### **MISSÃO 4 Q.A.**

Lucca Palhares 12137

Victor Ribeiro 46423

Guilherme Rodrigues 43860

### **Cenário 1:**

### **Função para Remover Duplicatas de uma Lista Mantendo a Ordem**

#### **1. Criação do Módulo**

Crie um arquivo Python chamado minhas\_funcoes.py e insira o seguinte código com a função remover\_duplicatas que contém um erro:

# minhas\_funcoes.py

def remover\_duplicatas(nums):

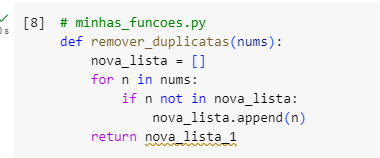
nova\_lista = []

for n in nums:

if n not in nova\_lista:

nova\_lista.append(n)

return nova\_lista



#### **2. Criação dos Testes Unitários**

Agora, crie outro arquivo chamado test\_minhas\_funcoes.py, onde você escreverá os testes para a função remover\_duplicatas.

# test\_minhas\_funcoes.py

import pytest

from minhas\_funcoes import remover\_duplicatas

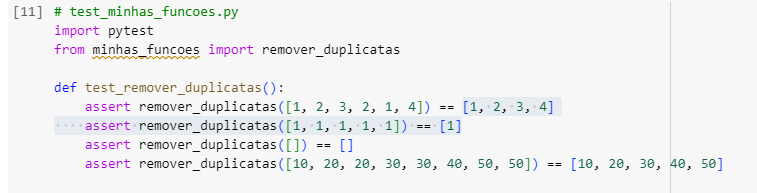
def test\_remover\_duplicatas():

assert remover\_duplicatas([1, 2, 3, 2, 1, 4]) == [1, 2, 3, 4]

assert remover\_duplicatas([1, 1, 1, 1, 1]) == [1]

assert remover\_duplicatas([]) == []

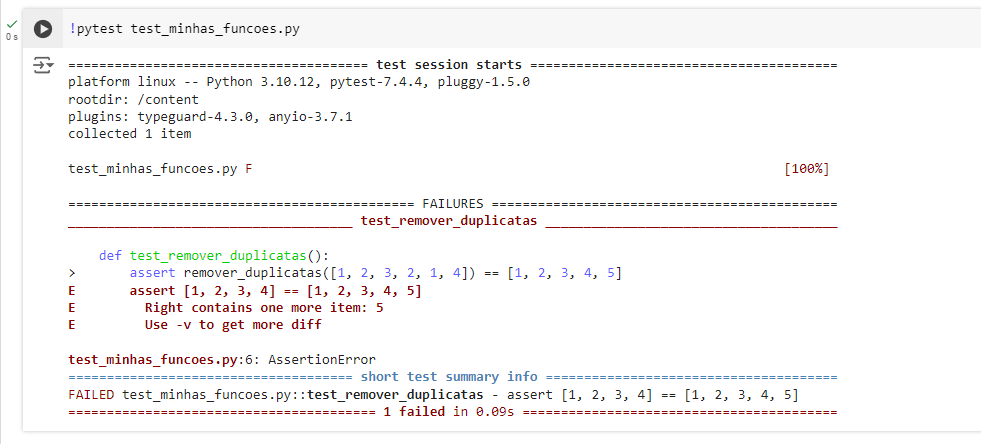
assert remover\_duplicatas([10, 20, 20, 30, 30, 40, 50, 50]) == [10, 20, 30, 40, 50]



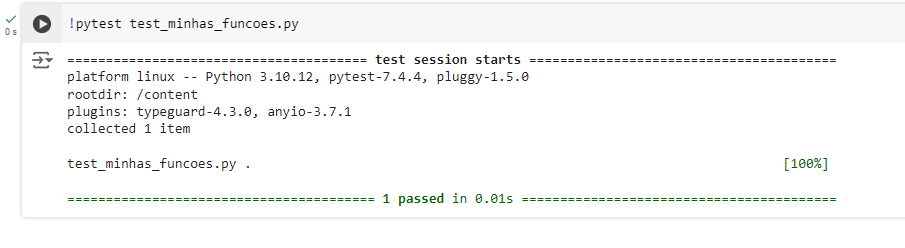
#### **3. Executando os Testes com erro**

No terminal, estando na mesma pasta onde os arquivos minhas\_funcoes.py e test\_minhas\_funcoes.py estão localizados, execute o seguinte comando para rodar os testes:

pytest test\_minhas\_funcoes.py



#### **3. Executando os Testes assertivo**



#### **4. Correção da Função**

Após rodar os testes, você pode perceber que a função funciona, mas é ineficiente para listas grandes. Vamos corrigi-la para usar um set, melhorando a eficiência:

# minhas\_funcoes.py

def remover\_duplicatas(nums):

seen = set()

nova\_lista = []

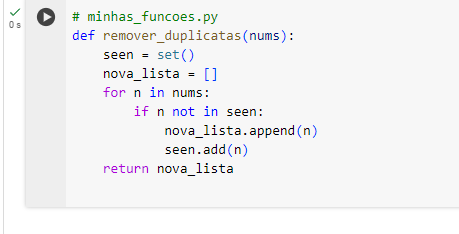
for n in nums:

if n not in seen:

nova\_lista.append(n)

