Módulo 10

DTECH

Introdução a *Orientação a Objetos*

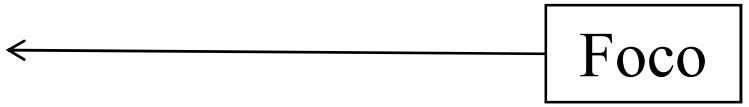
Algoritmos e Estruturas de Dados I

Python

(Rone Ilídio)

Orientação a Objetos

- Principais Conceitos
  - Classe
  - Herança
  - Encapsulamento
  - Polimorfismo



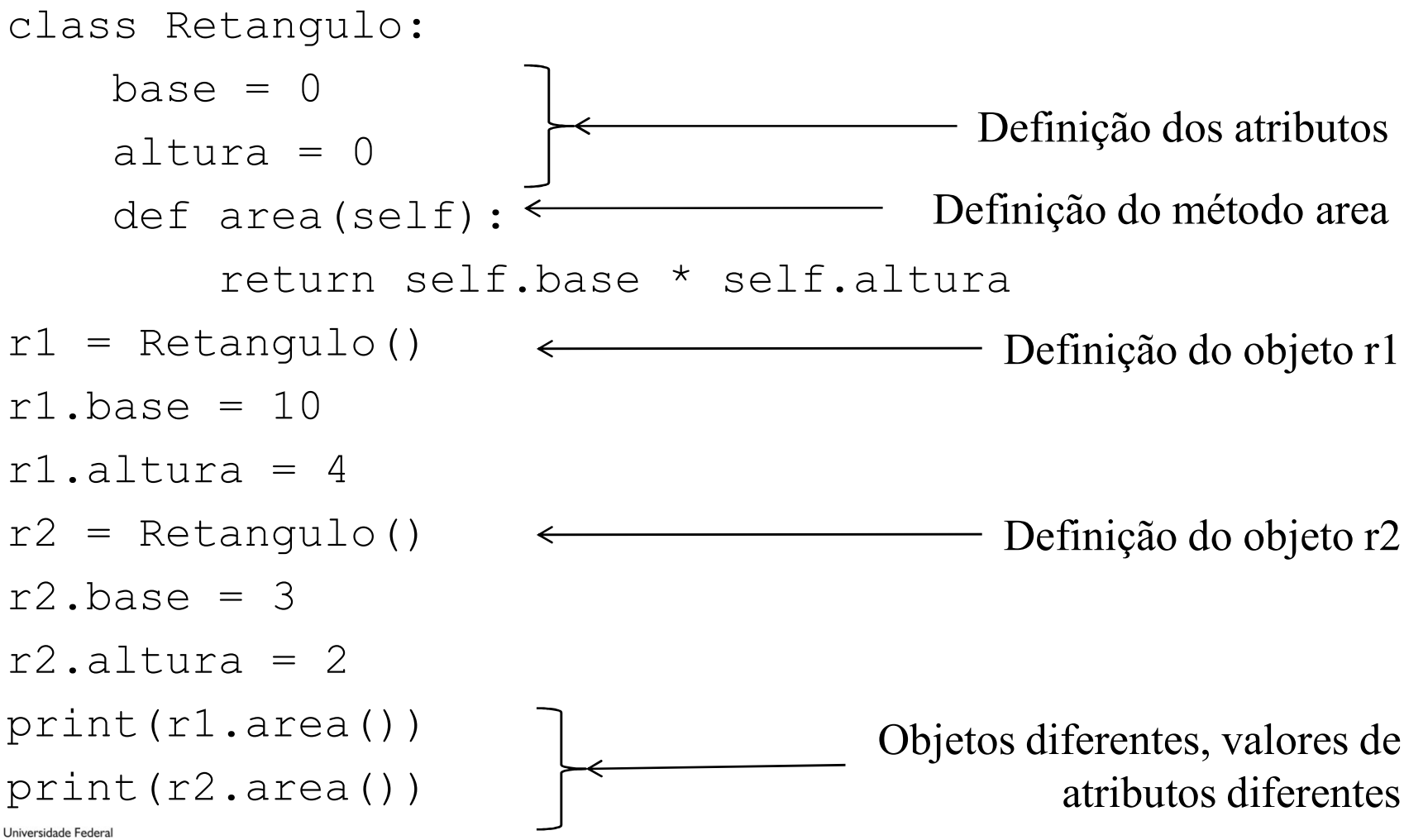
Classe

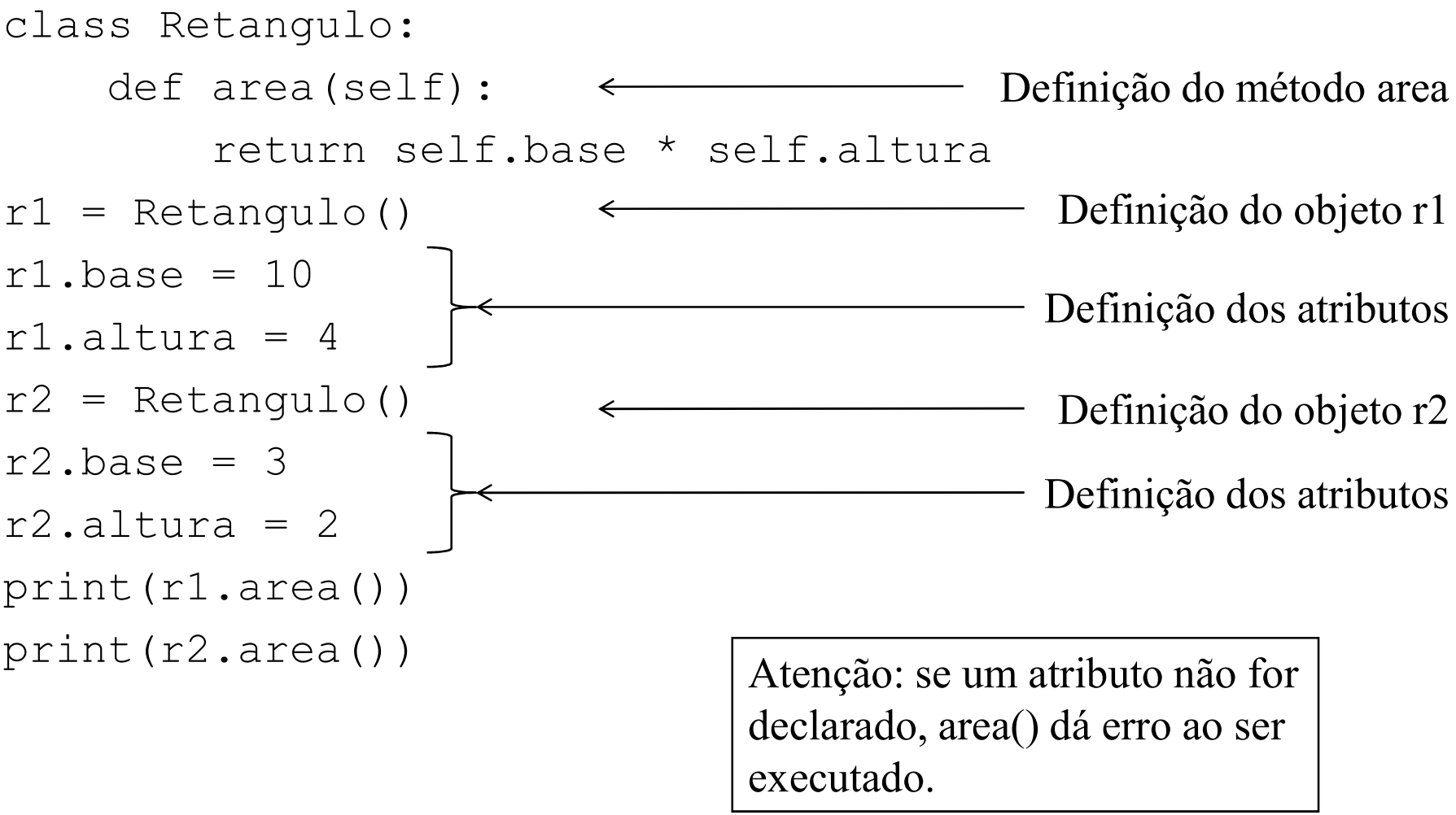
Classe

- Classe é um novo tipo criado pelo programador
- Uma classe é formada por atributos e métodos
  - Atributos: são variáveis
  - Métodos: são funções
- A criação de uma variável cujo tipo é uma classe recebe o nome de **instanciação**
- Essa variável é chamada de **Objeto**

Classe

- A classe funciona como um molde
- Uma classe modela no computador um objeto real que deseja-se manipular
- Cada objeto de uma mesma classe possui a mesma estrutura mas valores diferentes nos atributos


class Retangulo:
base = 0
altura = 0
def area(self):
return self.base * self.altura
r1 = Retangulo()
r1.base = 10
r1.altura = 4
r2 = Retangulo()
r2.base = 3
r2.altura = 2
print(r1.area())
print(r2.area())
Definição dos atributos
Definição do método area
Definição do objeto r1
Definição do objeto r2
Objetos diferentes, valores de atributos diferentes

