



FGA 0238 - Testes de Software – Turma: 02

Semestre: 2023.2

Nome: Raquel Temóteo Eucaria Pereira da Costa

Matrícula: 202045268

Equipe: 10 - Assertivos

Atividade 4 – TDD

1 Funcionalidade

1.1 Identificação da Issue

Projeto: QuantiFGA

Issue: [#199](#)

1.2 Especificação

Descrição: Criar função que evite o uso de caminhos de forma hard coding

Objetivo: Tornar o código mais flexível para diferentes sistemas operacionais e Facilitar o manutenção

Tarefas: Usar variáveis ou configurações externas para passar os caminhos; Criar uma função que receba a string do caminho e do arquivo; Se o caminho não for encontrado deve receber uma mensagem de erro; Se o arquivo não for encontrado deve receber uma mensagem de erro

Comentários: Sugestão Usar biblioteca os

1.3 Descrição da Funcionalidade

Uma função que possibilita definir os caminhos de arquivos de forma que seja visível para qualquer sistema operacional e informe entradas inválidas e se não foi encontrado o arquivo.

1.4 Ciclos

Ciclo 1 - Verificar se a entrada é nula (class TestEncontrarArquivoCsv(unittest.TestCase)
Teste (test_encontrar_arquivo_csv_raises_value_error_for_null_input)

Ciclo 2 - Verificar se a entrada é uma string
Teste (test_encontrar_arquivo_csv_raises_value_error_para_entrada_nao_string)

Ciclo 3 - Verificar se a pasta existe
Teste (test_encontrar_arquivo_csv_folder_does_exist)

Ciclo 4 - Verificar se o arquivo foi encontrado
Teste (test_encontrar_arquivo_not_found)

2 Execução

2.1 Primeiro Ciclo - Verificar se as entradas na função são nulas.

Código base:

```
caminhoArquivoTDD.py u x
backend > caminhoArquivoTDD.py > ...
1 import os
2 import unittest
3
4 def encontrarArquivoCsv(nome_arquivo, pasta):
5     caminho_pasta = os.path.join(os.path.dirname(os.path.abspath(__file__)), pasta)
6
7     arquivos = os.listdir(caminho_pasta)
8
9     for arquivo in arquivos:
10         if nome_arquivo in arquivo:
11             return os.path.join(caminho_pasta, arquivo)
12
```

Código teste:

```
13 class TestEncontrarArquivoCsv(unittest.TestCase):
14     def test_encontrar_arquivo_csv_raises_value_error_for_null_input(self):
15         with self.assertRaises(ValueError) as context:
16             encontrarArquivoCsv(None, "pasta_qualquer")
17
18         self.assertIn("0 nome do arquivo e a pasta não podem ser nulos.", str(context.exception))
19
```

Execução do teste(falha):

```
eucaria@eucaria-Latitude-3420:~/Documentos/Github/2022-2-QuantifGA$ /bin/python3 /home/eucaria/Documentos/Github/2022-2-QuantifGA/backend/caminhoArquivoTDD.py
E
ERROR: test_encontrar_arquivo_csv_raises_value_error_for_null_input (__main__.TestEncontrarArquivoCsv)
Traceback (most recent call last):
  File "/home/eucaria/Documentos/Github/2022-2-QuantifGA/backend/caminhoArquivoTDD.py", line 16, in test_encontrar_arquivo_csv_raises_value_error_for_null_input
    encontrarArquivoCsv(None, "pasta_qualquer")
  File "/home/eucaria/Documentos/Github/2022-2-QuantifGA/backend/caminhoArquivoTDD.py", line 7, in encontrarArquivoCsv
    arquivos = os.listdir(caminho_pasta)
FileNotFoundError: [Errno 2] No such file or directory: '/home/eucaria/Documentos/Github/2022-2-QuantifGA/backend/pasta_qualquer'

Ran 1 test in 0.000s

FAILED (errors=1)
```

