Fundamentos de Programação de Computadores

Python- Variáveis, expressões e funções

Docente: Fátima Leal

DCT DEPARTAMENTO CIÊNCIA E TECNOLOGIA

UPT UNIVERSIDADE PORTUCALENSE

Introdução ao Python

- Até agora vimos:
- Tipos de linguagens: alto e baixo nível.
- O Python é uma **linguagem interpretada**.
- Uma linguagem de **scripting**
- Primeiro Programa em Python: *Hello World*.
- Agora vamos estudar cada característica do Python.

Python: Variáveis

- **Variável** – nome que se refere a um valor que irá ser guardado. No Python, as variáveis não necessitam de ser declaradas
- **Atribuição** – instrução que dá valor a uma variável

```
nota = 17
disciplina = 'Fundamentos Prog de Computadores'


  print("Eu tive ", nota, " a ", disciplina)
```

```
================== RESTART: C:\Users\fatim\Desktop\example.py ==========
Eu tive  17  a  Fundamentos Prog de Computadores
>>>
```

Python: Variáveis

- É possível fazermos uma **reatribuição** à mesma variável.

```
>>> dia = 'Quinta-feira'
>>> dia
'Quinta-feira'
>>> dia = 29
>>> dia
29
```

Python: Variáveis

- Começam sempre com uma letra, seguidos de letras, números ou _
- Não podem ter espaços nem caracteres especiais
- Não podem ser **palavras reservadas** do Python

and	def	exec	if	not	return
assert	del	finally	import	or	try
break	elif	for	in	pass	while
class	else	from	is	print	yield
continue	except	global	lambda	raise	

Python: Variáveis e os tipos de dados

- **Float -** tipo de dados utilizado para guardar valores em vírgula flutuante;
- **Int -** tipo de dados utilizado para guardar valores inteiros.
- **Str -** tipo de dados utilizado para guardar sequências de carateres
- **bool** – tipo de dados utilizado para guardar valores booleanos (True e False)

Python: Variáveis e os tipos de dados

- Em Python podemos verificar o tipo de dados com a função ***type***.

```
>>> dia = 29
>>> type(dia)
<class 'int'>
>>>
```

```
>>> dia = 'Quinta-feira'
>>> type(dia)
<class 'str'>
>>>
```

```
>>> pi = 3.14
>>> type(pi)
<class 'float'>
>>>
```

Python: conversão de tipos de dados

- Em Python podemos fazer conversão de tipo de dados: *int, float, str.*

```
>>> pi = 3.14
>>> pi = int(pi)
>>> type(pi)
<class 'int'>
>>> pi
3
```

Conversão *float* para *int*

```
>>> a = 17
>>> float
<class 'float'>
>>> float (a)
17.0
```

Conversão *int* para *float*

```
>>> a = "2345"
>>> type(a)
<class 'str'>
>>> a = int("2345")
>>> type(a)
<class 'int'>
```

Conversão *str* para *int*

```
>>> int("23 alunos")
Traceback (most recent call last):
  File "<pyshell#33>", line 1, in <module>
    int("23 alunos")
ValueError: invalid literal for int() with base 10: '23 alunos'
```

Python: Instruções

- Instruções são sequências de comandos que o Python executa.

```
>>> dia = 'Quinta-feira'
>>> dia
'Quinta-feira'
>>> dia = 29
>>> dia
29
```

Atribuição

Pedir para mostrar o valor da variável dia

Python: Expressões

- Expressões são sequências de valores, variáveis, operadores ou chamadas a funções.

- O Python avalia a expressão e mostra o seu valor. O Python pode ser utilizado como **calculadora**.

```
>>> 10+10
20
>>>
```

- As variáveis e valores são por si só expressões. Ou seja, qual o resultado que o interpretador do Python irá mostrar para a sequência de expressões seguinte?

```
>>> x=10
>>> x= x+x
>>> x
```

Python: Variáveis, Expressões e Instruções

- Exemplos de expressões com operadores e operandos.

