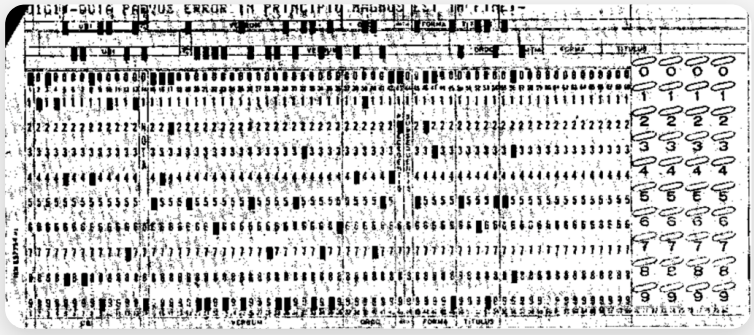
Filologia Digital - Módulo V

Para além da edição: o digital na investigação linguística

Sessão 1 - Conceitos básicos de Python



Índice

1. Funções e Parâmetros

📌 Objetivos do **Tópico 1**

✅ Compreender e utilizar Funções

- Saber definir funções em Python com `def()`.
- Entender a diferença entre **parâmetros** e **argumentos**.

🔢 Tipos de Dados Primitivos

- `int` → Números inteiros (ex: `5`, `-10`).
- `float` → Números decimais (ex: `3.14`, `-2.5`).
- `bool` → Valores lógicos (`True` ou `False`).

🔄 Instrução de Devolução (`return`)

- Compreender o uso da palavra-chave `return` para devolver valores de uma função.

➕➖ Operadores Aritméticos

Operador	Descrição	Exemplo
`+`	Adição	`2 + 3 → 5`
`-`	Subtração	`5 - 2 → 3`
`*`	Multiplicação	`4 * 3 → 12`
`/`	Divisão	`10 / 2 → 5.0`
`//`	Divisão inteira	`10 // 3 → 3`
`%`	Resto da divisão	`10 % 3 → 1`
`**`	Exponenciação	`2 ** 3 → 8`

🔁 Conversão Entre Tipos Numéricos

- `int(valor)` → Converte para inteiro.
- `float(valor)` → Converte para decimal.

⚖️ Operadores Relacionais

Operador	Significado	Exemplo
`==`	Igualdade	`5 == 5 → True`

Operador	Significado	Exemplo
`!=`	Diferente	`5 != 3 → True`
`<`	Menor que	`3 < 5 → True`
`>`	Maior que	`10 > 2 → True`
`<=`	Menor ou igual	`5 <= 5 → True`
`>=`	Maior ou igual	`7 >= 3 → True`

🔀 Operadores Lógicos

Operador	Significado	Exemplo
`not`	Negação	`not True → False`
`and`	Conjunção	`True and False → False`
`or`	Disjunção	`True or False → True`

1.1. Exercicios

📝 Exercício: Implementação de Funções

Desenvolva uma função para cada um dos objetivos apresentados nas alíneas seguintes. Todas as funções devem ser definidas utilizando apenas a instrução de retorno (`return ...`).

Para cada função, escreva expressões de teste que utilizem diferentes argumentos, garantindo que todas as possibilidades de resultados são abrangidas (por exemplo, verdadeiro/falso ou casos de arredondamento, como por excesso e por defeito).

✨ Exercício 1: Obter o dobro de um número.

A definição desta função é muito semelhante ao exemplo dado inicialmente.

In [ ]:

✨ Exercício 2: Obter a diferença entre dois números (dados em dois parâmetros).

In [ ]:

✨ Exercício 3: Obter o valor absoluto de um número.

```
# Exemplos:
    absolute(-2) → 2
```

```
absolute(4) → 4
```

In [ ]:

✨ Exercício 4: Obter a Percentagem de um Valor Relativamente a um Total

Dado um valor e um total, calcular a percentagem correspondente. O resultado deve ser um número real (`float`) no intervalo `[0.0, 1.0]` .

📌 **Exemplo:**
Se tivermos `2` num total de `8` , a percentagem será `0.25` (ou 25%).

```
def percentage(n, total):
    ...

# Testes
print(percentage(2, 8))   # 0.253
print(percentage(5, 20))  # 0.25
print(percentage(0, 10))  # 0.0
print(percentage(10, 0))  # 0.0 (caso de divisão por zero tratado)
```

In [ ]:

✨ Exercício 5: Obter a média entre dois números inteiros.

Note que a média entre dois inteiros não é necessariamente um inteiro, pelo que o tipo de retorno deverá ser decimal (`float`).

