HW05 - Static Code Analysis

Tarik Kdiry - September 30, 2018

1) Assignment Description:

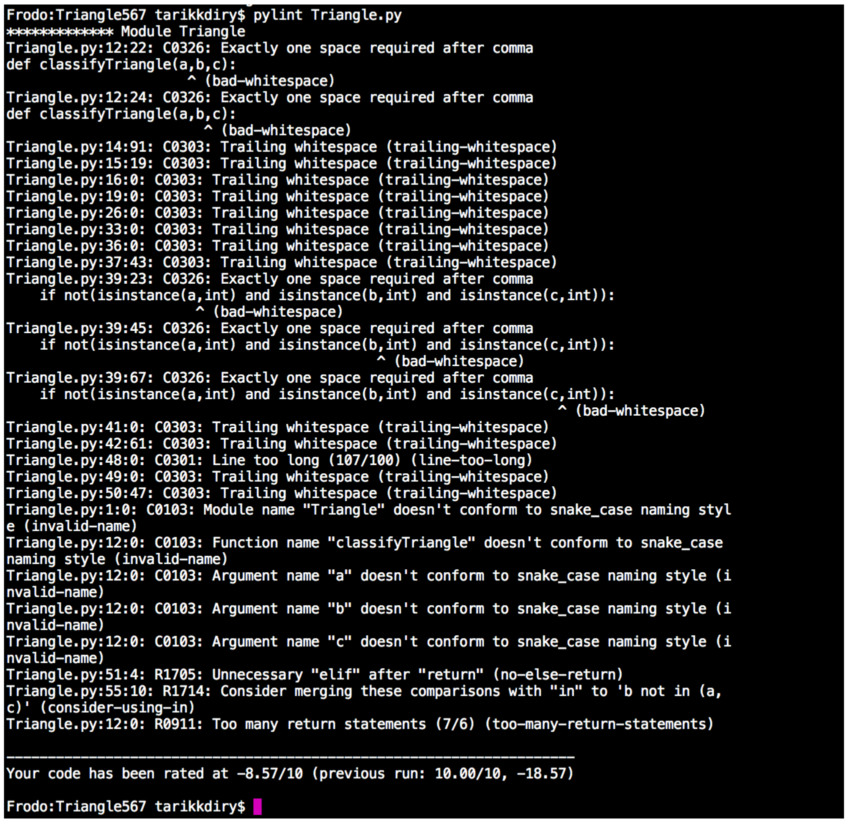
In this assignment, you will need to download and install the tools that you will need for static code analysis and code coverage. You will then run those tools locally on your laptop to get the results.

2) Github URL: https://github.com/tarikkdiry/Triangle567

3) Summary:

