

Тестирование в ситуациях с ошибками в doctest

The screenshot shows the PyCharm IDE interface. The top editor pane displays a Python file named `__init__.py` with the following code:

```
1 import unittest
2 class Test(object):
3     """
4     >>> a=Test(5)
5     >>> a.multiply_by_2()
6     54
7     """
8
9     def __init__(self, cabinet):
10         self._cabinet = cabinet
11
12     def multiply_by_2(self):
13         return self._cabinet * 2
14
15 if __name__ == "__main__":
16     import doctest
17     doctest.testmod()
18
19 class TestUM(unittest.TestCase):
20     def setUp(self):
21         pass
22     def tearDown(self):
```

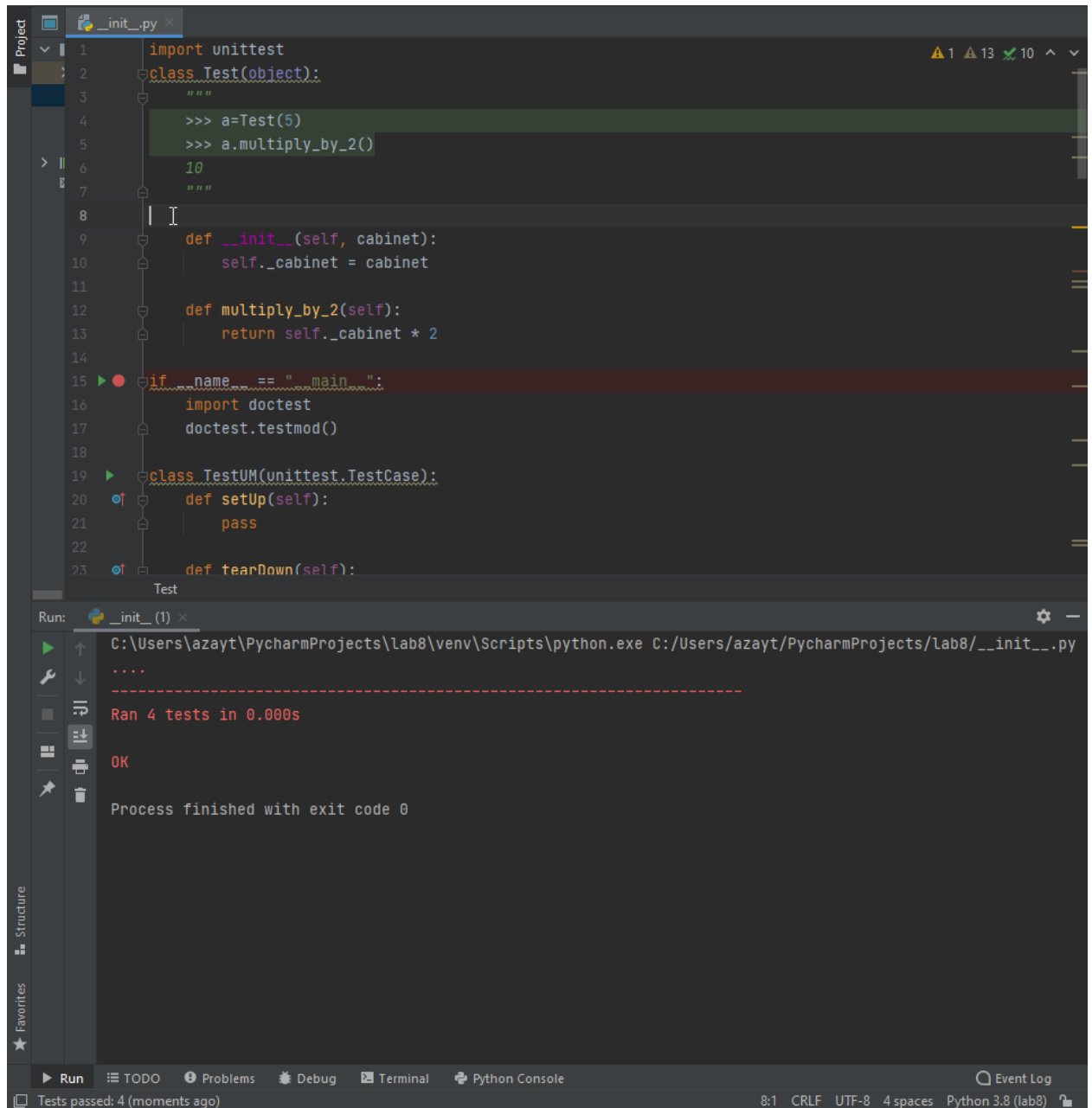
