

Python & TDD

Python

- Orientado a Objetos
- Tipagem Dinamica, porem forte
- Escopo definido pela indentação

Python

```
class NomeDaClasse :  
  
    def NomeDoMetodo(self):  
        var1 = 'string'  
        var2 = "outra string"  
        return var1 + " - " + var2
```

return var1 + " - " + var2

Python

```
class NomeDaClasse :  
  
    def metodoComIf(self, param):  
        var1 = 'string'  
        var2 = 123  
        if var2 > param :  
            return var1  
        else:  
            return var2
```


Python

```
class NomeDaClasse :  
  
    def metodoComWhile(self, qtd):  
        while qtd > 0:  
            qtd -= 1  
            print 'teste com while'
```

print 'teste com while'

Python

```
class NomeDaClasse :  
  
    def metodoComLista(self):  
        lista = [1,2,3,4,5]  
        for i in lista:  
            print i
```


Python

```
class NomeDaClasse :  
  
    def metodoComTupla(self):  
        tupla = {'chave' : 2, 'a' : 3, 'b' : 4}  
        for i in tupla:  
            print "chave : " + i + " | valor : " + str(tupla[i])
```

```
print "chave : " + i + " | valor : " + str(tupla[i])
```

Python

```
class NomeDaClasse :  
    def NomeDoMetodo(self):  
        ...  
    def metodoComIf(self, param):  
        ...  
    def metodoComWhile(self, qtd):  
        ...  
    def metodoComLista(self):  
        ...  
    def metodoComTupla(self):  
        ...
```

```
obj = NomeDaClasse()  
print obj.NomeDoMetodo()  
print obj.metodoComIf(30)  
obj.metodoComWhile(4)  
obj.metodoComLista()  
obj.metodoComTupla()
```


TDD

- Ferramenta para proteger o desenvolvedor dele próprio
- Os Testes são escritos antes do código
- *Feedback* instantâneo
- Serve como Documentação

TDD

```
import unittest

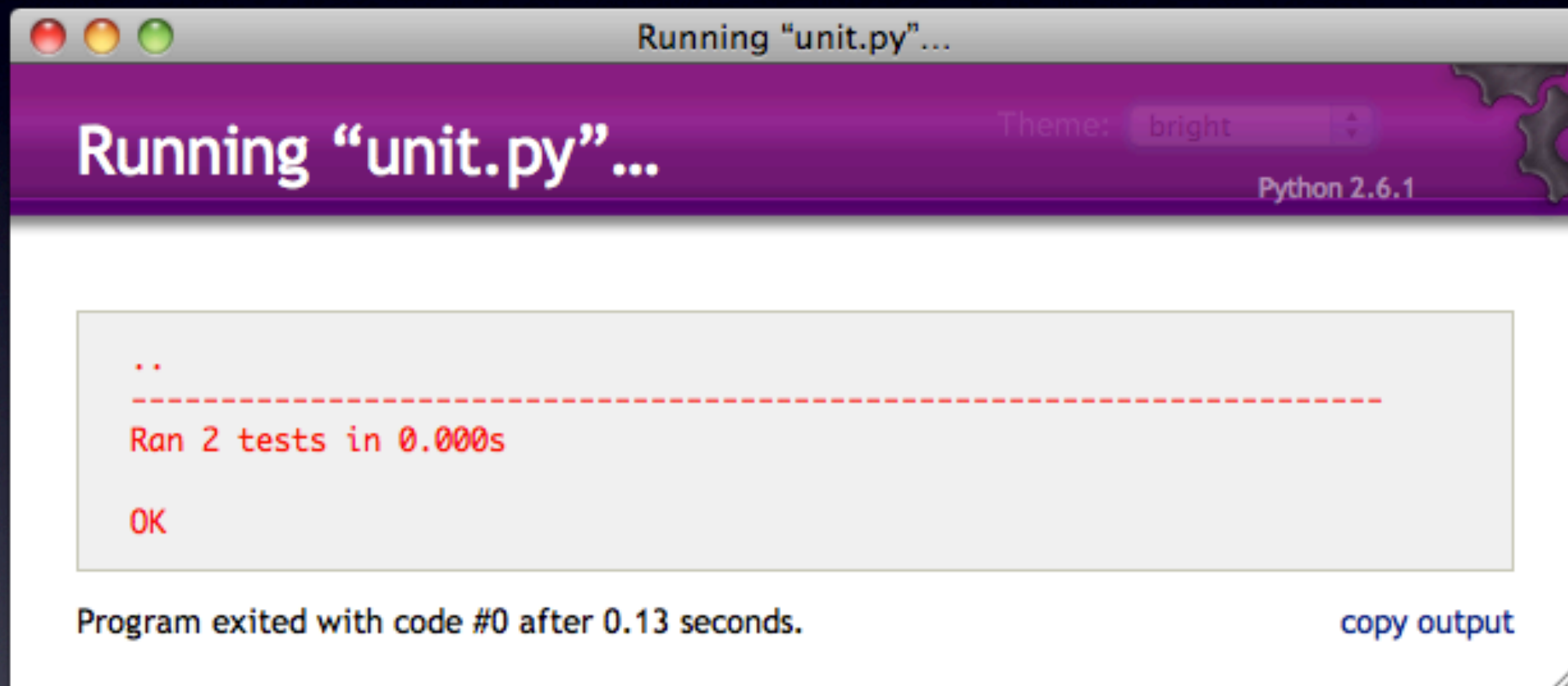
from test import NomeDaClasse

class testa_metodo(unittest.TestCase):
    def test1(self):
        obj = NomeDaClasse()
        self.assertEqual(obj.metodoComIf(100), 'string')

    def test2(self):
        obj = NomeDaClasse()
        self.assertEqual(obj.metodoComIf(150), 123)

unittest.main()
```