Refatoração:

```
20 def encontrarArquivoCsv(nome_arquivo, pasta):
21
22     if nome_arquivo is None or pasta is None:
23         raise ValueError("O nome do arquivo e a pasta não podem ser nulos.")
24
25     caminho_pasta = os.path.join(os.path.dirname(os.path.abspath(__file__)), pasta)
26
27     arquivos = os.listdir(caminho_pasta)
28
29     for arquivo in arquivos:
30         if nome_arquivo in arquivo:
31             return os.path.join(caminho_pasta, arquivo)
32
```

Execução do teste(sucesso):

```
eucaria@eucaria-Latitude-3420:~/Documentos/Github/2022-2-QuantifGA$ /bin/python3 /home/eucaria/Documentos/Github/2022-2-QuantifGA/back
ackend/caminhoArquivoTDD.py
:
-----
Ran 1 test in 0.000s
OK
```

2.2 Segundo Ciclo - Verificar se a entrada é uma string

Código base:

```
20 def encontrarArquivoCsv(nome_arquivo, pasta):
21
22     if nome_arquivo is None or pasta is None:
23         raise ValueError("O nome do arquivo e a pasta não podem ser nulos.")
24
25     caminho_pasta = os.path.join(os.path.dirname(os.path.abspath(__file__)), pasta)
26
27     arquivos = os.listdir(caminho_pasta)
28
29     for arquivo in arquivos:
30         if nome_arquivo in arquivo:
31             return os.path.join(caminho_pasta, arquivo)
32
```

Código teste:

```
33 class TestEncontrarArquivoCsv(unittest.TestCase):
34     def setUp(self):
35         self.pasta_teste = 'pasta_teste_temporaria'
36         os.makedirs(self.pasta_teste)
37
38     def tearDown(self):
39         os.rmdir(self.pasta_teste)
40
41     def test_encontrar_arquivo_csv_raises_value_error_para_entrada_nao_string(self):
42         with self.assertRaises(ValueError) as context:
43             encontrarArquivoCsv(123, self.pasta_teste)
44
45         self.assertIn("O nome do arquivo e a pasta devem ser strings.", str(context.exception))
46
```

Execução do teste(falha):

```
eucaria@eucaria-Latitude-3420:~/Documentos/Github/2022-2-QuantifGA$ /bin/python3 /home/eucaria/Documentos/Github/2022-2-QuantifGA/back
ackend/caminhoArquivoTDD.py
E
=====
ERROR: test_encontrar_arquivo_csv_raises_value_error_para_entrada_nao_string (__main__.TestEncontrarArquivoCsv)
Traceback (most recent call last):
  File "/home/eucaria/Documentos/Github/2022-2-QuantifGA/backend/caminhoArquivoTDD.py", line 43, in test_encontrar_arquivo_csv_rai
es_value_error_para_entrada_nao_string
    encontrarArquivoCsv(123, self.pasta_teste)
  File "/home/eucaria/Documentos/Github/2022-2-QuantifGA/backend/caminhoArquivoTDD.py", line 27, in encontrarArquivoCsv
    arquivos = os.listdir(caminho_pasta)
FileNotFoundError: [Errno 2] No such file or directory: '/home/eucaria/Documentos/Github/2022-2-QuantifGA/backend/pasta_teste_tempo
raria'
-----
Ran 1 test in 0.001s
FAILED (errors=1)
```

Refatoração:

```
47 def encontrarArquivoCsv(nome_arquivo, pasta):
48
49     if nome_arquivo is None or pasta is None:
50         raise ValueError("O nome do arquivo e a pasta não podem ser nulos.")
51
52     if not isinstance(nome_arquivo, str) or not isinstance(pasta, str):
53         raise ValueError("O nome do arquivo e a pasta devem ser strings.")
54
55     caminho_pasta = os.path.join(os.path.dirname(os.path.abspath(__file__)), pasta)
56
57     arquivos = os.listdir(caminho_pasta)
58
59     for arquivo in arquivos:
60         if nome_arquivo in arquivo:
61             return os.path.join(caminho_pasta, arquivo)
62
```

