CSE 4283/6283 - Software Testing & Quality Assurance

Assignment 4

Test-Driven Development and Unit Testing

Name	GitHub Account	NetID
Joshua Church	JoshuaQChurch	jqc10
Jeffry Herzog	SkylineHorizon	jjh258
Evan Farry	EvanFarry	emf134
Johnny Pongetti	Jpongetti	jpp192
Jon Williams	JDW751	jdw751
Corey Henry	cah835	cah835

GitHub Repository:

https://github.com/JoshuaQChurch/SoftwareTesting

Table of Contents

Overview 3

Lessons Learned 3

Test-Drive Development (TDD) Screenshots 4

Setup Instructions 15

Python Unit Test Framework Report 16

Control Flow Graph 18

Conclusions and Final Thoughts 19

Overview

Test-Driven Development is a software development process in which a variety of unit tests are repeatedly tested on source code. TDD is a cycle that repeats until acceptable standards have been reached. The TDD cycle: (1) Add a test, (2) Run all tests and see if the new test fails, (3) Write the code that causes the tests to pass, (4) Run all the new tests and verify that they pass, (5) Refactor / clean up the code, (6) Repeat the cycle to push forward the functionality. Within this paper, we will be discussing what lessons we learned during our experience with TDD, discussing the various failing and passing tests that we developed, reviewing our testing framework, and developing a control flow graph to show the flow of all possibilities within our BMI script.

Lessons Learned

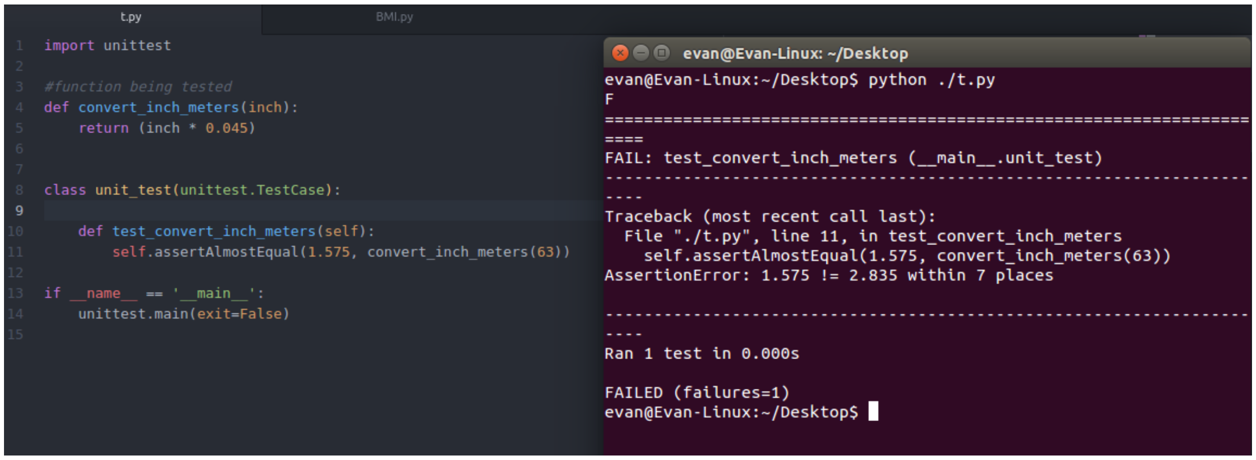
Throughout our experience with Test-Driven Development, we found many benefits using this method; however, we also encountered numerous drawbacks. Setting up our files and overall structure became an issue, along with coordinating the system as a whole. Once this was finalized, we realized the stability of this structure and found the state to be very manageable and organized. After the tests were developed, code implementation became trivial due to the step-by-step approach of TDD. As we proceeded to additional test cases, exception-and-error-handling checks previously implemented made the process of creating the newer test cases easier.

The application of TDD seems to depend on the size of the project. With smaller-scaled projects, this seems to cause a lot of wasted time and effort; however, larger-scaled projects could benefit from this process. Initially, we believed that skipping the unit tests and just developing code would be more beneficial. However, as more test cases were developed, it became apparent that we simply overlooked miniscule mistakes within our code that would have led to faults. These tests allowed us to quickly refactor code for the necessary changes. While implementing the TDD process, we were able to ensure 100% coverage of our project.

