

CADERNO DE EXERCÍCIOS

Criar aplicações em linguagem de programação Python

Caderno de Exercícios

Ação de formação	UC00618 — Criar aplicações em linguagem de programação Python (50 horas · 4,5 pontos de crédito)
Modalidade / Regime	Formação Modular — Ativos Empregados · Formação a Distância (Microsoft Teams)
Datas	08/10/2026 a 30/11/2026 · 2.ª e 5.ª feiras, 18:00–21:00
Formador	Luís Ferreira
Entidade formadora	IEFP, I.P. — Centro de Emprego da Maia
Contactos	Rua Dr. Carlos Felgueiras, 418 · 4470-157 Maia · Tel.: 220 989 190 · Email: ce.maia@iefp.pt
Referencial	https://catalogo.snq.gov.pt/ucDetalhe/355650



Cofinanciado pela
União Europeia

Os exercícios que se seguem devem ser resolvidos com os ficheiros do «10_Exercicios_Python.zip», que contém uma pasta por módulo com os scripts de partida (com # TODO) e a pasta dados/ com os ficheiros CSV, Excel e SQLite necessários. As soluções encontram-se no «11_Solucoes_Exercicios.zip» (formador). Entregue os exercícios de cada módulo no final da respetiva sessão, através do Teams.

 **Dica:** Descomprima o ZIP para uma pasta sem acentos nem espaços (ex.: C:\python_iefp) e abra essa pasta no VS Code (Ficheiro › Abrir Pasta). Os caminhos dos exercícios assumem que o terminal está nessa pasta.

Módulo 1 · Pensamento computacional e algoritmos

Exercício 1.1 — Média de notas em pseudocódigo

Escreva, em pseudocódigo, um algoritmo que leia três notas, calcule a média e indique «Aprovado» se a média for $\geq 9,5$ ou «Reprovado» caso contrário.

Ficheiro: m1_algoritmos/ex1_1_media.txt

Exercício 1.2 — Fluxograma par/ímpar

Desenhe (em papel ou numa ferramenta como o draw.io) o fluxograma de um algoritmo que lê um número e indica se é par ou ímpar. Descreva-o também em pseudocódigo no ficheiro indicado.

Ficheiro: m1_algoritmos/ex1_2_par_impar.txt

Exercício 1.3 — Decomposição de um problema

Decomponha o problema «criar um programa de gestão de uma pequena biblioteca» em subproblemas (o que é preciso guardar, que operações existem, que entradas e saídas há). Liste-os por ordem de prioridade.

Ficheiro: m1_algoritmos/ex1_3_decomposicao.txt

Módulo 2 · Python e ambiente de desenvolvimento

Exercício 2.1 — Primeiro script

Crie o script ola.py que pergunta o nome ao utilizador e imprime «Olá, <nome>! Bem-vindo(a) ao Python 3.14». Execute-o no terminal com `python ola.py` e também com o botão ► do VS Code.

Ficheiro: m2_ambiente/ex2_1_ola.py

Exercício 2.2 — Primeiro notebook

No VS Code, crie o notebook primeiro.ipynb com uma célula Markdown (título e o seu nome) e duas células de código: uma que calcula `2 ** 10` e outra que imprime a versão do Python (`import sys; print(sys.version)`).

Ficheiro: m2_ambiente/ex2_2_primeiro.ipynb

Exercício 2.3 — Terminal e pip

No terminal integrado do VS Code execute: `python --version`, `pip --version` e `pip list`. Instale o pacote requests com `pip install requests` e confirme com `pip show requests`. Registe os resultados no ficheiro indicado.

Ficheiro: m2_ambiente/ex2_3_terminal.txt

Módulo 3 · Fundamentos da linguagem

Exercício 3.1 — Calculadora de IVA

Peça ao utilizador um valor sem IVA e a taxa (6, 13 ou 23). Calcule o IVA e o total e mostre-os com duas casas decimais, usando f-strings.

