nome de sua superclasse, e o programa saber qual método deve ser invocado, especificamente (da super ou sub).** Ou seja, o objeto tem a capacidade de assumir diferentes formas (polimorfismo).
- É o que chamamos como polimorfismo de sobreposição.



Exemplo:

- Implemente um sistema que simule o abastecimento de veículos.
- Premissas:
 - Características relevantes: nome, marca, cor, quantidade de litros atualmente no tanque
 - Existem 2 tipos de veículos:
 - Veículos de uma forma geral (sem controle da capacidade do tanque)
 - Carros (com limite máximo de 50 litros no tanque)
- Seu sistema deve simular o abastecimento tanto de veículos , como de veículos do tipo Carro.



Necessidade do
uso de polimorfismo

- - Neste exemplo, eu posso criar uma instância de veículo e também posso criar uma instância de carro.
- Ambos objetos tem a habilidade de executar o método abastecer.
- Porém desejo inserir uma premissa: uma instância de Veículo não possui limite de capacidade para abastecer, enquanto que para uma instância de Carro existe o limite definido de 50 litros.

```
class Veiculo:
    def __init__(self, cor, marca, modelo, tanque):
        self.tanque = tanque
        self.modelo = modelo
        self.marca = marca
        self.cor = cor

    def abastecer(self, litros):
        print('Abastecendo...' + str(litros) + " litros")
        self.tanque += litros

    def __str__(self):
        print('O Veiculo: ' + self.marca + " " + self.modelo + " está com " +
              str(self.tanque) + " litros no tanque.")
```

```
from ex_polimorfismo_Veiculo import Veiculo

class Carro(Veiculo):
    def __init__(self, cor, marca, modelo, tanque, ano):
        super().__init__(cor, marca, modelo, tanque)
        self.ano = ano
        self.limit = 50

    def abastecer(self, litros):
        print('Tentando abastecer: ' + str(litros) + " litros")
        if self.tanque + litros > self.limit:
            print('Capacidade máxima alcançada, foi possível abastecer: ' +
                  str(self.limit - self.tanque) + ' litros.')
            self.tanque = self.limit
        else:
            self.tanque += litros
```

```
2 from ex_polimorfismo_Carro import Carro
3 from ex_polimorfismo_Veiculo import Veiculo
4
5 v1 = Veiculo('azul', 'Scania', 'R450', 10)
6 c1 = Carro('Grafite', 'Fiat', 'Argo', 25, 2020)
7
8 # instancia da classe Veiculo abastecendo
9 v1.__str__()
10 v1.abastecer(50)
11 v1.__str__()
12 v1.abastecer(30)
13 v1.__str__()
14 v1.abastecer(40)
15 v1.__str__()
16
17 print()
```

```
17 print()
18 # instância da classe Carro abastecendo
19 # a instancia c1 irá executar o método abastecer que está
20 #definido na Classe Carro (polimorfismo)
21 #irá executar o método __str__() na super classe Veiculo(herança)
22 c1.__str__()
23 c1.abastecer(10)
24 c1.__str__()
25 c1.abastecer(20)
26 c1.__str__()
27 c1.abastecer(40)
28 c1.__str__()
29
```


Exercício

- 1) Complementar o método abastecer do exemplo do Veículo/Carro:
 - 1.a) permitindo abastecer até encher o tanque. Enviar mensagem avisando que o tanque está cheio e quantos litros foi possível abastecer.
 - 1.b) criar um método para impressão dos atributos de veículos e carros (__str__())
 - 1.c) Simular a partir do programa principal situações de abastecimento e mostrar os relatórios.