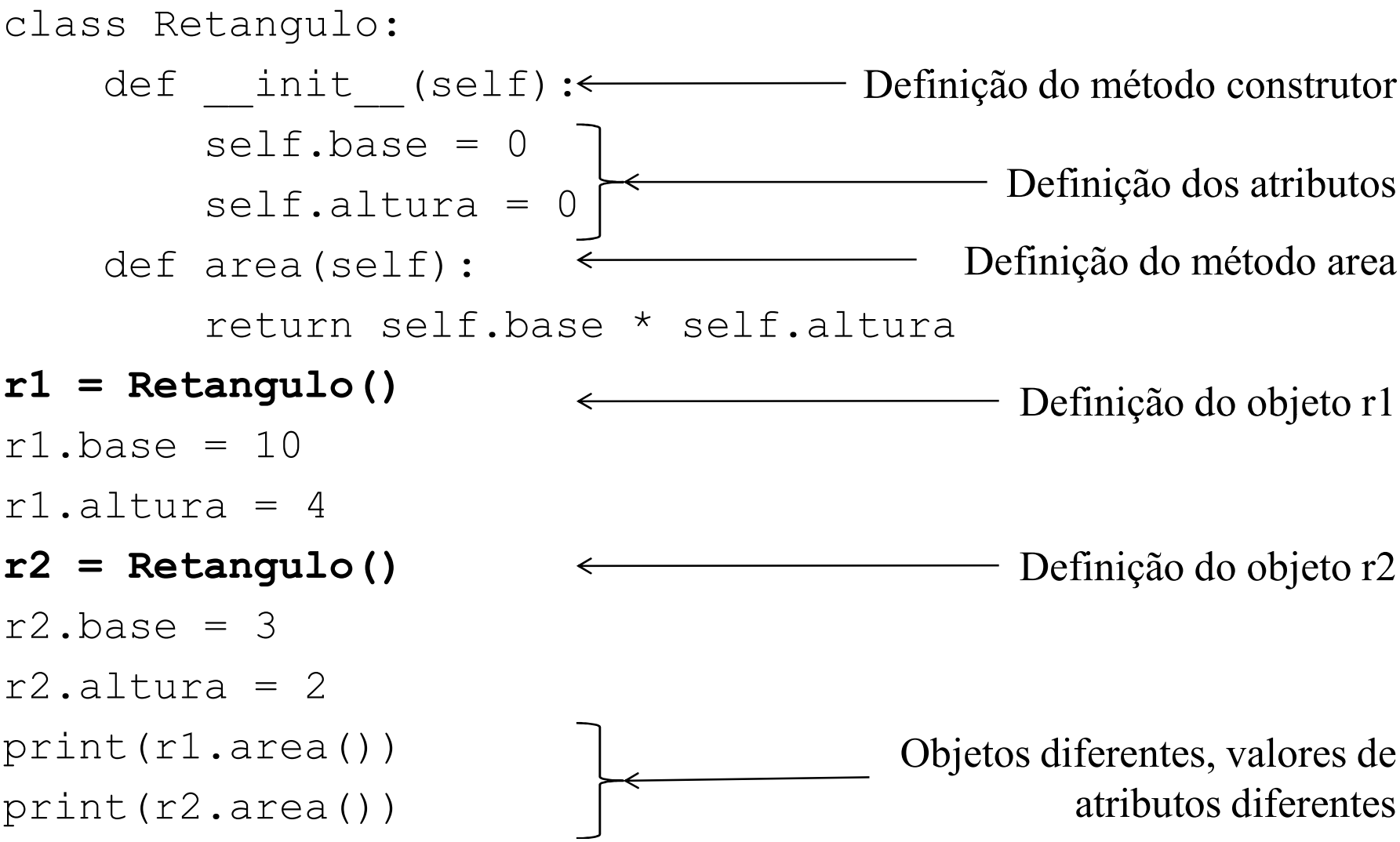


Atenção: se um atributo não for declarado, area() dá erro ao ser executado.

Método Construtor

Método Construtor

- Executado na criação do objeto
- Declarado com __init__()
- Se não for declarado, o Python entende como um construtor vazio (não recebe parâmetros e não possui código)


class Retangulo:
def __init__(self):
self.base = 0
self.altura = 0
def area(self):
return self.base * self.altura
r1 = Retangulo()
r1.base = 10
r1.altura = 4
r2 = Retangulo()
r2.base = 3
r2.altura = 2
print(r1.area())
print(r2.area())
Definição do método construtor
Definição dos atributos
Definição do método area
Definição do objeto r1
Definição do objeto r2
Objetos diferentes, valores de atributos diferentes

```
class Ponto:
    def __init__(self,x,y):
        self.x=x
        self.y=y
    def pontomedio(self, p):
        xm = (self.x + p.x)/2
        ym = (self.y + p.y)/2
        pm = Ponto(xm,ym)
        return pm
a = Ponto(10,10)
b = Ponto(20,20)
c = a.pontomedio(b)
print(c.x,',',c.y)
```