Ficheiro: `m3_fundamentos/ex3_1_iva.py`

Exercício 3.2 — Manipulação de strings

Dada uma frase introduzida pelo utilizador, mostre: o número de caracteres, o número de vogais, a frase invertida e a frase em maiúsculas sem espaços.

Ficheiro: `m3_fundamentos/ex3_2_strings.py`

Exercício 3.3 — Listas de notas

A partir da lista `notas = [12, 15.5, 9, 18, 14, 7.5, 16]`, calcule a média, o máximo e o mínimo, ordene por ordem decrescente e crie, com uma compreensão de lista, a lista das notas positivas ($\geq 9,5$).

Ficheiro: `m3_fundamentos/ex3_3_listas.py`

Exercício 3.4 — Agenda com dicionários

Crie um dicionário que associa nomes a números de telemóvel. Implemente as operações: adicionar contacto, procurar por nome, remover contacto e listar todos os contactos por ordem alfabética.

Ficheiro: `m3_fundamentos/ex3_4_agenda.py`

Exercício 3.5 — Menu com match

Construa um menu de texto (1 — Somar, 2 — Subtrair, 3 — Multiplicar, 4 — Dividir, 0 — Sair) usando a instrução `match`. Trate a divisão por zero.

Ficheiro: `m3_fundamentos/ex3_5_menu_match.py`

Exercício 3.6 — Ciclos: adivinhar e FizzBuzz

a) Jogo de adivinhar: o programa escolhe um número de 1 a 100 (random) e o utilizador tenta acertar, recebendo as pistas «maior»/«menor» (ciclo `while`). b) FizzBuzz de 1 a 50 (ciclo `for`).

Ficheiro: `m3_fundamentos/ex3_6_ciclos.py`

Módulo 4 · Funções, ficheiros, iteradores e geradores

Exercício 4.1 — Funções com docstrings

Escreva as funções `area_circulo(raio)`, `celsius_para_fahrenheit(c)` e `e_primo(n)`, todas com docstring e anotações de tipo. Teste-as a partir de um bloco `if __name__ == "__main__":`.

Ficheiro: `m4_funcoes_ficheiros/ex4_1_funcoes.py`

Exercício 4.2 — Módulo próprio

Mova as funções do exercício 4.1 para o módulo `utilitarios.py` e importe-as no script principal. Adicione uma função `lambda` e use `map/filter` para converter e filtrar uma lista de temperaturas.

Ficheiro: `m4_funcoes_ficheiros/ex4_2_modulo.py`

Exercício 4.3 — Ficheiro de log

Leia o ficheiro `dados/servidor.log` e conte quantas linhas são INFO, WARNING e ERROR. Escreva um relatório em `relatorio.txt` com os totais e a lista das linhas ERROR. Trate o caso de o ficheiro não existir.

Ficheiro: `m4_funcoes_ficheiros/ex4_3_log.py`

Exercício 4.4 — Geradores

Crie um gerador `pares(limite)` que produz os números pares até ao limite e um gerador `ler_linhas(caminho)` que lê um ficheiro linha a linha sem o carregar todo em memória. Use-os para contar as linhas do log.

Ficheiro: `m4_funcoes_ficheiros/ex4_4_geradores.py`

Módulo 5 · Programação orientada a objetos

Exercício 5.1 — Classe ContaBancaria

Implemente a classe `ContaBancaria` com `titular`, `saldo`, os métodos `depositar`, `levantar` (não permitir saldo negativo) e `__str__`. Crie duas contas e simule movimentos.

Ficheiro: `m5_poo/ex5_1_conta.py`

Exercício 5.2 — Herança: veículos

Crie a classe `Veiculo` (`marca`, `modelo`, `ano`) e as subclasses `Carro` (`n.º de portas`) e `Moto` (`cilindrada`). Sobreponha o método `descricao()` e demonstre o polimorfismo numa lista de veículos.

Ficheiro: `m5_poo/ex5_2_heranca.py`

Exercício 5.3 — Decorador cronómetro

Escreva o decorador `@cronometro` que imprime o tempo de execução da função decorada. Aplique-o a uma função que soma os primeiros `N` números e a uma `dataclass` `Produto` com um método com desconto.