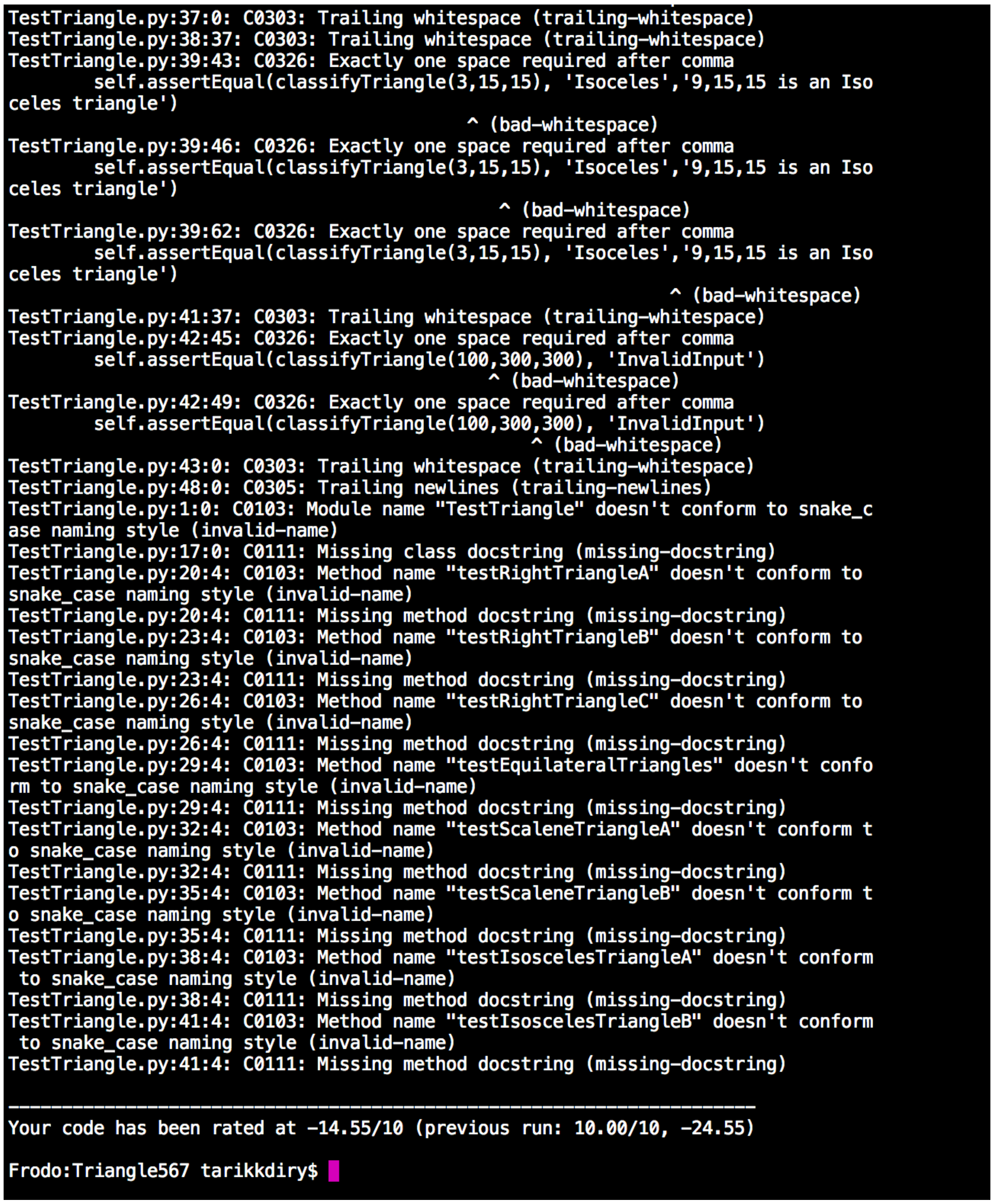
Before changes/First PyLint iteration:

Triangle.py



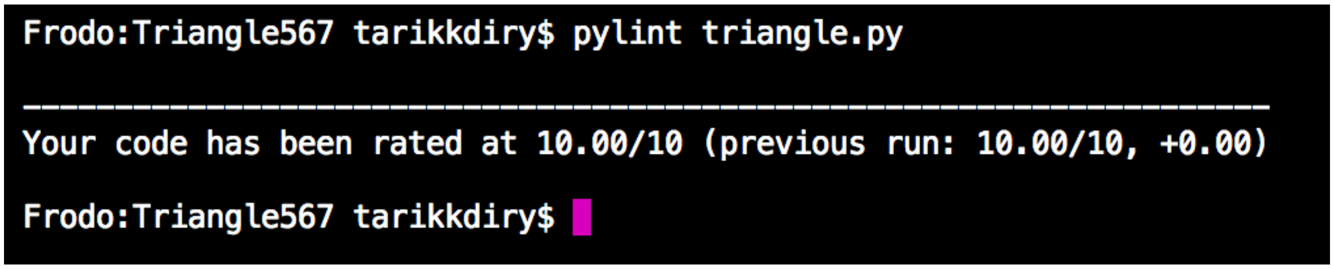
```
Frodo:Triangle567 tarikkdiry$ pylint Triangle.py
************* Module Triangle
Triangle.py:12:22: C0326: Exactly one space required after comma
def classifyTriangle(a,b,c):
                      ^ (bad-whitespace)
Triangle.py:12:24: C0326: Exactly one space required after comma
def classifyTriangle(a,b,c):
                        ^ (bad-whitespace)
Triangle.py:14:91: C0303: Trailing whitespace (trailing-whitespace)
Triangle.py:15:19: C0303: Trailing whitespace (trailing-whitespace)
Triangle.py:16:0: C0303: Trailing whitespace (trailing-whitespace)
Triangle.py:19:0: C0303: Trailing whitespace (trailing-whitespace)
Triangle.py:26:0: C0303: Trailing whitespace (trailing-whitespace)
Triangle.py:33:0: C0303: Trailing whitespace (trailing-whitespace)
Triangle.py:36:0: C0303: Trailing whitespace (trailing-whitespace)
Triangle.py:37:43: C0303: Trailing whitespace (trailing-whitespace)
Triangle.py:39:23: C0326: Exactly one space required after comma
    if not(isinstance(a,int) and isinstance(b,int) and isinstance(c,int)):
                       ^ (bad-whitespace)
Triangle.py:39:45: C0326: Exactly one space required after comma
    if not(isinstance(a,int) and isinstance(b,int) and isinstance(c,int)):
                                             ^ (bad-whitespace)
Triangle.py:39:67: C0326: Exactly one space required after comma