The bottom pane shows the output of the test run. It indicates that 4 tests were run successfully in 0.000s. However, there is a failure in the doctest. The output shows the failed example and the expected vs. got values:

```
OK
*****
File "C:/Users/azayt/PycharmProjects/lab8/__init__.py", line 5, in __main__.Test
Failed example:
  a.multiply_by_2()
Expected:
  54
Got:
  10
*****
1 items had failures:
  1 of  2 in __main__.Test
***Test Failed*** 1 failures.

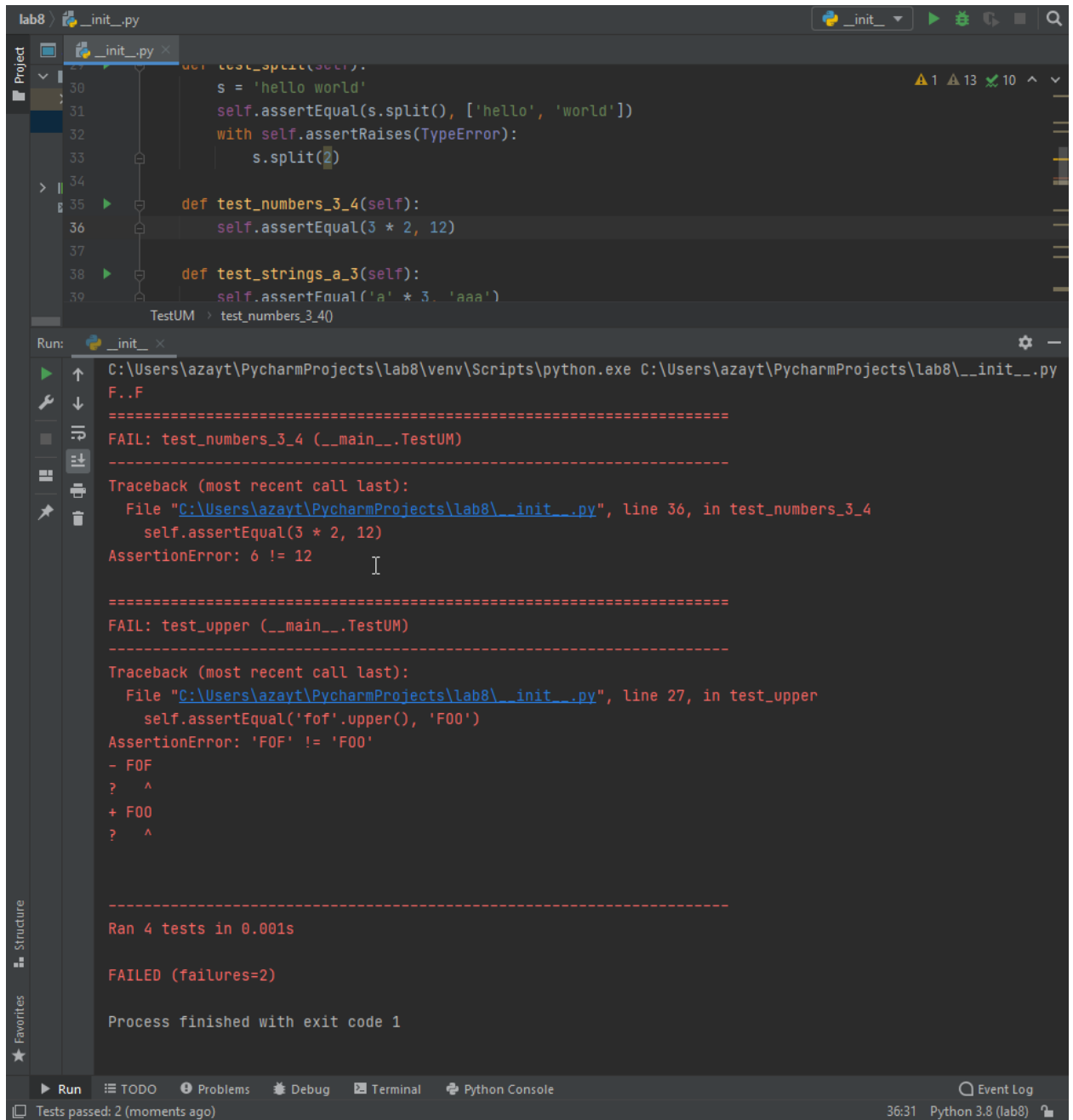
Process finished with exit code 0
```

The status bar at the bottom indicates that 4 tests passed and 1 failed. The Python version is 3.8.