Test-Driven Development (TDD) Screenshots

This section aims provide visualization and understanding of how unit tests are developed and used within the coding process. The layout of the images is in the following order: The image on **top** will display a **failed** unit test, while the image on the **bottom** will display a **passed** unit test.

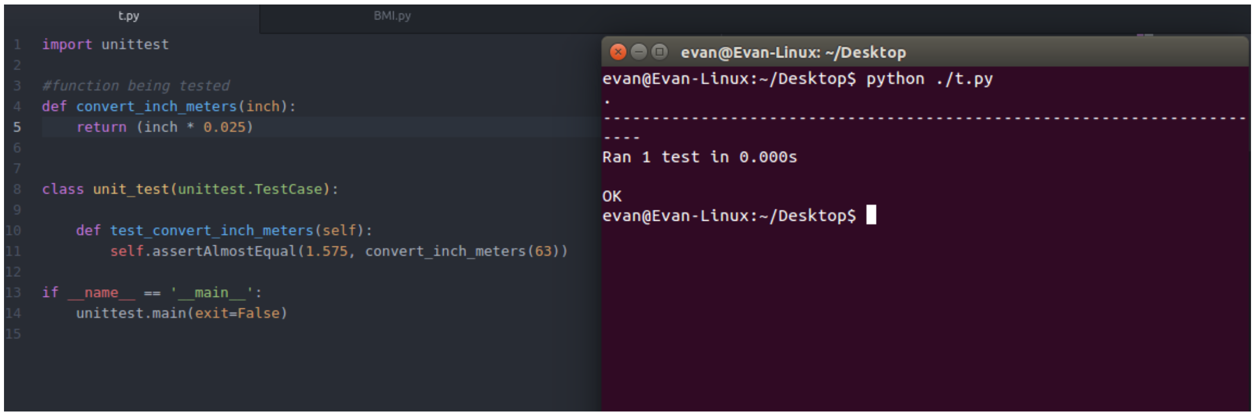
***Figure 1a**: This image shows a **failed** conversion from inches to meters during the **BMI** section*



```python
import unittest

#function being tested
def convert_inch_meters(inch):
    return (inch * 0.045)


class unit_test(unittest.TestCase):

    def test_convert_inch_meters(self):
        self.assertAlmostEqual(1.575, convert_inch_meters(63))

if __name__ == '__main__':
    unittest.main(exit=False)
```

```
evan@Evan-Linux:~/Desktop$ python ./t.py
F
==================================================================
====
FAIL: test_convert_inch_meters (__main__.unit_test)
------------------------------------------------------------------
----
Traceback (most recent call last):
  File "./t.py", line 11, in test_convert_inch_meters
    self.assertAlmostEqual(1.575, convert_inch_meters(63))
AssertionError: 1.575 != 2.835 within 7 places

------------------------------------------------------------------
----
Ran 1 test in 0.000s

FAILED (failures=1)
evan@Evan-Linux:~/Desktop$
```