    if not(isinstance(a,int) and isinstance(b,int) and isinstance(c,int)):
                                                                   ^ (bad-whitespace)
Triangle.py:41:0: C0303: Trailing whitespace (trailing-whitespace)
Triangle.py:42:61: C0303: Trailing whitespace (trailing-whitespace)
Triangle.py:48:0: C0301: Line too long (107/100) (line-too-long)
Triangle.py:49:0: C0303: Trailing whitespace (trailing-whitespace)
Triangle.py:50:47: C0303: Trailing whitespace (trailing-whitespace)
Triangle.py:1:0: C0103: Module name "Triangle" doesn't conform to snake_case naming styl
e (invalid-name)
Triangle.py:12:0: C0103: Function name "classifyTriangle" doesn't conform to snake_case
naming style (invalid-name)
Triangle.py:12:0: C0103: Argument name "a" doesn't conform to snake_case naming style (i
nvalid-name)
Triangle.py:12:0: C0103: Argument name "b" doesn't conform to snake_case naming style (i
nvalid-name)
Triangle.py:12:0: C0103: Argument name "c" doesn't conform to snake_case naming style (i
nvalid-name)
Triangle.py:51:4: R1705: Unnecessary "elif" after "return" (no-else-return)
Triangle.py:55:10: R1714: Consider merging these comparisons with "in" to 'b not in (a,
c)' (consider-using-in)
Triangle.py:12:0: R0911: Too many return statements (7/6) (too-many-return-statements)

----------------------------------------------------------------------
Your code has been rated at -8.57/10 (previous run: 10.00/10, -18.57)

Frodo:Triangle567 tarikkdiry$
```

