

Christian Szablewski-Paz

SSW 567

Dr. Saremi

11 September 2021

Deliverable 2: Intended test cases failed from function `classify_triangle_with_errors()` and all test cases passed from function `classify_triangle()`

```
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
You should consider upgrading via the '/Users/christian/Documents/GitHub/ssw567HW1/pytest-env/bin/python3 -m pip install --upgrade pip' command.
(pytest-env) christian@Christians-MacBook-Pro ssw567HW1 % pytest main.py
===== test session starts =====
platform darwin -- Python 3.9.7, pytest-6.2.5, py-1.10.0, pluggy-1.0.0
rootdir: /Users/christian/Documents/GitHub/ssw567HW1
collected 0 items

===== no tests ran in 0.00s =====
ERROR: file or directory not found: main.py

(pytest-env) christian@Christians-MacBook-Pro ssw567HW1 % cd pytest-env
(pytest-env) christian@Christians-MacBook-Pro pytest-env % pytest main.py
===== test session starts =====
platform darwin -- Python 3.9.7, pytest-6.2.5, py-1.10.0, pluggy-1.0.0
rootdir: /Users/christian/Documents/GitHub/ssw567HW1/pytest-env
collected 16 items

main.py .....FFFF [100%]

===== FAILURES =====
_____ test_classify_triangle_1_with_errors _____
>
def test_classify_triangle_1_with_errors():
    assert classify_triangle_with_errors(1, 1) == "equilateral"
E       AssertionError: assert 'this will error' == 'equilateral'
E       - equilateral
E       + this will error
main.py:58: AssertionError
_____ test_classify_triangle_4_with_errors _____
>
def test_classify_triangle_4_with_errors():
    assert classify_triangle_with_errors(3, 2, 3) == "isosceles"
E       AssertionError: assert 'this will error' == 'isosceles'
E       - isosceles
E       + this will error
main.py:61: AssertionError
_____ test_classify_triangle_7_with_errors _____
>
def test_classify_triangle_7_with_errors():
    assert classify_triangle_with_errors(3, 4, 5) == "right"
E       AssertionError: assert 'this will error' == 'right'
E       - right
E       + this will error
```

```
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
main.py .....FFFF [100%]

===== FAILURES =====
_____ test_classify_triangle_1_with_errors _____
>
def test_classify_triangle_1_with_errors():
    assert classify_triangle_with_errors(1, 1) == "equilateral"
E       AssertionError: assert 'this will error' == 'equilateral'
E       - equilateral
E       + this will error
main.py:58: AssertionError
_____ test_classify_triangle_4_with_errors _____
>
def test_classify_triangle_4_with_errors():
    assert classify_triangle_with_errors(3, 2, 3) == "isosceles"
E       AssertionError: assert 'this will error' == 'isosceles'
E       - isosceles
E       + this will error
main.py:61: AssertionError
_____ test_classify_triangle_7_with_errors _____
>
def test_classify_triangle_7_with_errors():
    assert classify_triangle_with_errors(3, 4, 5) == "right"
E       AssertionError: assert 'this will error' == 'right'
E       - right
E       + this will error
main.py:64: AssertionError
_____ test_classify_triangle_12_with_errors _____
>
def test_classify_triangle_12_with_errors():
    assert classify_triangle_with_errors(5, 4, 3) == "scalene"
E       AssertionError: assert 'this will error' == 'scalene'
E       - scalene
E       + this will error
main.py:67: AssertionError

===== short test summary info =====
FAILED main.py::test_classify_triangle_1_with_errors - AssertionError: assert 'this will error' == 'equila...
FAILED main.py::test_classify_triangle_4_with_errors - AssertionError: assert 'this will error' == 'isocse...
FAILED main.py::test_classify_triangle_7_with_errors - AssertionError: assert 'this will error' == 'right'
FAILED main.py::test_classify_triangle_12_with_errors - AssertionError: assert 'this will error' == 'scalene'
4 failed, 12 passed in 0.00s
```

### Deliverable 3:

- The only real challenges I faced during this was figuring out the pytest library terminal commands and its “assert()” function syntax. The coding was relatively straight forward, I spent most of my time mainly just finding what I needed in documentation.
- The requirements specification for this assignment were easy to follow and gave a lot of flexibility in terms of the coding.
- I did not encounter any challenges with the tools besides accidentally running “python3 main.py” instead of “pytest main.py” at first.
- I tested my function by testing for each expected output at least 3 times with different inputs. At a high level, this means I tested all requirements and use cases at least once, as well as ensured each line of code was run at least 3 times. I also made a function that intentionally gave wrong output, just for the sake of testing pyest since it was my first time using it. I was able to see that all of my pytest unit tests failed as expected for that function.

Deliverable 4: <https://github.com/cs-paz/ssw567HW1/tree/main/pytest-env>