***Figure 1b:** This image shows a **passed** conversion from inches to meters during the **BMI** section*

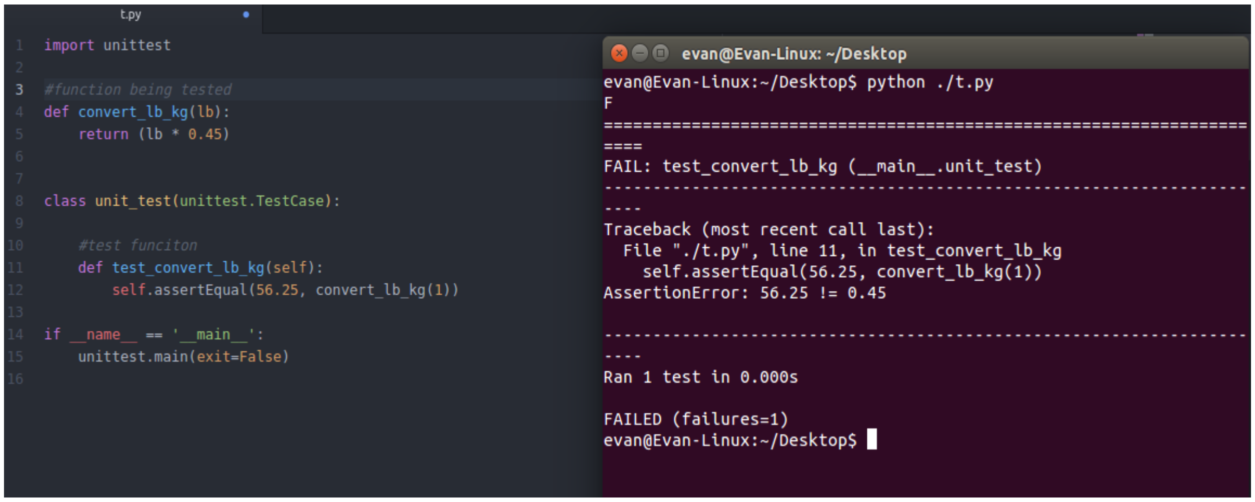


```python
import unittest

#function being tested
def convert_inch_meters(inch):
    return (inch * 0.025)


class unit_test(unittest.TestCase):

    def test_convert_inch_meters(self):
        self.assertAlmostEqual(1.575, convert_inch_meters(63))

if __name__ == '__main__':
    unittest.main(exit=False)
```

```
evan@Evan-Linux:~/Desktop$ python ./t.py
.
------------------------------------------------------------------
----
Ran 1 test in 0.000s

OK
evan@Evan-Linux:~/Desktop$
```

Test-Driven Development (TDD) Screenshots Continued…

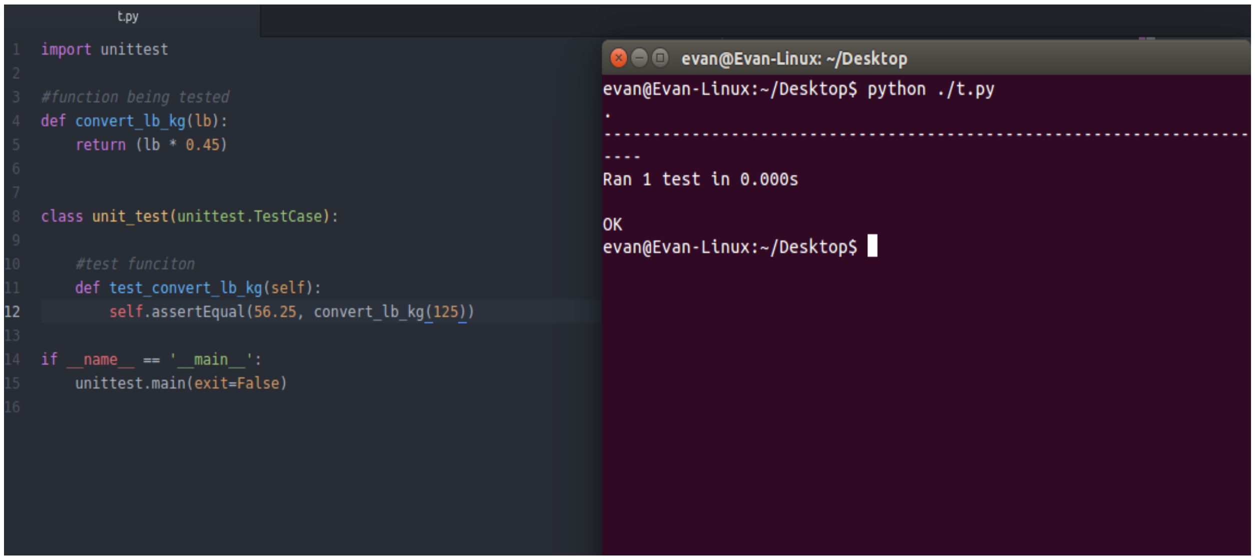
Figure 2a: *This image shows a **failed** conversion from pounds to kilograms during the **BMI** section*



```python
import unittest

#function being tested
def convert_lb_kg(lb):
    return (lb * 0.45)


class unit_test(unittest.TestCase):

    #test funciton
    def test_convert_lb_kg(self):
        self.assertEqual(56.25, convert_lb_kg(1))

if __name__ == '__main__':
    unittest.main(exit=False)
```

```
evan@Evan-Linux:~/Desktop$ python ./t.py
F
==================================================================
====
FAIL: test_convert_lb_kg (__main__.unit_test)
------------------------------------------------------------------
----
Traceback (most recent call last):
  File "./t.py", line 11, in test_convert_lb_kg
    self.assertEqual(56.25, convert_lb_kg(1))
AssertionError: 56.25 != 0.45

------------------------------------------------------------------
----
Ran 1 test in 0.000s

FAILED (failures=1)
evan@Evan-Linux:~/Desktop$
```