- Podemos ter operações com:
  1. Parêntises ()
  2. Exponenciação **
  3. Multiplicação, divisão, Resto da divisão e Divisão inteira * / % //
  4. Adição e subtração + -

```
>>> (5+9)*(15-7)
```

```
>>> 2**3
8
```

```
>>> 7%3
1
>>> 7//3
2
>>> 7/3
2.3333333333333335
```

```
>>> horas=10
>>> minutos=20
>>> horas*60+minutos
620
```

instruções

Variáveis

expressões

Python: Import de bibliotecas

- O Python disponibiliza **bibliotecas** que contém **funções** que poderemos reutilizar:
- Veremos alguns exemplos durantes as próximas aulas
- Para começar, vamos apenas explorar a biblioteca que disponibiliza **funções matemáticas (math)**
- Para utilizar estas funções precisamos de **importa-las (import)** para o nosso ambiente python.

```
>>>import math
>>>dir(math)
```

```
from math import func1, func2, ...
```

```
>>> from math import sqrt, pi
>>> 2*pi
6.283185307179586
>>>
```

Python: Entrada e saída de dados

- Todos os programas, por vezes precisam receber dados de alguma fonte. De seguida, processam-nos e mostram o respetivo resultado. Em Python:
  - **print()** – permite mostrar dados ao utilizador. Equivale ao comando Escrever da algoritmia.
  - **input()** – Recebe dados do utilizador.

```
nome = input ('Introduz o teu nome: ')

================== RESTART: C:\Users\fatim\Desktop\example.py ==========
Introduz o teu nome: Fátima Leal
```

Toma em atenção que todos os dados pedidos ao utilizador são devolvidos ao programa sempre em formato ***string***

Python: Entrada e saída de dados

- Para valores numéricos será necessário fazer uma conversão para ***int*** ou ***float***.

```
idade = input ('Introduz a tua idade: ')
idade = int(idade)
```

Python: Exemplo prático

- Como vimos podemos combinar: variáveis, expressões e instruções.

- Apliquemos os conceitos desenvolvendo um pequeno programa que determina a área da circunferência. O raio deverá ser introduzido pelo utilizador.

```
raio = input("Introduza o raio da circunferencia: ")
raio = float(raio)
pi = 3.14
area= pi*raio**2
print("Area da circunferencia calculada: ", area)
```

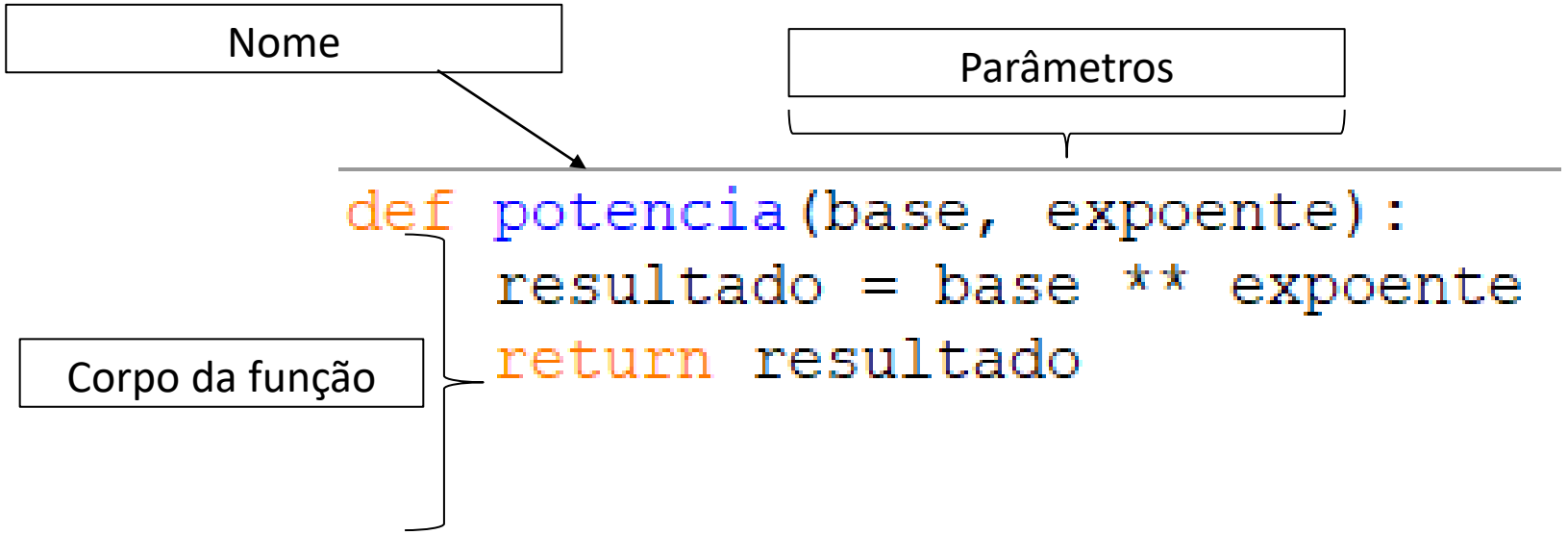
Ou, numa versão composta

```
raio = float (input("Introduza o raio da circunferencia: "))
pi = 3.14
print("Area da circunferencia calculada: ", pi*raio**2)
```