Ficheiro: `m5_poo/ex5_3_decorador.py`

Módulo 6 · Tratamento de dados

Exercício 6.1 — CSV com o módulo csv

Leia `dados/vendas.csv` com o módulo `csv` e calcule o total de vendas por região, sem usar `Pandas`.

Ficheiro: `m6_dados/ex6_1_csv.py`

Exercício 6.2 — Pandas: CSV

Com `Pandas`, leia `dados/vendas.csv`, mostre `head/info/describe`, calcule o total por região e por mês (`groupby`) e exporte o resumo para `resumo_vendas.xlsx`.

Ficheiro: `m6_dados/ex6_2_pandas_csv.py`

Exercício 6.3 — Pandas: Excel

Leia `dados/clientes.xlsx`, filtre os clientes do Porto com mais de 30 anos, crie a coluna «escalão etário» e grave o resultado em `clientes_porto.csv`.

Ficheiro: `m6_dados/ex6_3_pandas_excel.py`

Exercício 6.4 — SQL com sqlite3 e Pandas

Ligue-se a `dados/loja.sqlite`, liste as tabelas, execute uma consulta com JOIN entre encomendas e produtos e importe o resultado para um DataFrame com `read_sql`. Calcule o valor total por cliente.

Ficheiro: `m6_dados/ex6_4_sql.py`

Módulo 7 · Análise gráfica e bibliotecas

Exercício 7.1 — Gráficos com Matplotlib

A partir de `dados/vendas.csv`, crie um gráfico de barras com o total por mês e um gráfico circular com a distribuição por região. Adicione títulos e legendas e grave em PNG.

Ficheiro: `m7_graficos/ex7_1_matplotlib.py`

Exercício 7.2 — Turtle

Com a biblioteca `turtle`, desenhe um quadrado, um triângulo e um polígono de N lados (função `poligono(n, lado)`). Depois desenhe uma «flor» repetindo o polígono com rotações.

Ficheiro: `m7_graficos/ex7_2_turtle.py`

Exercício 7.3 — Interface gráfica (tkinter) ou jogo (Pygame)

Escolha uma opção: a) conversor de temperaturas com `tkinter` (caixa de texto, botão e etiqueta de resultado); b) bola que se move e ressalta nas paredes com `Pygame`.

Ficheiro: `m7_graficos/ex7_3_tkinter.py`

Módulo 8 · Testes, depuração, desempenho e documentação

Exercício 8.1 — Testes unitários

Escreva testes unitários (`unittest`) para as funções do módulo `utilitarios.py` (exercício 4.2), incluindo casos limite (raio 0, número 1, número negativo). Execute-os no terminal e no VS Code.

Ficheiro: `m8_testes/test_utilitarios.py`

Exercício 8.2 — Depuração

O script `bugs.py` contém quatro erros (sintaxe, lógica, tipo e índice). Use o depurador do VS Code para os encontrar e corrigir; comente cada correção.

Ficheiro: `m8_testes/ex8_2_bugs.py`

Exercício 8.3 — Benchmark e profiling

Compare, com `timeit`, duas formas de somar os quadrados de 1 a 1 000 000 (ciclo `for` vs. `sum` com gerador). Analise o script com `cProfile` e a memória com `tracemalloc`.

Ficheiro: `m8_testes/ex8_3_benchmark.py`

Exercício 8.4 — Manual de utilizador

Escreva o README.md do seu projeto final: objetivo, instalação (requisitos e pip install), como executar, exemplos de utilização e autor. Inclua docstrings em todas as funções do projeto.

Ficheiro: m8_testes/README_projeto.md

Projeto final (individual)

Desenvolva uma pequena aplicação em Python, à sua escolha, que integre pelo menos: funções, uma classe, leitura de dados de um ficheiro (CSV/Excel/SQL), um gráfico ou uma interface simples, testes unitários e um README com o manual de utilizador. Sugestões: gestor de despesas, análise de vendas da sua empresa, controlo de stocks, agenda de tarefas, jogo simples. Apresentação na sessão 16 (5 minutos).