TDD



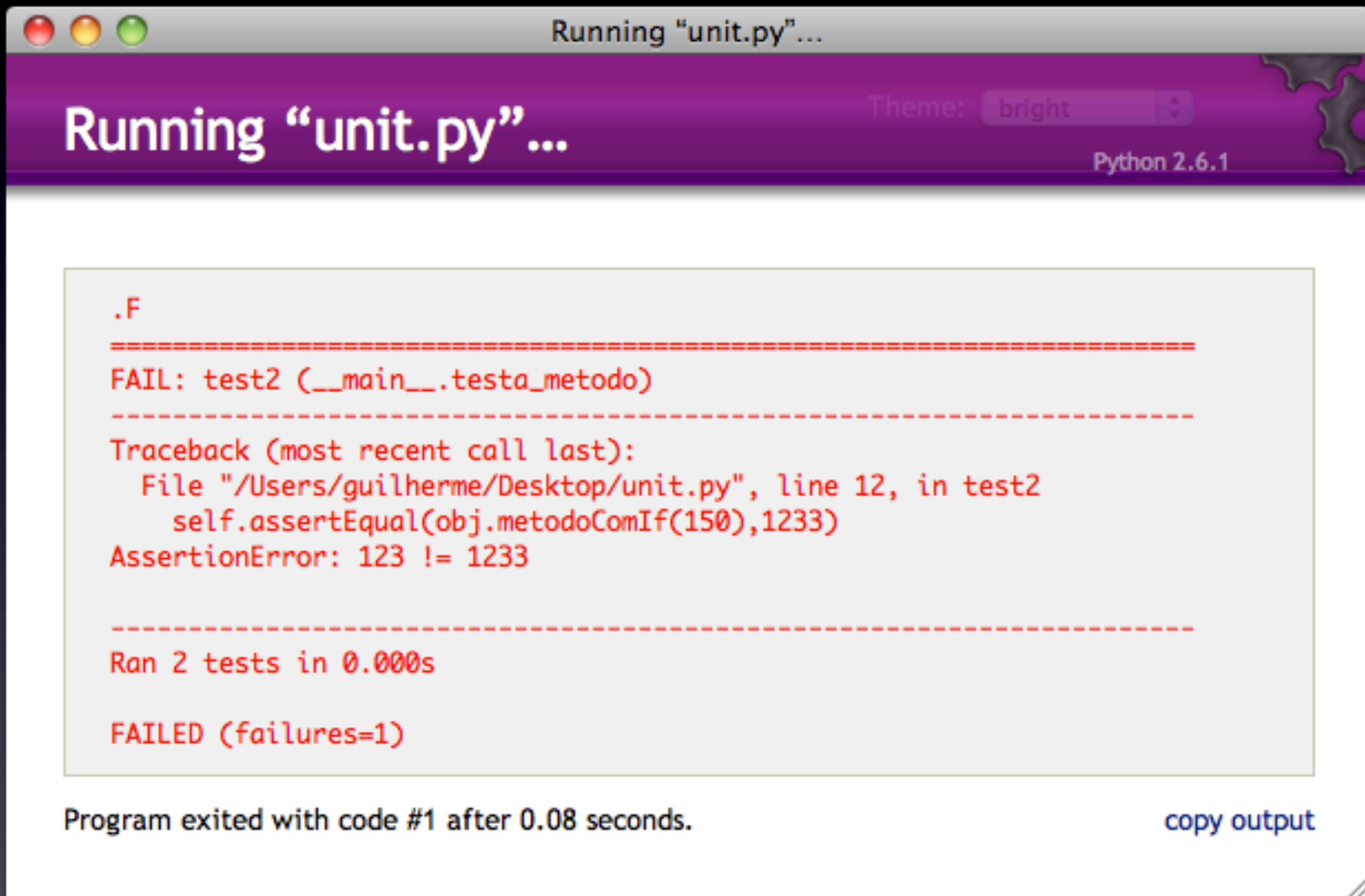
The screenshot shows a terminal window with a title bar that reads "Running 'unit.py'...". The window has a purple header bar with the same text "Running 'unit.py'..." on the left, a "Theme: bright" dropdown menu in the center, and "Python 2.6.1" on the right. The main content area is white and contains the following text in red:
..

Ran 2 tests in 0.000s

OK
Below this text, it says "Program exited with code #0 after 0.13 seconds." and a blue link "copy output" is visible on the right.

```
..  
-----  
Ran 2 tests in 0.000s  
  
OK  
  
Program exited with code #0 after 0.13 seconds.  
copy output
```

TDD



```
Running "unit.py"...
```

Theme: bright Python 2.6.1

```
.F
=====
FAIL: test2 (__main__.testa_metodo)
-----
Traceback (most recent call last):
  File "/Users/guilherme/Desktop/unit.py", line 12, in test2
    self.assertEqual(obj.metodoComIf(150),1233)
AssertionError: 123 != 1233

-----
Ran 2 tests in 0.000s

FAILED (failures=1)
```

Program exited with code #1 after 0.08 seconds. [copy output](#)