

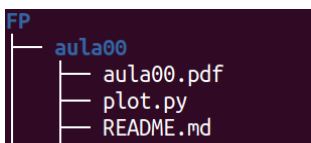
## Introdução à programação em Python

### Tópicos

- Instalação do Python
- O modo interativo e o modo script.
- Utilização da linha de comandos (num emulador de terminal)
- Edição e execução de programas em Python.

### Exercícios

1. Antes de começar, confirme que criou uma pasta (diretório) chamada FP (ou FundamentosProgramacao)<sup>1</sup> para guardar os materiais desta disciplina. E deve ter extraído a pasta da aula dentro de FP, para manter os materiais bem organizados.



2. No seu computador, siga as instruções para [instalar Python](#) que encontra na página da cadeira.
3. Abra uma janela de terminal (ver como [aqui](#)) e, na linha de comandos, introduza o comando `python3` para executar o Python em modo interativo.

```

jmr@darkmatter: ~
~$ python3
Python 3.8.2 (default, Jul 16 2020, 14:00:26)
[GCC 9.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
  
```

(Se aparecer um erro de “comando não encontrado” ou parecido, então o Python pode ter sido instalado com o nome `python` ou apenas `py`. Experimente.)

Em modo interativo, o interpretador de Python pode ser usado como uma calculadora. Introduza uma expressão para calcular quantos segundos tem um dia completo (use o asterisco `*` para indicar uma multiplicação). Experimente fazer outros cálculos simples.

4. Usando o Python em modo interativo, execute as seguintes instruções:

```

20-3
type(17)
1+2.3
type(1+2.3)
'Paris'
type('Paris')
'Paris'/2
# Dá erro! Qual? Porquê?
  
```

<sup>1</sup> É boa ideia evitar espaços e caracteres acentuados no nome das pastas e ficheiros.

A função `type(x)` permite determinar o tipo de dados do valor `x`. Para cada uma das expressões abaixo, tente prever o valor e tipo de dados (`int`, `float`, `str`, ...) do resultado, ou se dá erro. Depois confirme no Python.

Expressão	Valor	Tipo
<code>1 + 2 * 5</code>		
<code>17 / 3.0</code>		
<code>17 / 3</code>		
<code>17 // 3</code>		
<code>17 % 3.0</code>		
<code>5.0 / 0.75</code>		
<code>5.0 // 0.75</code>		
<code>'tau' + 'rus'</code>		
<code>'tau' + 2</code>		
<code>'tau' * 2</code>		

5. Em Python podemos guardar valores em variáveis para depois os reutilizar. Por exemplo, para guardar as dimensões de um retângulo pode usar as seguintes instruções.

```
largura = 21.0
altura = 29.7
```

Agora pode calcular a área do retângulo, guardá-la numa variável e mostrar o seu valor:

```
area = largura * altura
area
```

Consegue calcular, guardar e mostrar o perímetro? Dê um nome sugestivo à nova variável.

No final, termine o modo interativo do Python, carregando em `Ctrl-D` (ou `Ctrl-Z`, `Enter`, em Windows).

6. Abra um editor de texto (por exemplo, o “Bloco de notas”/Notepad do Windows, “Text editor”/gedit no Linux), reescreva as instruções que usou no exercício anterior e grave num ficheiro com o nome `retangulo.py`, dentro da pasta `aula00`. Acabou de criar um programa (script) em Python. Para o executar o programa, regresse ao terminal e introduza o comando:

```
python3 retangulo.py
```

Aparece um erro de “ficheiro não encontrado” ou parecido? Isso acontece porque o terminal não “está” a executar na pasta que contém o ficheiro. Para resolver isso terá de usar o comando `cd` para mudar de diretório (pesquise como usar ou pergunte ao professor). Também é útil aprender a usar o comando `ls` (ou `dir`) para listar o conteúdo da pasta.

Quando estiver na pasta certa, o comando não dará erro, mas é natural que não produza resultado nenhum, porque em modo script o Python não mostra os resultados automaticamente! No editor, corrija o programa para mostrar os resultados explicitamente usando a função `print`, grave e volte a executar o programa. Repita o

processo até o programa funcionar. No fim pode comparar o seu programa com a solução em `solution.py`.

7. Altere o programa anterior para *pedir* ao utilizador as dimensões do retângulo (usando a função `input`).<sup>2</sup> Corra o programa várias vezes, fornecendo dados diferentes de cada vez. Atenção: a função `input` devolve um valor de tipo `string`; tem de o converter para o tipo desejado usando a função `float`!
8. Execute o programa `welcome.py` para ver o que acontece. Modifique o programa para que o X seja substituído pelo curso indicado pelo utilizador.
9. O programa `plot.py` traça os gráficos de duas funções. Experimente executá-lo. Terá de fechar a janela para o terminar. Este programa usa *módulos* extra e funções poderosas que ainda não conhece. No entanto, se abrir o ficheiro, deve conseguir identificar algumas variáveis e expressões. Experimente colocar uma instrução `print` para imprimir o valor de alguma variável ou modifique um parâmetro de alguma expressão. Depois grave e volte a executar o para ver o que acontece.
10. Altere o programa anterior para gerar um terceiro gráfico com o produto das funções `y1` e `y2`. Trace o gráfico com linhas e bolas verdes. (Pesquise como se usa a função `plot` do módulo `matplotlib` ou peça ajuda aos colegas ou professor.)
11. Em FP recomendamos o livro eletrónico “How to Think Like a Computer Scientist, Interactive Edition”. Em casa, deverá ler as secções e resolver os exercícios recomendados no ficheiro `README.md`.

---

<sup>2</sup>Quando dizemos que o programa “*lê*” ou “*pede*” algum valor, quer dizer que tem de fazer `input` e o utilizador terá de introduzir um valor sempre que correr o programa.