Работа doctest без ошибок



Тестирование в ситуациях с ошибками unittest



The screenshot displays the PyCharm IDE interface. The top editor shows a Python file `__init__.py` with the following code:

```
27 def test_split(self):
30     s = 'hello world'
31     self.assertEqual(s.split(), ['hello', 'world'])
32     with self.assertRaises(TypeError):
33         s.split(2)
34
35 def test_numbers_3_4(self):
36     self.assertEqual(3 * 2, 12)
37
38 def test_strings_a_3(self):
39     self.assertEqual('a' * 3, 'aaa')
```

The bottom panel shows the output of the test run. The command executed is `C:\Users\azayt\PycharmProjects\lab8\venv\Scripts\python.exe C:\Users\azayt\PycharmProjects\lab8__init__.py`. The output indicates that 4 tests were run, but 2 failed:

```
F..F
=====
FAIL: test_numbers_3_4 (__main__.TestUM)
=====
Traceback (most recent call last):
  File "C:\Users\azayt\PycharmProjects\lab8\__init__.py", line 36, in test_numbers_3_4
    self.assertEqual(3 * 2, 12)
AssertionError: 6 != 12

=====
FAIL: test_upper (__main__.TestUM)
=====
Traceback (most recent call last):
  File "C:\Users\azayt\PycharmProjects\lab8\__init__.py", line 27, in test_upper
    self.assertEqual('fof'.upper(), 'F00')
AssertionError: 'FOF' != 'F00'
- FOF
?  ^
+ F00
?  ^

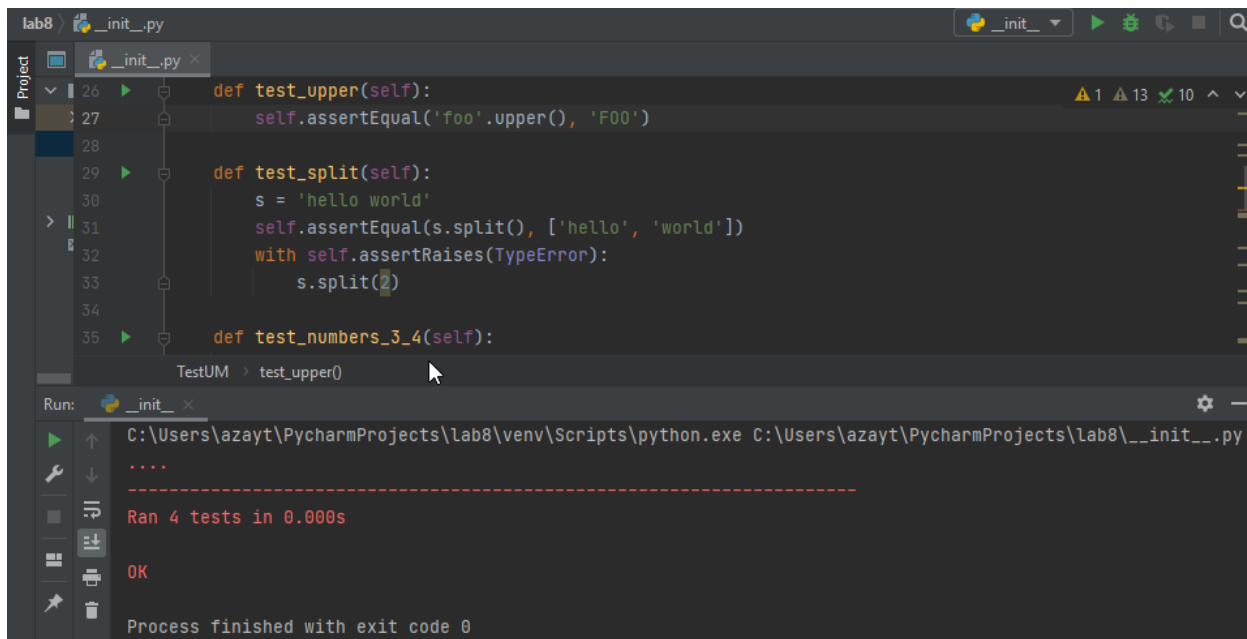
-----
Ran 4 tests in 0.001s

FAILED (failures=2)

Process finished with exit code 1
```

The status bar at the bottom shows "Tests passed: 2 (moments ago)" and "36:31 Python 3.8 (lab8)".

Работа без ошибок



The screenshot displays the PyCharm IDE interface. The top pane shows the source code of a Python file named `__init__.py` within a project named `lab8`. The code defines three test methods: `test_upper`, `test_split`, and `test_numbers_3_4`. The `test_split` method includes an assertion for string splitting and a context manager to verify that a `TypeError` is raised when `split(2)` is called. The bottom pane shows the output of running these tests, indicating that all four tests passed successfully in 0.000 seconds.

```
lab8 > __init__.py
def test_upper(self):
    self.assertEqual('foo'.upper(), 'F00')

def test_split(self):
    s = 'hello world'
    self.assertEqual(s.split(), ['hello', 'world'])
    with self.assertRaises(TypeError):
        s.split(2)

def test_numbers_3_4(self):
```

TestUM > test_upper()

Run: C:\Users\azayt\PycharmProjects\lab8\venv\Scripts\python.exe C:\Users\azayt\PycharmProjects\lab8__init__.py

....

Ran 4 tests in 0.000s

OK

Process finished with exit code 0