Definição do método construtor

Parâmetros obrigatórios na chamada do construtor

Objetos Como Parâmetros de Funções

Objetos Como Parâmetros de Funções

- Como classes são tipos, objetos podem sem passadas como parâmetros de funções (como qualquer outro tipo de dados)
- A passagem de um objeto é sempre por referência

Objetos Como Parâmetros de Funções

```
class Conta:
    def __init__(self,correntista,saldo):
        self.correntista = correntista
        self.saldo = saldo
def acrescentajuros(conta,juros):
    conta.saldo = conta.saldo * (1 + juros)
c = Conta('Zé Mané',1000)
acrescentajuros(c,0.1)
print(c.saldo)
```

Lista de Objetos

Lista de Objetos

- Lista onde cada elemento é um objeto
- Todos os objetos possuem a mesma estrutura e valores diferentes para seus atributos

```
class Venda:
    def __init__(self,vproduto=0, desconto=0, cliente=''):
        self.vproduto = vproduto
        self.desconto = desconto
        self.cliente = cliente
    def valorvenda(self):
        return self.vproduto * (1-self.desconto)
v1 = Venda(10,0.05,'Zé Mané')
v2 = Venda(15,0.1,'Maria')
lista = list()
lista.append(v1)
lista.append(v2)
for i in lista:
    print(i.valorvenda())
```

Exercícios

1. Crie uma classe denominada Pessoa, com os atributos nome, idade e email. Seu programa principal deve criar 2 objetos do tipo pessoa e preenche-los com dados do usuário. Exiba todos os dados na tela.
2. Crie uma classe Produto com os atributos nome, preço e estoque. Seu programa principal deve ter uma lista do tipo produto. Crie um menu com opções para: inserir, excluir, procurar e sair.
3. Crie uma classe chamada Triangulo (com base, altura e um método area) e uma classe chamada Círculo (com raio e um método area). Faça o usuário preencher uma lista com 5 Triângulos e outra com 5 círculo. Exiba na tela o triângulo e o círculo com maior área.

Exercício 1

```
class Pessoa:
    def __init__(self,nome,idade,email):
        self.nome = nome
        self.idade = idade
        self.email = email
p1 = Pessoa(input('Nome:'),int(input('Idade:')),input('Email:'))
p2 = Pessoa(input('Nome:'),int(input('Idade:')),input('Email:'))
print('Nome:',p1.nome,' Idade:',p1.idade,' Email:',p1.email)
print('Nome:',p2.nome,' Idade:',p2.idade,' Email:',p2.email)
```

Exercício 2


```
class Produto:
    def __init__(self,nome,preco,estoque):
        self.nome = nome
        self.preco = preco
        self.estoque = estoque

lista = list()
while True:
    print('\nDigite:\n1-Inserir produto')
    print('2-Excluir produto')
    print('3-Procurar produto')
    print('4-Sair')
    op = input('==>')
```

Exercício 2


```
if op == '1':
    n = input('Nome do produto:')
    p = float(input('Preço:'))
    e = float(input('Estoque:'))
    prod = Produto(n,p,e)
    lista.append(prod)
```

Exercício 2


```
  if op == '2':
      n = input('Digite o nome do produto a ser excl
uído:')
      encontrou = False
      for i in lista:
          if i.nome == n:
              print('Removido!')
              encontrou = True
              lista.remove(i)
      if not(encontrou):
          print('Produto não encontrado')
```

```
if op == '3':
    n = input('Digite o nome do produto procurado:')
    encontrou = False
    for i in lista:
        if i.nome == n:
            print('Nome:',i.nome)
            print('Preço:',i.preco)
            print('Estoque:',i.estoque)
            encontrou = True
    if not(encontrou):
        print('Produto não encontrado')
if op == '4':
    break
```

Exercício 2

Exercício 3


```
class Triangulo:
    def __init__(self,base,altura):
        self.base = base
        self.altura = altura
    def area(self):
        return (self.base * self.altura)/2
class Circulo:
    def __init__(self,raio):
        self.raio = raio
    def area(self):
        return 3.14 * self.raio ** 2
```

Exercício 3


```
ltri = list()
lcir = list()

for i in range(5):
    base = float(input('Informe a base:'))
    altura = float(input('Informe a altura:'))
    t = Triangulo(base, altura)
    ltri.append(t)

for i in range(5):
    raio = float(input('Informe o raio:'))
    c = Circulo(raio)
    lcir.append(c)
```

Exercício 3


```
m = 0
for i in range(5):
    if ltri[i].area() > ltri[m].area():
        m = i
print('Triângulo --
  > base:',ltri[m].base,' altura:',ltri[m].altura)
m = 0
for i in range(5):
    if lcir[i].area() > lcir[m].area():
        m = i
print('Círculo --> raio:',lcir[m].raio)
```