Execução do teste(sucesso):

```
FAILED (errors=1)
eucaria@eucaria-Latitude-3420:~/Documentos/Github/2022-2-QuantifGA$ /bin/python3 /home/eucaria/Documentos/Github/2022-2-QuantifGA/b
ackend/caminhoArquivoTDD.py
.
-----
Ran 1 test in 0.000s
OK
```

2.3 Terceiro Ciclo - Verificar se a pasta existe

Código base:

```
47 def encontrarArquivoCsv(nome_arquivo, pasta):
48
49     if nome_arquivo is None or pasta is None:
50         raise ValueError("O nome do arquivo e a pasta não podem ser nulos.")
51
52     if not isinstance(nome_arquivo, str) or not isinstance(pasta, str):
53         raise ValueError("O nome do arquivo e a pasta devem ser strings.")
54
55     caminho_pasta = os.path.join(os.path.dirname(os.path.abspath(__file__)), pasta)
56
57     arquivos = os.listdir(caminho_pasta)
58
59     for arquivo in arquivos:
60         if nome_arquivo in arquivo:
61             return os.path.join(caminho_pasta, arquivo)
62
```

Código teste:

```
63 class TestEncontrarArquivoCsv(unittest.TestCase):
64     def setUp(self):
65         self.pasta_teste = 'pasta_teste_temporaria'
66         os.makedirs(self.pasta_teste)
67
68     def tearDown(self):
69         os.rmdir(self.pasta_teste)
70
71     def test_encontrar_arquivo_csv_folder_does_exist(self):
72         with self.assertRaises(ValueError) as context:
73             encontrarArquivoCsv('arquivo.csv', 'pasta_inexistente')
74
75         self.assertIn("A pasta pasta_inexistente não foi encontrada.", str(context.exception))
76
```

Execução do teste(falha):

```
eucaria@eucaria-Latitude-3420:~/Documentos/Github/2022-2-QuantifGA$ /bin/python3 /home/eucaria/Documentos/Github/2022-2-QuantifGA/backend/caminhoArquivoTDD.py
E
=====
ERROR: test_encontrar_arquivo_csv_folder_does_exist (__main__.TestEncontrarArquivoCsv)
Traceback (most recent call last):
  File "/home/eucaria/Documentos/Github/2022-2-QuantifGA/backend/caminhoArquivoTDD.py", line 73, in test_encontrar_arquivo_csv_folder_does_exist
    encontrarArquivoCsv('arquivo.csv', 'pasta_inexistente')
  File "/home/eucaria/Documentos/Github/2022-2-QuantifGA/backend/caminhoArquivoTDD.py", line 57, in encontrarArquivoCsv
    arquivos = os.listdir(caminho_pasta)
FileNotFoundError: [Errno 2] No such file or directory: '/home/eucaria/Documentos/Github/2022-2-QuantifGA/backend/pasta_inexistente'

-----
Ran 1 test in 0.000s
FAILED (errors=1)
```

Refatoração:

```
78 def encontrarArquivoCsv(nome_arquivo, pasta):
79
80     if nome_arquivo is None or pasta is None:
81         raise ValueError("O nome do arquivo e a pasta não podem ser nulos.")
82
83     if not isinstance(nome_arquivo, str) or not isinstance(pasta, str):
84         raise ValueError("O nome do arquivo e a pasta devem ser strings.")
85
86     caminho_pasta = os.path.join(os.path.dirname(os.path.abspath(__file__)), pasta)
87
88     if not os.path.exists(caminho_pasta):
89         raise ValueError(f"A pasta {pasta} não foi encontrada.")
90
91     arquivos = os.listdir(caminho_pasta)
92
93     for arquivo in arquivos:
94         if nome_arquivo in arquivo:
95             return os.path.join(caminho_pasta, arquivo)
```

Execução do teste(sucesso):

```
eucaria@eucaria-Latitude-3420:~/Documentos/Github/2022-2-QuantifGA$ /bin/python3 /home/eucaria/Documentos/Github/2022-2-QuantifGA/backend/caminhoArquivoTDD.py
.
-----
Ran 1 test in 0.000s
OK
```

2.4 Quarto Ciclo - Verificar se o arquivo existe

Código base:

```
78 def encontrarArquivoCsv(nome_arquivo, pasta):
79
80     if nome_arquivo is None or pasta is None:
81         raise ValueError("O nome do arquivo e a pasta não podem ser nulos.")
82
83     if not isinstance(nome_arquivo, str) or not isinstance(pasta, str):
84         raise ValueError("O nome do arquivo e a pasta devem ser strings.")
85
86     caminho_pasta = os.path.join(os.path.dirname(os.path.abspath(__file__)), pasta)
87
88     if not os.path.exists(caminho_pasta):
89         raise ValueError(f"A pasta {pasta} não foi encontrada.")
90
91     arquivos = os.listdir(caminho_pasta)
92
93     for arquivo in arquivos:
94         if nome_arquivo in arquivo:
95             return os.path.join(caminho_pasta, arquivo)
```

Código teste:

```
97 class TestEncontrarArquivoCsv(unittest.TestCase):
98     def setUp(self):
99         self.pasta_teste = 'pasta_teste_temporaria'
100         os.makedirs(self.pasta_teste)
101
102     def tearDown(self):
103         os.rmdir(self.pasta_teste)
104
105     def test_encontrar_arquivo_not_found(self):
106         resultado = encontrarArquivoCsv('arquivo_inexistente.csv', self.pasta_teste)
107
108         mensagem_esperada = f'0 arquivo arquivo_inexistente.csv não foi encontrado na pasta {self.pasta_teste}'
109         self.assertEqual(resultado, mensagem_esperada)
110
```

Execução do teste(falha):

```
eucaria@eucaria-Latitude-3420:~/Documentos/Github/2022-2-QuantifGA/backend$ python3 -m unittest caminhoArquivoTDD.py
F
FAIL: test_encontrar_arquivo_not_found (caminhoArquivoTDD.TestEncontrarArquivoCsv)
Traceback (most recent call last):
  File "/home/eucaria/Documentos/Github/2022-2-QuantifGA/backend/caminhoArquivoTDD.py", line 109, in test_encontrar_arquivo_not_fou
nd
    self.assertEqual(resultado, mensagem_esperada)
AssertionError: None != '0 arquivo arquivo_inexistente.csv não fo[40 chars]aria'
-----
Ran 1 test in 0.000s

FAILED (failures=1)
```

Refatoração:

```
111 def encontrarArquivoCsv(nome_arquivo, pasta):
112
113     if nome_arquivo is None or pasta is None:
114         raise ValueError("0 nome do arquivo e a pasta não podem ser nulos.")
115
116     if not isinstance(nome_arquivo, str) or not isinstance(pasta, str):
117         raise ValueError("0 nome do arquivo e a pasta devem ser strings.")
118
119     caminho_pasta = os.path.join(os.path.dirname(os.path.abspath(__file__)), pasta)
120
121     if not os.path.exists(caminho_pasta):
122         raise ValueError(f"A pasta {pasta} não foi encontrada.")
123
124     arquivos = os.listdir(caminho_pasta)
125
126     for arquivo in arquivos:
127         if nome_arquivo in arquivo:
128             return os.path.join(caminho_pasta, arquivo)
129
130     return f'0 arquivo {nome_arquivo} não foi encontrado na pasta {pasta}'
131
```

Execução do teste(sucesso):

```
eucaria@eucaria-Latitude-3420:~/Documentos/Github/2022-2-QuantifGA/backend$ python3 -m unittest caminhoArquivoTDD.py
.
Ran 1 test in 0.000s

OK
```

3 Código Fonte Testes

Versão final do código:

```
caminhoArquivoTDD.py X
backend > caminhoArquivoTDD.py ? ...
1 import os
2 import unittest
3
4 def encontrarArquivoCsv(nome_arquivo, pasta):
5     caminho_pasta = os.path.join(os.path.dirname(os.path.abspath(__file__)), pasta)
6
7     arquivos = os.listdir(caminho_pasta)
8
9     for arquivo in arquivos:
10         if nome_arquivo in arquivo:
11             return os.path.join(caminho_pasta, arquivo)
12
13 class TestEncontrarArquivoCsv(unittest.TestCase):
14     def test_encontrar_arquivo_csv_raises_value_error_for_null_input(self):
15         with self.assertRaises(ValueError) as context:
16             encontrarArquivoCsv(None, "pasta_qualquer")
17
18         self.assertIn("O nome do arquivo e a pasta não podem ser nulos.", str(context.exception))
19
20 def encontrarArquivoCsv(nome_arquivo, pasta):
21
22     if nome_arquivo is None or pasta is None:
23         raise ValueError("O nome do arquivo e a pasta não podem ser nulos.")
24
25     caminho_pasta = os.path.join(os.path.dirname(os.path.abspath(__file__)), pasta)
26
27     arquivos = os.listdir(caminho_pasta)
28
29     for arquivo in arquivos:
30         if nome_arquivo in arquivo:
31             return os.path.join(caminho_pasta, arquivo)
32
33 class TestEncontrarArquivoCsv(unittest.TestCase):
34     def setUp(self):
35         self.pasta_teste = 'pasta_teste_temporaria'
36         os.makedirs(self.pasta_teste)
37
38     def tearDown(self):
39         os.rmdir(self.pasta_teste)
40
41     def test_encontrar_arquivo_csv_raises_value_error_para_entrada_nao_string(self):
42         with self.assertRaises(ValueError) as context:
43             encontrarArquivoCsv(123, self.pasta_teste)
44
45         self.assertIn("O nome do arquivo e a pasta devem ser strings.", str(context.exception))
46
47 def encontrarArquivoCsv(nome_arquivo, pasta):
48
49     if nome_arquivo is None or pasta is None:
50         raise ValueError("O nome do arquivo e a pasta não podem ser nulos.")
51
52     if not isinstance(nome_arquivo, str) or not isinstance(pasta, str):
53         raise ValueError("O nome do arquivo e a pasta devem ser strings.")
54
55     caminho_pasta = os.path.join(os.path.dirname(os.path.abspath(__file__)), pasta)
56
57     arquivos = os.listdir(caminho_pasta)
58
59     for arquivo in arquivos:
60         if nome_arquivo in arquivo:
61             return os.path.join(caminho_pasta, arquivo)
62
```

```
63
64 class TestEncontrarArquivoCsv(unittest.TestCase):
65     def setUp(self):
66         self.pasta_teste = 'pasta_teste_temporaria'
67         os.makedirs(self.pasta_teste)
68
69     def tearDown(self):
70         os.rmdir(self.pasta_teste)
71
72     def test_encontrar_arquivo_csv_folder_does_exist(self):
73         with self.assertRaises(ValueError) as context:
74             encontrarArquivoCsv('arquivo.csv', 'pasta_inexistente')
75
76         self.assertIn("A pasta pasta_inexistente não foi encontrada.", str(context.exception))
77
78 def encontrarArquivoCsv(nome_arquivo, pasta):
79
80     if nome_arquivo is None or pasta is None:
81         raise ValueError("O nome do arquivo e a pasta não podem ser nulos.")
82
83     if not isinstance(nome_arquivo, str) or not isinstance(pasta, str):
84         raise ValueError("O nome do arquivo e a pasta devem ser strings.")
85
86     caminho_pasta = os.path.join(os.path.dirname(os.path.abspath(__file__)), pasta)
87
88     if not os.path.exists(caminho_pasta):
89         raise ValueError(f"A pasta {pasta} não foi encontrada.")
90
91     arquivos = os.listdir(caminho_pasta)
92
93     for arquivo in arquivos:
94         if nome_arquivo in arquivo:
95             return os.path.join(caminho_pasta, arquivo)
96
97 class TestEncontrarArquivoCsv(unittest.TestCase):
98     def setUp(self):
99         self.pasta_teste = 'pasta_teste_temporaria'
100         os.makedirs(self.pasta_teste)
101
102     def tearDown(self):
103         os.rmdir(self.pasta_teste)
104
105     def test_encontrar_arquivo_not_found(self):
106         resultado = encontrarArquivoCsv('arquivo_inexistente.csv', self.pasta_teste)
107
108         mensagem_esperada = f'O arquivo arquivo_inexistente.csv não foi encontrado na pasta {self.pasta_teste}'
109         self.assertEqual(resultado, mensagem_esperada)
110
111 def encontrarArquivoCsv(nome_arquivo, pasta):
112
113     if nome_arquivo is None or pasta is None:
114         raise ValueError("O nome do arquivo e a pasta não podem ser nulos.")
115
116     if not isinstance(nome_arquivo, str) or not isinstance(pasta, str):
117         raise ValueError("O nome do arquivo e a pasta devem ser strings.")
118
119     caminho_pasta = os.path.join(os.path.dirname(os.path.abspath(__file__)), pasta)
120
121     if not os.path.exists(caminho_pasta):
122         raise ValueError(f"A pasta {pasta} não foi encontrada.")
123
124     arquivos = os.listdir(caminho_pasta)
125
126     for arquivo in arquivos:
127         if nome_arquivo in arquivo:
128             return os.path.join(caminho_pasta, arquivo)
129
130     return f'O arquivo {nome_arquivo} não foi encontrado na pasta {pasta}'
131
132
```

4 Resultado Final Execução Testes

```
ncif@eucaria-Latitude-3420:~/Documents/Github/2022-2-QuantifGA/backend$ python3 -m unittest caminhoArquivoTDD.py
.
Ran 1 test in 0.001s

OK
```



5 Código Fonte da Funcionalidade Implementada

```
✓ adicionando a função TDD
raqueleucaria comprometida 9 minutos atrás
1 pai fac7287 cometer a28cc95
Mostrando 1 arquivo alterado com 131 adições e 0 exclusões.
Espaço em branco Ignorar espaços em branco Dividir Unificado

131 backend/caminhoArquivoTDD.py
---
00 -0,0 +1.131 00
1 + import sys
2 + import os
3 + import unittest
4 + import sys
5 + def encontrarArquivoCsv ( nome_arquivo, pasta ):
6 +     caminho_pasta = os . caminho . join ( os . path . dirname ( os . path . abspath ( __file__ )), pasta )
7 +     arquivos = os . listdir ( caminho_pasta )
8 +     para arquivo em arquivos :
9 +         if nome_arquivo no arquivo :
10 +             retornar os . caminho . juntar ( caminho_pasta , arquivo )
11 +
12 + classe TestEncontrarArquivoCsv ( unittest.TestCase ) : _
13 + def test_encontrar_arquivo_csv_raises_value_error_for_null_input ( self ) :
14 +     consigo mesmo . assertRaises ( ValueError ) como contexto :
15 +         encontrarArquivoCsv ( None , "pasta_qualquer" )
16 +
17 + auto . assertIn ( "0 nome do arquivo e a pasta não podem ser nulos." , str ( context . exceção ))
18 +
19 + def encontrarArquivoCsv ( nome_arquivo, pasta ):
20 +
21 + se nome_arquivo for None ou pasta for None :
22 +     raise ValueError ( "0 nome do arquivo e a pasta não podem ser nulos." )
23 +
24 + caminho_pasta = os . caminho . join ( os . path . dirname ( os . path . abspath ( __file__ )), pasta )
25 +
26 + arquivos = os . listdir ( caminho_pasta )
27 +
28 + para arquivo em arquivos :
29 +     if nome_arquivo no arquivo :
30 +         retornar os . caminho . juntar ( caminho_pasta , arquivo )
31 +
32 +
33 + classe TestEncontrarArquivoCsv ( unittest.TestCase ) : _
34 + def configuração ( self ) :
35 +     auto . pasta_teste = 'pasta_teste_temporaria'
36 +     os . makedirs ( self . pasta_teste )
37 +
38 + def tearDown ( self ) :
39 +     os . rmdir ( self . pasta_teste )
40 +
41 + def test_encontrar_arquivo_csv_raises_value_error_para_entrada_nao_string ( self ) :
42 +     consigo mesmo . assertRaises ( ValueError ) como contexto :
43 +         encontrarArquivoCsv ( 123 , self . pasta_teste )
44 +
45 + auto . assertIn ( "0 nome do arquivo e a pasta devem ser strings." , str ( context . exceção ))
46 +
47 + def encontrarArquivoCsv ( nome_arquivo, pasta ):
48 +
49 + se nome_arquivo for None ou pasta for None :
50 +     raise ValueError ( "0 nome do arquivo e a pasta não podem ser nulos." )
51 +
52 + se não for isinstance ( nome_arquivo , str ) ou não for isinstance ( pasta , str ) :
53 +     raise ValueError ( "0 nome do arquivo e a pasta devem ser strings." )
54 +
55 + caminho_pasta = os . caminho . join ( os . path . dirname ( os . path . abspath ( __file__ )), pasta )
56 +
57 + arquivos = os . listdir ( caminho_pasta )
58 +
59 + para arquivo em arquivos :
60 +     if nome_arquivo no arquivo :
61 +         retornar os . caminho . juntar ( caminho_pasta , arquivo )
62 +
63 + classe TestEncontrarArquivoCsv ( unittest.TestCase ) : _
64 + def configuração ( self ) :
65 +     auto . pasta_teste = 'pasta_teste_temporaria'
66 +     os . makedirs ( self . pasta_teste )
67 +
68 + def tearDown ( self ) :
69 +     os . rmdir ( self . pasta_teste )
70 +
71 + def test_encontrar_arquivo_csv_folder_does_exist ( self ) :
72 +     consigo mesmo . assertRaises ( ValueError ) como contexto :
73 +         encontrarArquivoCsv ( 'arquivo.csv' , 'pasta_inexistente' )
74 +
75 + auto . assertIn ( "Uma pasta pasta_inexistente não foi encontrada." , str ( context . exceção ))
76 +
77 +
78 + def encontrarArquivoCsv ( nome_arquivo, pasta ):
79 +
80 + se nome_arquivo for None ou pasta for None :
81 +     raise ValueError ( "0 nome do arquivo e a pasta não podem ser nulos." )
82 +
83 + se não for isinstance ( nome_arquivo , str ) ou não for isinstance ( pasta , str ) :
84 +     raise ValueError ( "0 nome do arquivo e a pasta devem ser strings." )
85 +
86 + caminho_pasta = os . caminho . join ( os . path . dirname ( os . path . abspath ( __file__ )), pasta )
87 +
88 + se não for . caminho . existe ( caminho_pasta ) :
89 +     raise ValueError ( f"Uma pasta { pasta } não foi encontrada." )
90 +
91 + arquivos = os . listdir ( caminho_pasta )
92 +
93 + para arquivo em arquivos :
94 +     if nome_arquivo no arquivo :
95 +         retornar os . caminho . juntar ( caminho_pasta , arquivo )
96 +
97 + classe TestEncontrarArquivoCsv ( unittest.TestCase ) : _
98 + def configuração ( self ) :
99 +     auto . pasta_teste = 'pasta_teste_temporaria'
100 +     os . makedirs ( self . pasta_teste )
101 +
102 + def tearDown ( self ) :
103 +     os . rmdir ( self . pasta_teste )
104 +
105 + def test_encontrar_arquivo_not_found ( self ) :
106 +     resultado = encontrarArquivoCsv ( 'arquivo_inexistente.csv' , self . pasta_teste )
107 +
108 + mensagem_esperada = f"0 arquivo arquivo_inexistente.csv não foi encontrado na pasta { self . pasta_teste }"
109 +     auto . assertEquals ( resultado , mensagem_esperada )
110 +
111 +
```




```
110 +
111 + def encontrarArquivoCsv ( nome_arquivo , pasta ) :
112 +
113 +     se nome_arquivo for None ou pasta for None :
114 +         raise ValueError ( "O nome do arquivo e a pasta não podem ser nulos." )
115 +
116 +     se não for isinstance ( nome_arquivo , str ) ou não for isinstance ( pasta , str ) :
117 +         raise ValueError ( "O nome do arquivo e a pasta devem ser strings." )
118 +
119 +     caminho_pasta = os . caminho . join ( os . path . dirname ( os . path . abspath ( __file__ ) ) , pasta )
120 +
121 +     se não for . caminho . existe ( caminho_pasta ) :
122 +         raise ValueError ( f"Uma pasta { pasta } não foi encontrada." )
123 +
124 +     arquivos = os . listdir ( caminho_pasta )
125 +
126 +     para arquivo em arquivos :
127 +         if nome_arquivo no arquivo :
128 +             retornar os . caminho . juntar ( caminho_pasta , arquivo )
129 +
130 +     return f"O arquivo { nome_arquivo } não foi encontrado na pasta { pasta } "
131 +
```