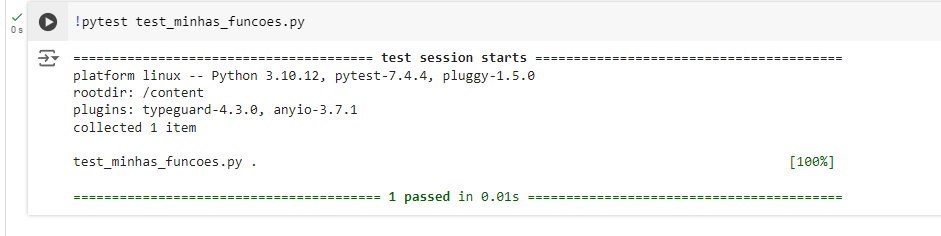
seen.add(n)

return nova\_lista



#### **5. Executando os Testes novamente**

No terminal, estando na mesma pasta onde os arquivos minhas\_funcoes.py e test\_minhas\_funcoes.py estão localizados, execute o seguinte comando para rodar os testes:pytest test\_minhas\_funcoes.py



### **Cenário 2: Função para Calcular o Fatorial de um Número**

#### **1. Criação do Módulo**

Adicione a função calcular\_fatorial ao arquivo minhas\_funcoes.py:

# minhas\_funcoes.py

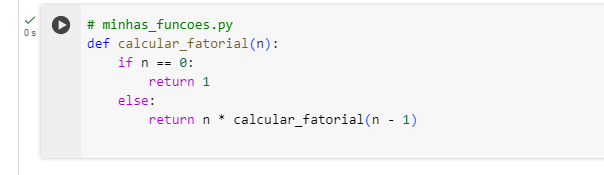
def calcular\_fatorial(n):

if n == 0:

return 1

else:

return n \* calcular\_fatorial(n - 1)



#### **2. Criação dos Testes Unitários**

Adicione os testes para a função calcular\_fatorial no arquivo test\_minhas\_funcoes.py:

# test\_minhas\_funcoes.py

import pytest

from minhas\_funcoes import calcular\_fatorial

def test\_calcular\_fatorial():

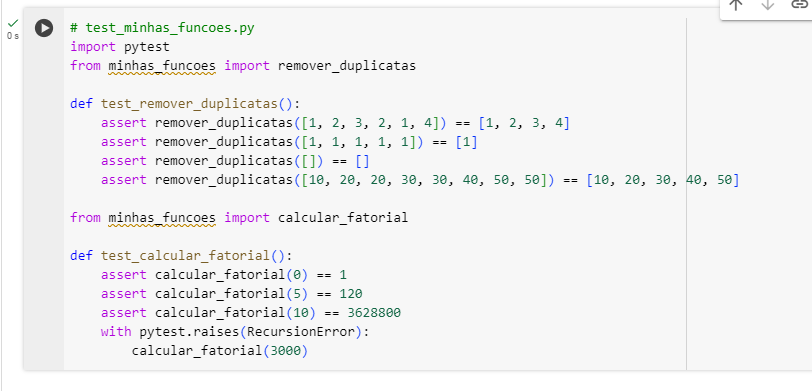
assert calcular\_fatorial(0) == 1

assert calcular\_fatorial(5) == 120

assert calcular\_fatorial(10) == 3628800

with pytest.raises(RecursionError):

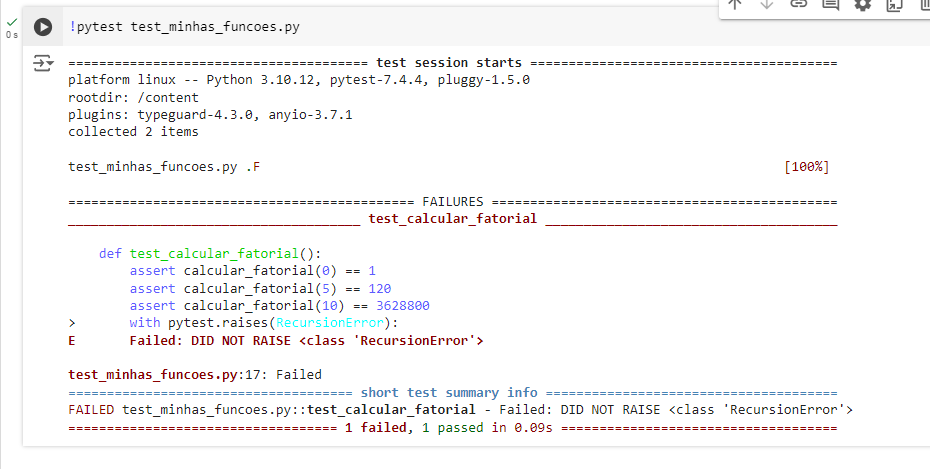
calcular\_fatorial(3000)



#### **3. Executando os Testes com erro**

No terminal, estando na mesma pasta onde os arquivos minhas\_funcoes.py e test\_minhas\_funcoes.py estão localizados, execute o seguinte comando para rodar os testes:

pytest test\_minhas\_funcoes.py



#### **4. Correção da Função**

Após rodar os testes, você pode perceber que a função funciona, mas é ineficiente para listas grandes. Vamos corrigi-la para usar um set, melhorando a eficiência:

# minhas\_funcoes.py

def remover\_duplicatas(nums):

seen = set()

nova\_lista = []

for n in nums:

if n not in seen:

nova\_lista.append(n)

seen.add(n)

return nova\_lista