Figure 2b: *This image shows a **passed** conversion from pounds to kilograms during the **BMI** section*

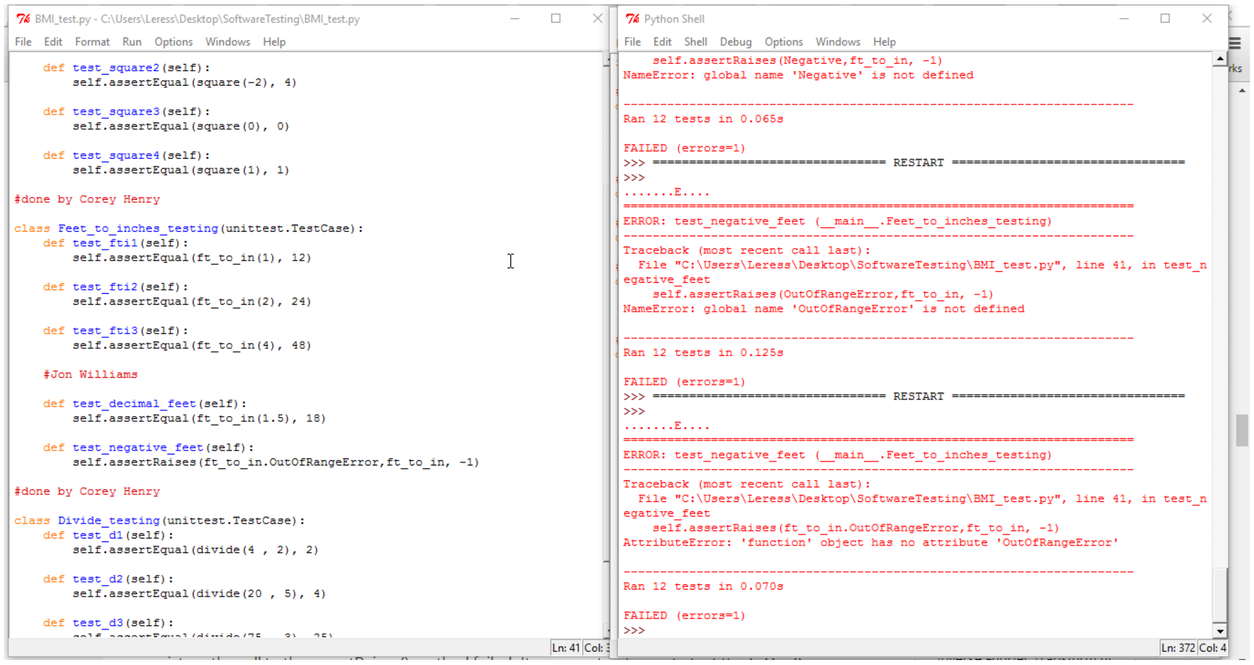


```python
import unittest

#function being tested
def convert_lb_kg(lb):
    return (lb * 0.45)


class unit_test(unittest.TestCase):

    #test funciton
    def test_convert_lb_kg(self):
        self.assertEqual(56.25, convert_lb_kg(125))

if __name__ == '__main__':
    unittest.main(exit=False)
```

```
evan@Evan-Linux:~/Desktop$ python ./t.py
.
------------------------------------------------------------------
----
Ran 1 test in 0.000s

OK
evan@Evan-Linux:~/Desktop$
```