Python: Funções

- Funções são **blocos de código** que realizam determinadas tarefas que normalmente necessitam de ser executadas várias vezes dentro da aplicação.
- A sintaxe de uma função é definida por três partes: **nome, parâmetros e o seu corpo.**
- Em Python podemos implementar e executar funções isoladamente. Para isso teremos que utilizar a palavra reservada **def.**
- Analisemos o seguinte exemplo:

Python: Funções



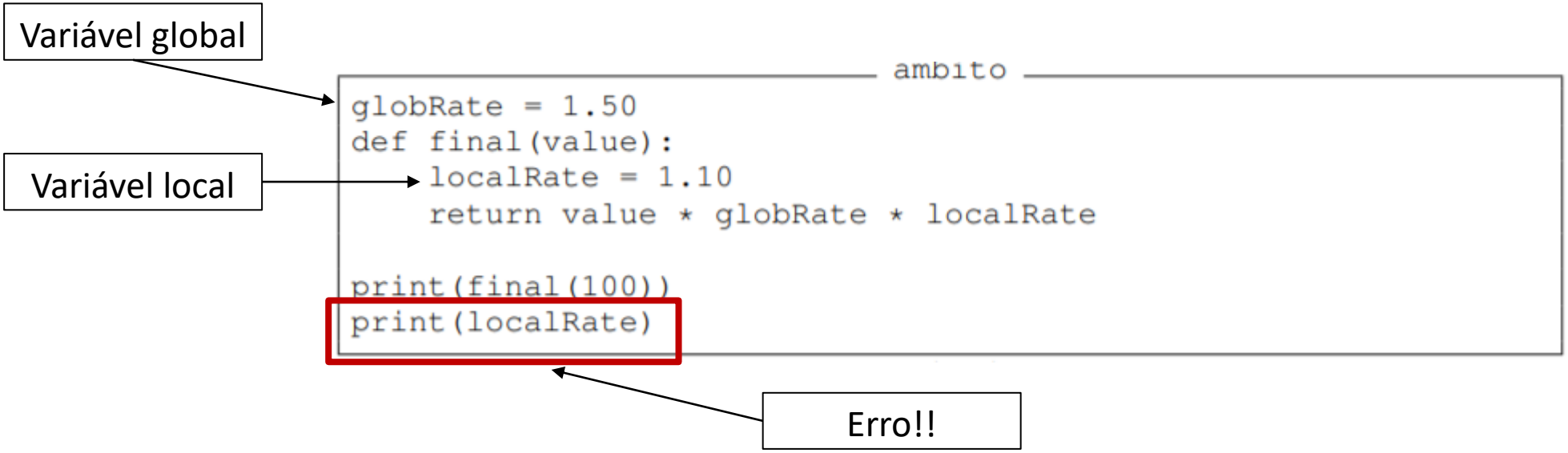
Nota que o corpo da função tem que estar alinhado com o seu nome.

Python: Funções

- Definir uma função não a executa
- Para a executarmos temos que a chamar
- As variáveis definidas na função são chamadas **variáveis locais**
  - **Apenas existem dentro da função**
  - **Desaparecem quando a função termina**
  - **Não podem ser usadas fora dela**
- Os parâmetros da função também são variáveis locais
- As funções podem chamar outras funções

Python: Variáveis locais e globais

- **âmbito local**: quando precisa do valor de uma variável, Python procura primeiro definições locais numa função;
- **âmbito global**: se não encontrar, procura no âmbito global (variáveis globais).



UNIVERSIDADE
PORTUCALENSE

Do conhecimento à prática.