TestTriangle.py



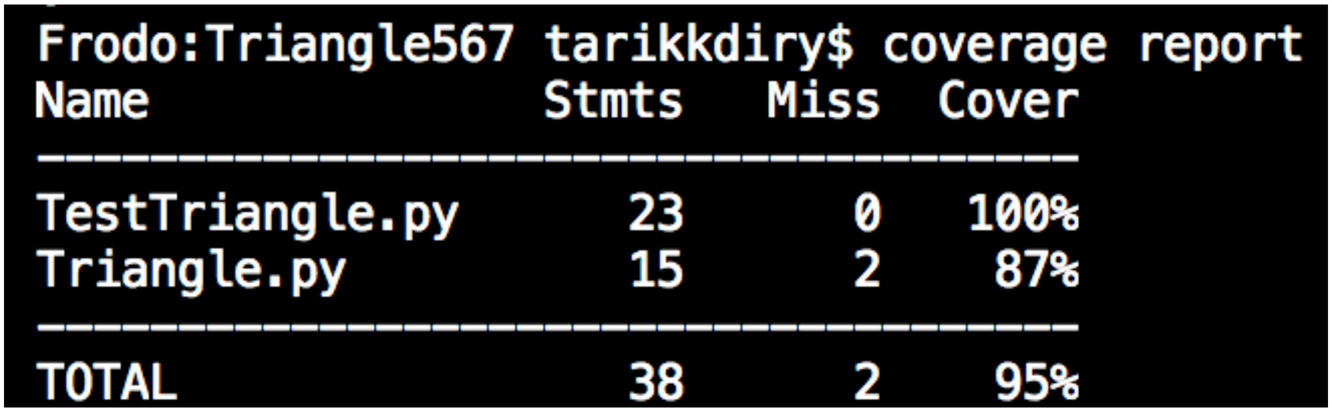
```
TestTriangle.py:37:0: C0303: Trailing whitespace (trailing-whitespace)
TestTriangle.py:38:37: C0303: Trailing whitespace (trailing-whitespace)
TestTriangle.py:39:43: C0326: Exactly one space required after comma
        self.assertEqual(classifyTriangle(3,15,15), 'Isoceles','9,15,15 is an Iso
celes triangle')
                                            ^ (bad-whitespace)
TestTriangle.py:39:46: C0326: Exactly one space required after comma
        self.assertEqual(classifyTriangle(3,15,15), 'Isoceles','9,15,15 is an Iso
celes triangle')
                                               ^ (bad-whitespace)
TestTriangle.py:39:62: C0326: Exactly one space required after comma
        self.assertEqual(classifyTriangle(3,15,15), 'Isoceles','9,15,15 is an Iso
celes triangle')
                                                               ^ (bad-whitespace)
TestTriangle.py:41:37: C0303: Trailing whitespace (trailing-whitespace)
TestTriangle.py:42:45: C0326: Exactly one space required after comma
        self.assertEqual(classifyTriangle(100,300,300), 'InvalidInput')
                                              ^ (bad-whitespace)
TestTriangle.py:42:49: C0326: Exactly one space required after comma
        self.assertEqual(classifyTriangle(100,300,300), 'InvalidInput')
                                                  ^ (bad-whitespace)
TestTriangle.py:43:0: C0303: Trailing whitespace (trailing-whitespace)
TestTriangle.py:48:0: C0305: Trailing newlines (trailing-newlines)
TestTriangle.py:1:0: C0103: Module name "TestTriangle" doesn't conform to snake_c
ase naming style (invalid-name)
TestTriangle.py:17:0: C0111: Missing class docstring (missing-docstring)
TestTriangle.py:20:4: C0103: Method name "testRightTriangleA" doesn't conform to
snake_case naming style (invalid-name)
TestTriangle.py:20:4: C0111: Missing method docstring (missing-docstring)
TestTriangle.py:23:4: C0103: Method name "testRightTriangleB" doesn't conform to
snake_case naming style (invalid-name)
TestTriangle.py:23:4: C0111: Missing method docstring (missing-docstring)
TestTriangle.py:26:4: C0103: Method name "testRightTriangleC" doesn't conform to
snake_case naming style (invalid-name)
TestTriangle.py:26:4: C0111: Missing method docstring (missing-docstring)
TestTriangle.py:29:4: C0103: Method name "testEquilateralTriangles" doesn't confo
rm to snake_case naming style (invalid-name)
TestTriangle.py:29:4: C0111: Missing method docstring (missing-docstring)
TestTriangle.py:32:4: C0103: Method name "testScaleneTriangleA" doesn't conform t
o snake_case naming style (invalid-name)
TestTriangle.py:32:4: C0111: Missing method docstring (missing-docstring)
TestTriangle.py:35:4: C0103: Method name "testScaleneTriangleB" doesn't conform t
o snake_case naming style (invalid-name)
TestTriangle.py:35:4: C0111: Missing method docstring (missing-docstring)
TestTriangle.py:38:4: C0103: Method name "testIsoscelesTriangleA" doesn't conform
 to snake_case naming style (invalid-name)
TestTriangle.py:38:4: C0111: Missing method docstring (missing-docstring)
TestTriangle.py:41:4: C0103: Method name "testIsoscelesTriangleB" doesn't conform
 to snake_case naming style (invalid-name)
TestTriangle.py:41:4: C0111: Missing method docstring (missing-docstring)

----------------------------------------------------------------------
Your code has been rated at -14.55/10 (previous run: 10.00/10, -24.55)

Frodo:Triangle567 tarikkdiry$
```

