



# pytest

**Ferramenta de Testes Automatizados**

# Ferramentas de testes automatizados

São programas ou sistemas de software utilizados para auxiliar desenvolvedores e testers de forma automatizada na execução e avaliação de testes em um aplicativo ou sistema.

Essas ferramentas são amplamente utilizadas no campo de desenvolvimento para a garantia de qualidade do software, para a redução de tempo e de esforços necessários para a execução de testes manuais.



# Pytest



## Origem

- Framework de testes em linguagem Python
- Criado por Holger Krekel
- Lançado em 2004.
- Surgiu pela necessidade de realizar testes mais simples e flexível
- <https://docs.pytest.org/en/7.4.x/>

## Popularidade

- Possui sintaxe clara e concisa
- Suporta testes
  - Unitários
  - Integração
  - Aceitação
- Altamente configurável por meio de plugins

# Desenvolvimento



**2004**

## Lançamento

Projeto de código aberto de uma ferramenta de teste simples e eficaz.



**2006**

## Aprimoramento

Foi necessário o desenvolvimento de novos recursos e seus aprimoramentos.



**2010**

## Popularização

Tornou-se uma das principais opções para testes em Python por sua simplicidade e capacidade de extensão



**Atualmente**

## Aplicação

Altamente desenvolvido e mantido, possui uma grande comunidade e uma ampla gama de plugins para extensões



# Utilização

Para a instalação do Pytest, é necessário inserir o seguinte código em seu terminal:

Logo em seguida, é possível verificar a versão instalada antes da aplicação.

Para sua utilização é necessário que possua o Python instalado.

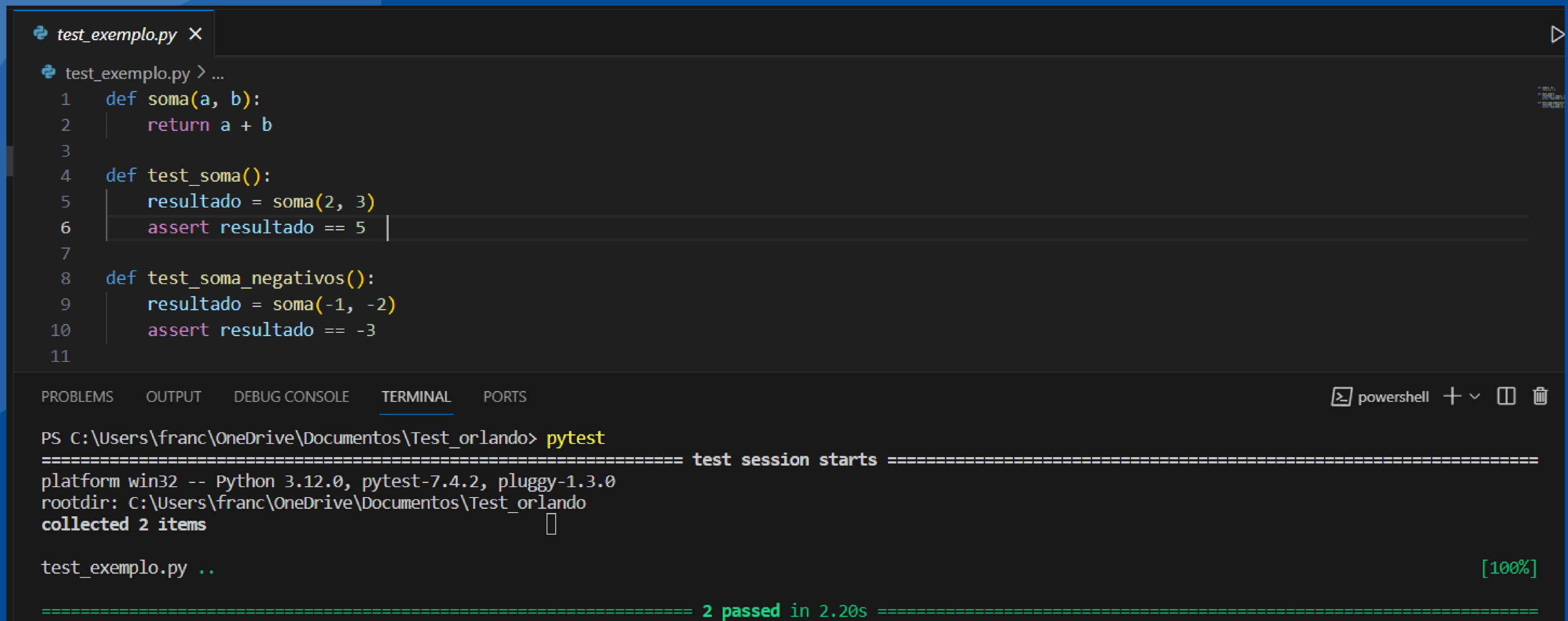
Link para a instalação:  
<https://www.python.org/downloads/>



```
PS C:\Users\franc> pip install -U pytest
Requirement already satisfied: pytest in c:\users\franc\appdata\local\programs\python\python312\lib\site-packages (7.4.2)
Requirement already satisfied: iniconfig in c:\users\franc\appdata\local\programs\python\python312\lib\site-packages (from pytest) (2.0.0)
Requirement already satisfied: packaging in c:\users\franc\appdata\local\programs\python\python312\lib\site-packages (from pytest) (23.2)
Requirement already satisfied: pluggy<2.0,>=0.12 in c:\users\franc\appdata\local\programs\python\python312\lib\site-packages (from pytest) (1.3.0)
Requirement already satisfied: colorama in c:\users\franc\appdata\local\programs\python\python312\lib\site-packages (from pytest) (0.4.6)
PS C:\Users\franc> pytest --version
pytest 7.4.2
```