Test-Driven Development (TDD) Screenshots Continued…

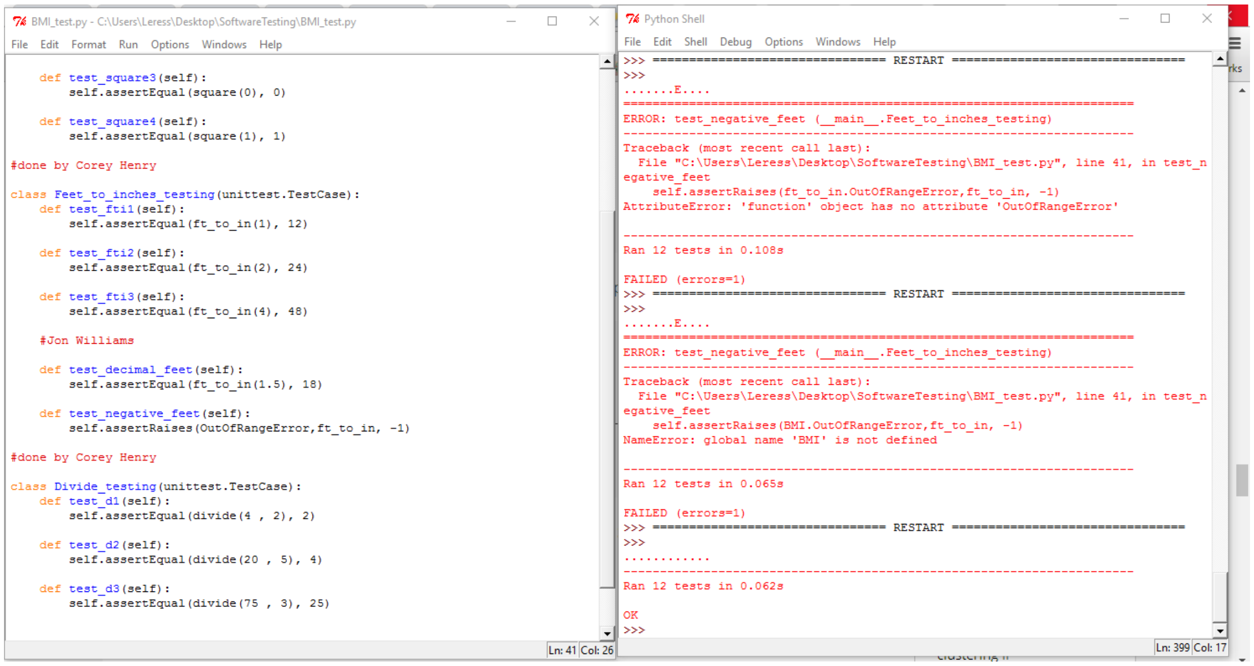
Figure 3a: *This image shows a* ***failed*** *negative input test during the* ***BMI*** *section*



```
    def test_square2(self):
        self.assertEqual(square(-2), 4)

    def test_square3(self):
        self.assertEqual(square(0), 0)

    def test_square4(self):
        self.assertEqual(square(1), 1)

#done by Corey Henry

class Feet_to_inches_testing(unittest.TestCase):
    def test_fti1(self):
        self.assertEqual(ft_to_in(1), 12)

    def test_fti2(self):
        self.assertEqual(ft_to_in(2), 24)

    def test_fti3(self):
        self.assertEqual(ft_to_in(4), 48)

    #Jon Williams

    def test_decimal_feet(self):
        self.assertEqual(ft_to_in(1.5), 18)

    def test_negative_feet(self):
        self.assertRaises(ft_to_in.OutOfRangeError,ft_to_in, -1)

#done by Corey Henry

class Divide_testing(unittest.TestCase):
    def test_d1(self):
        self.assertEqual(divide(4 , 2), 2)

    def test_d2(self):
        self.assertEqual(divide(20 , 5), 4)

    def test_d3(self):
```

```
    self.assertRaises(Negative,ft_to_in, -1)
NameError: global name 'Negative' is not defined

----------------------------------------------------------------------
Ran 12 tests in 0.065s

FAILED (errors=1)
>>> ================================ RESTART ================================
>>> 
.......E....
======================================================================
ERROR: test_negative_feet (__main__.Feet_to_inches_testing)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "C:\Users\Leress\Desktop\SoftwareTesting\BMI_test.py", line 41, in test_n
egative_feet
    self.assertRaises(OutOfRangeError,ft_to_in, -1)
NameError: global name 'OutOfRangeError' is not defined

----------------------------------------------------------------------
Ran 12 tests in 0.125s

FAILED (errors=1)
>>> ================================ RESTART ================================
>>> 
.......E....
======================================================================
ERROR: test_negative_feet (__main__.Feet_to_inches_testing)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "C:\Users\Leress\Desktop\SoftwareTesting\BMI_test.py", line 41, in test_n
egative_feet
    self.assertRaises(ft_to_in.OutOfRangeError,ft_to_in, -1)
AttributeError: 'function' object has no attribute 'OutOfRangeError'

----------------------------------------------------------------------
Ran 12 tests in 0.070s

FAILED (errors=1)
>>> 
```