After corrections:



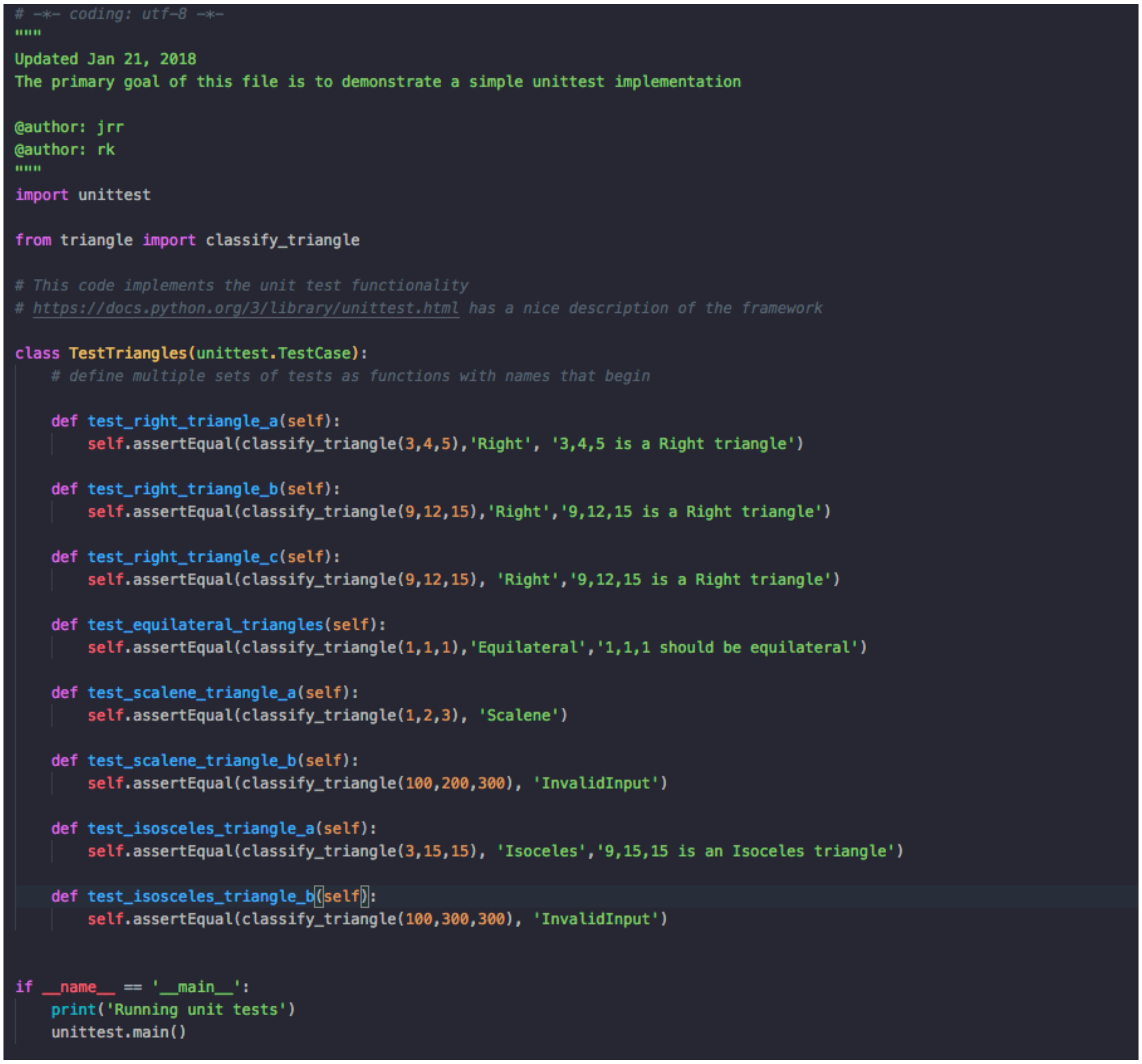
```
Frodo:Triangle567 tarikkdiry$ pylint triangle.py

------------------------------------------------------------------
Your code has been rated at 10.00/10 (previous run: 10.00/10, +0.00)

Frodo:Triangle567 tarikkdiry$ 
```

Coverage.py (Test coverage was already at a higher percentage than 80%)



```
Frodo:Triangle567 tarikkdiry$ coverage report
Name                 Stmts   Miss  Cover
----------------------------------------
TestTriangle.py         23      0   100%
Triangle.py             15      2    87%
----------------------------------------
TOTAL                   38      2    95%
```

Tests:



```python
# -*- coding: utf-8 -*-
"""
Updated Jan 21, 2018
The primary goal of this file is to demonstrate a simple unittest implementation

@author: jrr
@author: rk
"""
import unittest

from triangle import classify_triangle

# This code implements the unit test functionality
# https://docs.python.org/3/library/unittest.html has a nice description of the framework

class TestTriangles(unittest.TestCase):
    # define multiple sets of tests as functions with names that begin

    def test_right_triangle_a(self):
        self.assertEqual(classify_triangle(3,4,5),'Right', '3,4,5 is a Right triangle')

    def test_right_triangle_b(self):
        self.assertEqual(classify_triangle(9,12,15),'Right','9,12,15 is a Right triangle')

    def test_right_triangle_c(self):
        self.assertEqual(classify_triangle(9,12,15), 'Right','9,12,15 is a Right triangle')

    def test_equilateral_triangles(self):
        self.assertEqual(classify_triangle(1,1,1),'Equilateral','1,1,1 should be equilateral')

    def test_scalene_triangle_a(self):
        self.assertEqual(classify_triangle(1,2,3), 'Scalene')

    def test_scalene_triangle_b(self):
        self.assertEqual(classify_triangle(100,200,300), 'InvalidInput')

    def test_isosceles_triangle_a(self):
        self.assertEqual(classify_triangle(3,15,15), 'Isoceles','9,15,15 is an Isoceles triangle')

    def test_isosceles_triangle_b(self):
        self.assertEqual(classify_triangle(100,300,300), 'InvalidInput')


if __name__ == '__main__':
    print('Running unit tests')
    unittest.main()
```

4) Reflection

Throughout this entire process, I have learned how important it is to scrutinize every line and every function name to watch for typos, trailing white-spaces, trailing lines etc. Along with that, I have learned how to apply PyLint and Coverage.py to monitor and ensure the quality of my code.

5) *I pledge my honor that I have abided by the Stevens Honor System.*

-Tarik Kdiry