# Exemplo Correto

Para iniciar um teste em que deseja utilizar o Pytest, é necessário que o documento seja denominado com o prefixo "test\_", já que a ferramenta procura por esses arquivos.



```
test_exemplo.py X
test_exemplo.py > ...
1  def soma(a, b):
2      return a + b
3
4  def test_soma():
5      resultado = soma(2, 3)
6      assert resultado == 5
7
8  def test_soma_negativos():
9      resultado = soma(-1, -2)
10     assert resultado == -3
11

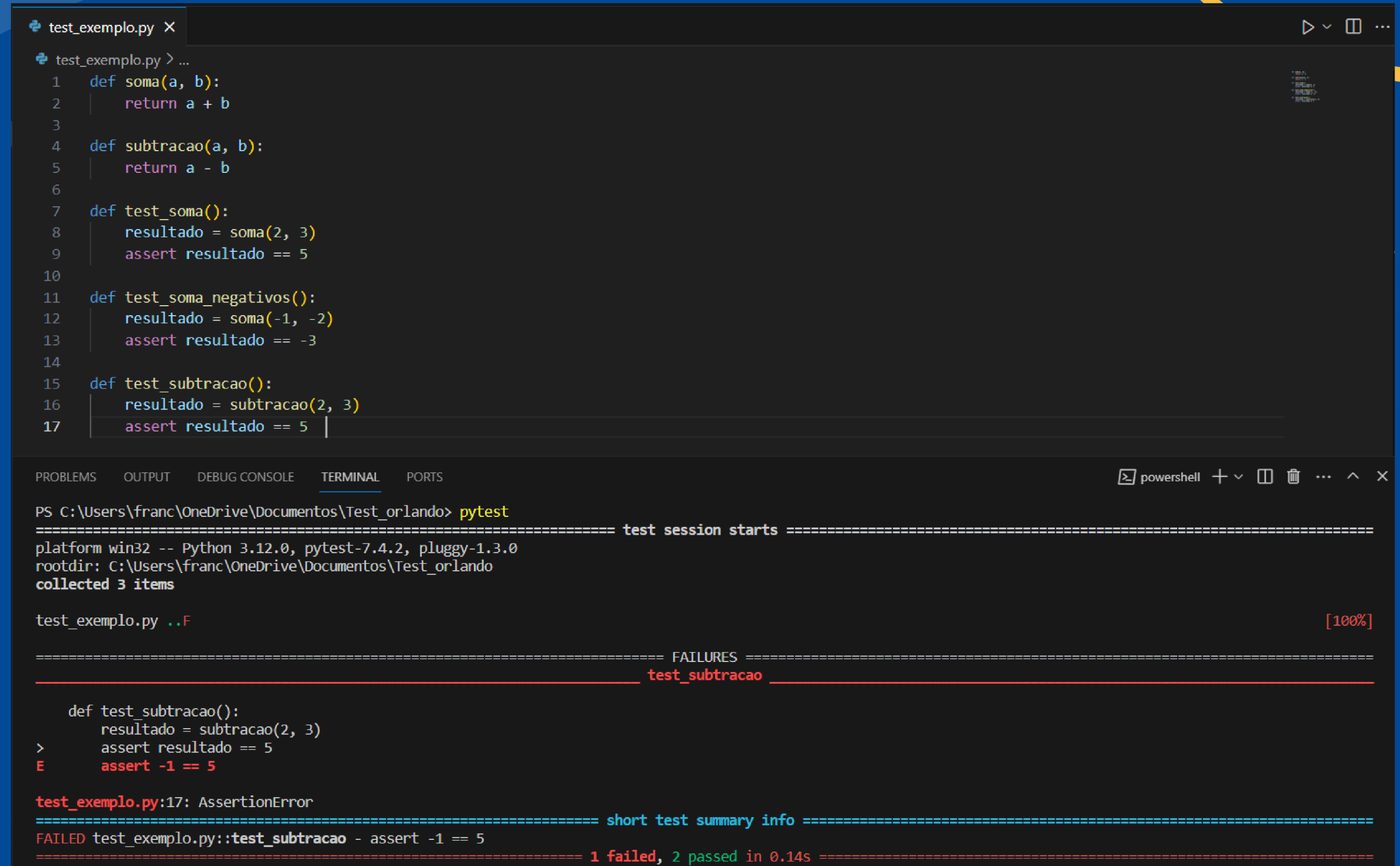
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
PS C:\Users\franc\OneDrive\Documentos\Test_orlando> pytest
===== test session starts =====
platform win32 -- Python 3.12.0, pytest-7.4.2, pluggy-1.3.0
rootdir: C:\Users\franc\OneDrive\Documentos\Test_orlando
collected 2 items

test_exemplo.py .. [100%]

===== 2 passed in 2.20s =====
```

# Exemplo Incorreto

Neste exemplo temos 2 testes que passam e 1 que é reprovado. Então é apresentado o resultado esperado e onde pode encontrar o erro para o ajuste



```
test_exemplo.py X
test_exemplo.py > ...
1 def soma(a, b):
2     return a + b
3
4 def subtracao(a, b):
5     return a - b
6
7 def test_soma():
8     resultado = soma(2, 3)
9     assert resultado == 5
10
11 def test_soma_negativos():
12     resultado = soma(-1, -2)
13     assert resultado == -3
14
15 def test_subtracao():
16     resultado = subtracao(2, 3)
17     assert resultado == 5

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\franc\OneDrive\Documentos\Test_orlando> pytest
===== test session starts =====
platform win32 -- Python 3.12.0, pytest-7.4.2, pluggy-1.3.0
rootdir: C:\Users\franc\OneDrive\Documentos\Test_orlando
collected 3 items

test_exemplo.py ..F [100%]

===== FAILURES =====
test_subtracao

def test_subtracao():
    resultado = subtracao(2, 3)
> assert resultado == 5
E assert -1 == 5

test_exemplo.py:17: AssertionError
===== short test summary info =====
FAILED test_exemplo.py::test_subtracao - assert -1 == 5
===== 1 failed, 2 passed in 0.14s =====
```

# Caroliny Cardoso de França

6º DSM - Qualidade e Testes de Software