Figure 3b: *This image shows a* ***passed*** *negative input test during the* ***BMI*** *section*

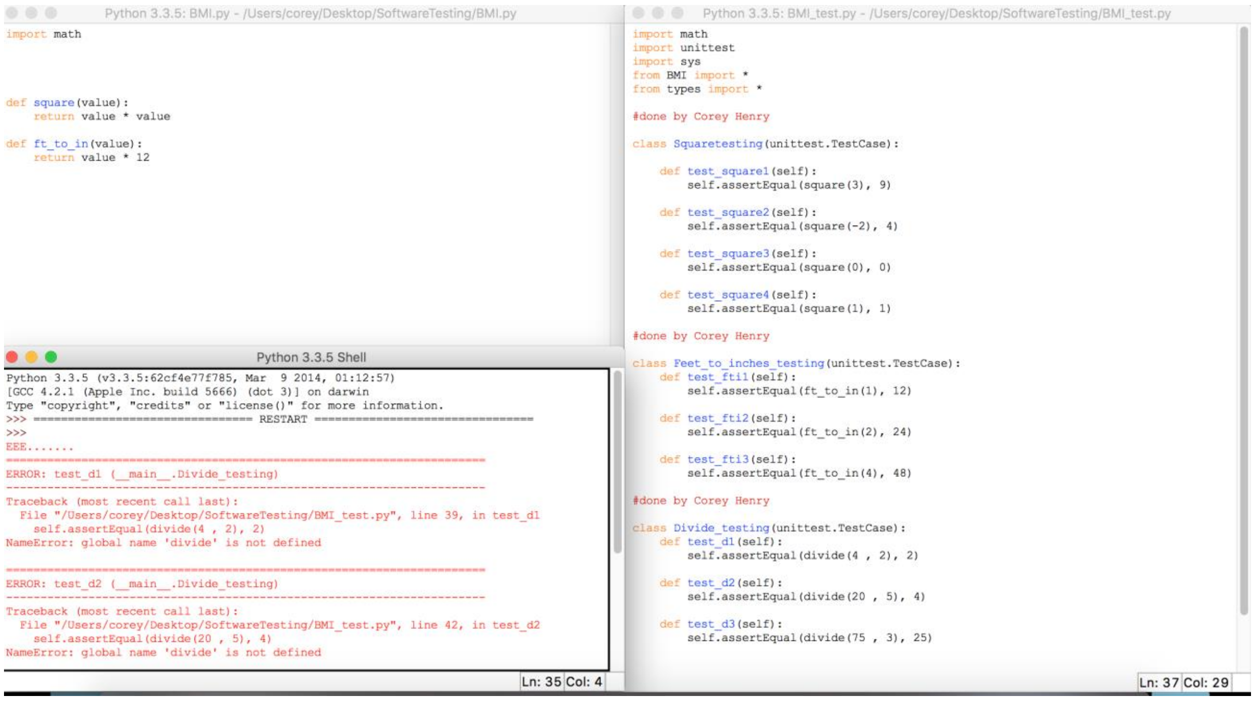


```
    def test_square3(self):
        self.assertEqual(square(0), 0)

    def test_square4(self):
        self.assertEqual(square(1), 1)

#done by Corey Henry

class Feet_to_inches_testing(unittest.TestCase):
    def test_fti1(self):
        self.assertEqual(ft_to_in(1), 12)

    def test_fti2(self):
        self.assertEqual(ft_to_in(2), 24)

    def test_fti3(self):
        self.assertEqual(ft_to_in(4), 48)

    #Jon Williams

    def test_decimal_feet(self):
        self.assertEqual(ft_to_in(1.5), 18)

    def test_negative_feet(self):
        self.assertRaises(OutOfRangeError,ft_to_in, -1)

#done by Corey Henry

class Divide_testing(unittest.TestCase):
    def test_d1(self):
        self.assertEqual(divide(4 , 2), 2)

    def test_d2(self):
        self.assertEqual(divide(20 , 5), 4)

    def test_d3(self):
        self.assertEqual(divide(75 , 3), 25)
```

```
>>> ================================ RESTART ================================
>>> 
.......E....
======================================================================
ERROR: test_negative_feet (__main__.Feet_to_inches_testing)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "C:\Users\Leress\Desktop\SoftwareTesting\BMI_test.py", line 41, in test_n
egative_feet
    self.assertRaises(ft_to_in.OutOfRangeError,ft_to_in, -1)
AttributeError: 'function' object has no attribute 'OutOfRangeError'

----------------------------------------------------------------------
Ran 12 tests in 0.108s

FAILED (errors=1)
>>> ================================ RESTART ================================
>>> 
.......E....
======================================================================
ERROR: test_negative_feet (__main__.Feet_to_inches_testing)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "C:\Users\Leress\Desktop\SoftwareTesting\BMI_test.py", line 41, in test_n
egative_feet
    self.assertRaises(BMI.OutOfRangeError,ft_to_in, -1)
NameError: global name 'BMI' is not defined

----------------------------------------------------------------------
Ran 12 tests in 0.065s

FAILED (errors=1)
>>> ================================ RESTART ================================
>>> 
............
----------------------------------------------------------------------
Ran 12 tests in 0.062s

OK
>>> 
```