In [ ]:

✨ Exercício 6: Arredondar um número decimal para inteiro.

Nesta função é necessário aplicar a truncagem (converter um decimal para `inteiro` , descartando a parte decimal).

📌 **Exemplos:**

```
# Testes
    rounded(4.3) => 4
    rounded(4.7) => 5
```

In [ ]:

✨ Exercício 7: Saber se um número inteiro é negativo.

O valor devolvido pela função deverá indicar se o número passado como argumento é ou não negativo, mediante um valor booleano (`bool` , que assume apenas os valores True ou False).

📌 **Exemplos:**

```
def is_negative(n):
    ...

# Testes
    is_negative(-2) => True
    is_negative(3) => False
```

In [ ]:

✨ Exercício 8: Utilizando o operador aritmético de divisão inteira (%)

a) Saber se um número inteiro é impar.

In [ ]:

b) Saber se um número inteiro é par.

In [ ]:

c) Saber se um número inteiro é múltiplo de outro.

In [ ]:

✨ Exercício 9: Saber se um número inteiro corresponde a um dígito (i.e. está entre 0 e 9)

É necessário combinar duas condições lógicas usando o operador `and`.

In [ ]:

✨ Exercício 10: Saber se um número está incluído num dado intervalo (fechado).

```
# Exemplos:

    is_included(5, 4, 9) => True (5 está incluído em [4, 9])
    is_included(5, 6, 9) => False (5 não está incluído em [6, 9])
```

In [ ]:

✨ Exercício 11: Saber se um número está excluído de um dado intervalo (fechado).

Experimente uma solução com disjunção lógica e outra com negação da condição anterior.

```
# Exemplos:

    is_excluded(4, 4, 9) => False (4 não está excluído de [4, 9])
    is_excluded(4, 5, 9) => True (4 está excluído de [5, 9])
```

In [ ]:

✨ Exercício 12: Saber se um carácter corresponde a uma vogal minúscula.

Exemplos:

```
    is_vowel('a') => True
    is_vowel('z') => False
```

2. Funções e Parâmetros

🎯 Objetivos do **Tópico 2**

- **Instrução de atribuição**: `=`
  Atribui um valor a uma variável. Exemplo: `x = 5`, onde `x` recebe o valor 5.

- **Estrutura de seleção**: `if`

```
# Permite tomar decisões com base em condições.
# Exemplo:
if x > 10:
    print("Maior que 10")
```

- **Estrutura de repetição**: `while`

```
# Executa um bloco de código repetidamente enquanto a condição for verdadeira.
# Exemplo:
while x < 10:
    x += 1
```

2.1. Exercicios

✨ Exercício 1: Obter o quociente da divisão inteira, sem recorrer ao operador de divisão inteira (`//`).

```
# Exemplos:

        divide(31, 5) → 6
        divide(35, 7) → 5
```

In [ ]:

✨ Exercício 2: Obter uma potência de 2, i.e. 2^n (sem usar o operador `**`), dado o expoente num argumento que se assume ser um número natural *n* (diferente de zero).

```
# Exemplo:

        power_of_two(4) → 16
        power_of_two(8) → 256
```

In [ ]:

✨ Exercício 3: Obter o somatório dos números naturais até um dado *n*.

```
# Exemplo:

        sum_of_naturals_up_to(5) → 15      # (1 + 2 + 3 + 4 + 5)
```

In [ ]:

✨ Exercício 4: Obter o somatório dos números naturais pares compreendidos num dado intervalo fechado.

```
Exemplo:

        sum_of_even_numbers_between(3, 8) → 18      # (4 + 6 + 8)
```

In [ ]:

✨ Exercício 5: Obter o primeiro algarismo da representação decimal de um número inteiro.

```
Exemplos:

        first_digit(2011) → 2
        first_digit(1998) → 1
```

In [ ]:

✨ Exercício 6: Obter o n-ésimo número da **sequência de Fibonacci**, considerando que o primeiro número da sequência tem número de ordem zero.

Recorde que os primeiros números da sequência de Fibonacci são 0, 1, 1, 2, 3, 5, 8, 13, 21, ...

```
Exemplo:

        fibonacci(7) → 13
```

In [ ]:

Exercício extra

✨ Exercício 7: Defina uma função que recebe como argumento dois números naturais *m* e *n* e que calcula o máximo divisor comum desses dois números, usando o **algoritmo de Euclides**.

Exemplo:

```
gcd(25, 30) → 5
```

In [ ]: