DESENVOLVIMENTO ORIENTADO A TESTES (TDD)

INF1301 – Programação Modular

TDD em Python

Aluno.py

```
def insere_aluno(matricula, nome):
      matricula = matricula.strip()
      nome = nome.strip()
      if matricula == '' or nome == '':
            return 2
      for aluno in alunos:
            if aluno['matricula'] == matricula:
                  return 1
      novo_aluno = {'matricula': matricula, 'nome': nome}
      alunos.append(novo_aluno)
      return 0
```

Teste.py

```
import unittest
from entidades.aluno import *

def test_01_inserir_aluno_ok_condicao_retorno(self):
      print("Caso de Teste 01 - Inserir com sucesso")
      retorno_esperado = insere_aluno('9014513','Flavio')
      self.assertEqual(retorno_esperado, 0)
```

TDD em Python

Teste.py (cont.)

```
def test_02_inserir_aluno_ok_inserido_com_sucesso(self):
        print("Caso de Teste 02 - Verifica inserção")
        self.assertIn({'matricula':'9014513',
        'nome':'Flavio'},gera_relacao_alunos())

def test_03_inserir_aluno_nok_ja_existente(self):
        print("Caso de Teste 03 - Impede a inserção caso " +
                " já exista a matricula inserida")
        retorno_esperado = insere_aluno('9014513','Flavio')
        self.assertEqual(retorno_esperado, 1 )
```

Aluno.py

```
def gera_relacao_alunos():
        relacao_alunos = []
        for aluno in alunos:
                relacao_alunos.append(aluno)
        return relacao_alunos
```

TDD em Python

Aluno.py

```
def altera_aluno(matricula, campo, valor):
      for aluno in alunos:
      if aluno['matricula'] == matricula:
      aluno[campo] = valor
      return 0
      return 1
```

Teste.py

```
def test_05_alterar_aluno_ok_condicao_retorno(self):
      print("Caso de Teste 05 - Alterar com sucesso")
      insere_aluno('9011122','João')
      insere_aluno('9012233','Ana')
      retorno_esperado = Altera_aluno('9011122','nome',
      'Matheus')
      self.assertEqual(retorno_esperado, 0)

def test_06_alterar_aluno_ok_alterado_com_sucesso(self):
      print("Caso de Teste 06 - Verifica se alterou " +
      " efetivamente")
      self.assertIn({'matricula':'9011122','nome':'Matheus'},
      gera_relacao_alunos())
```

TDD em Python

Aluno.py

```
def exclui_aluno(matricula):
      for aluno in alunos:
            if aluno['matricula'] == matricula:
                  alunos.remove(aluno)
                  return 0
      return 1
```

Teste.py

```
def test_08_excluir_aluno_ok_condicao_retorno(self):
      print("Caso de Teste 08 - Excluir com sucesso")
      retorno_esperado = exclui_aluno('9012233')
      self.assertEqual(retorno_esperado, 0)

def test_09_excluir_aluno_ok_excluido_com_sucesso(self):
      print("Caso de Teste 09 - Verifica se excluiu”)
      self.assertNotIn({'matricula':'9012233',
            'nome':'Ana'}, ‘gera_relacao_alunos())
```

TDD em Python

Aluno.py

```
def gera_relacao_alunos():
        relacao_alunos = []
        for aluno in alunos:
                relacao_alunos.append(aluno)
        return relacao_alunos
```

Teste.py

```
def test_13_gerar_relacao_alunos(self):
        print("Caso de Teste 13 - Gerar relacao de
alunos")
        self.assertEqual([{'matricula':'9014513',
'nome':'Flavio'}, {'matricula':'9011122',
'nome':'Matheus'}], gera_relacao_alunos())
unittest.main()      # ao final para executar os testes
```

TDD em Python

Teste executou com sucesso (resultado obtido = resultado esperado)

```
Caso de Teste 01 - Inserir com sucesso
.Caso de Teste 02 - Verifica inserção
.Caso de Teste 03 - Impede a inserção caso já exista a
matricula inserida
.Caso de Teste 05 - Alterar com sucesso
.Caso de Teste 06 - Verifica se alterou
.Caso de Teste 08 - Excluir com sucesso
.Caso de Teste 09 - Verifica se excluiu
.Caso de Teste 13 - Gerar relacao de alunos
.
----------------------------------------------------------------
Ran 8 tests in 0.065s

OK
```

TDD em Python

Teste executou com erro

Exemplo: no teste trocamos o nome da função para uma inexistente

```
def test_01_inserir_aluno_ok_condicao_retorno(self):
        retorno_esperado = insera_aluno('9014513','Flavio')
        self.assertEqual(retorno_esperado, 0)
```

Segue o relatório de execução do teste

```
Caso de Teste 01 - Condicao de Retorno 0 ao inserir com sucesso
ECaso de Teste 02 - Verifica se inseriu efetivamente
FCaso de Teste 03 - Impede a inserção caso já exista a
matricula inserida
FCaso de Teste 05 - Condição de Retorno 0 ao alterar com
sucesso
. Caso de Teste 06 - Verifica se alterou efetivamente
. Caso de Teste 08 - Condição de Retorno 0 ao excluir com
sucesso
. Caso de Teste 09 - Verifica se excluiu efetivamente
```

TDD em Python

Teste executou com erro

```
======================================================================
ERROR: test_01_inserir_aluno_ok_condicao_retorno (__main__.TestAluno)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "testes.py", line 22, in test_01_inserir_aluno_ok_condicao_retorno
    retorno_esperado = insera_aluno('9014513','Flavio')
NameError: name 'insera_aluno' is not defined

======================================================================
FAIL: test_02_inserir_aluno_ok_inserido_com_sucesso (__main__.TestAluno)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "testes.py", line 27, in
test_02_inserir_aluno_ok_inserido_com_sucesso

self.assertIn({'matricula':'9014513','nome':'Flavio'},gera_relacao_alunos(
))

AssertionError: {'matricula': '9014513', 'nome': 'Flavio'} not found in []
```

TDD em Python

Teste executou com erro

```
======================================================================
FAIL: test_03_inserir_aluno_nok_ja_existente (__main__.TestAluno)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "testes.py", line 32, in test_03_inserir_aluno_nok_ja_existente
    self.assertEqual(retorno_esperado, 1 )
AssertionError: 0 != 1

----------------------------------------------------------------------
Ran 7 tests in 0.058s

FAILED (failures=2, errors=1)

------------------
(program exited with code: 1)
```

TDD em Python

Seguem alguns comandos

- assert: criar assertivas customizadas
- assertEqual(a, b): checa se a e b são iguais
- assertNotEqual(a, b): checa se a e b são diferentes
- assertIn(a, b): check se a faz parte de b
- assertNotIn(a, b): check se a não faz parte de b
- assertFalse(a): checa se o valor de a é falso
- assertTrue(a): check se o valor de a é verdadeiro

<u>Nomenclatura do relatório de testes</u>

E = Erro ao executar o código a ser testado.

F = Falha quando o código a ser testado funciona, porém gera um comportamento diferente do esperado