Link do projeto: <https://github.com/raqueleucaria/2022-2-QuantiFGA/>

Link do commit:

<https://github.com/fga-eps-mds/2022-2-QuantiFGA/commit/a20cc357c961f5f88b6a4c58de46b40cec4c8596?diff=unified&w=0>

6 Pull Request

Link do PR: <https://github.com/fga-eps-mds/2022-2-QuantiFGA/pull/201>

Adicionando função de gerar caminho #201

raqueleucaria wants to merge 3 commits into `fga-eps-mds:main` from `raqueleucaria:main`

Conversation 0 · Commits 3 · Checks 2 · Files changed 2 +132 -0

raqueleucaria commented 21 minutes ago Member

A função cria o caminho para o diretório desejado da seguinte forma:

- Utiliza a biblioteca os
- Verifica as entradas (nulas e não string)
- Verifica se o caminho e o arquivo existe
- Utiliza a metodologia TDD

O intuito é que ela seja usada posteriormente no restante do código.

[#199](#)

raqueleucaria and others added 3 commits 8 hours ago

- add gitignore** `f4c7207`
- adicionando a funcao TDD** `a20cc35` Verified
- Merge pull request #1 from raqueleucaria/funcaoCaminho** `1b6d6d7` Verified

raqueleucaria added `tests` `backend` labels 21 minutes ago

raqueleucaria self-assigned this 21 minutes ago

Add more commits by pushing to the `main` branch on `raqueleucaria/2022-2-QuantiFGA`.

Reviewers
No reviews
[Still in progress? Convert to draft](#)

Assignees
raqueleucaria

Labels
`backend` `tests`

Projects
None yet

Milestone
No milestone

Development
Successfully merging this pull request may close these issues.
None yet

Notifications Customize
[Unsubscribe](#)
You're receiving notifications because you authored the thread.



7 Conclusão

A incorporação do Desenvolvimento Orientado por Testes (TDD) transformou minha abordagem no desenvolvimento de software. Ao pensar mais nos requisitos e escrever testes antes da implementação das funcionalidades, não apenas assegurou a qualidade do código, mas também adotou uma mentalidade proativa em relação às possíveis falhas. Essa prática não apenas proporciona feedback imediato, mas também exerce uma influência positiva na arquitetura do código, gerando soluções mais robustas e testáveis.