Test-Driven Development (TDD) Screenshots Continued…

Figure 4a: *This image shows a **failed** division test during the **BMI** section*

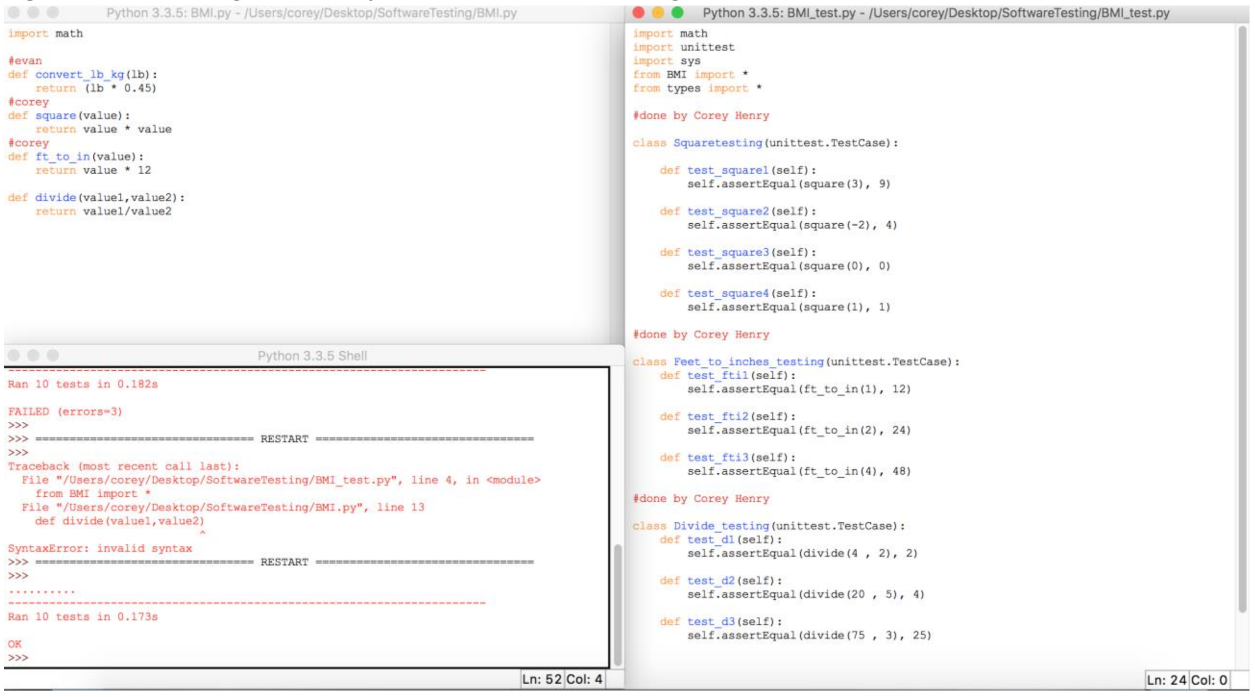

Python 3.3.5: BMI.py - /Users/corey/Desktop/SoftwareTesting/BMI.py
Python 3.3.5: BMI_test.py - /Users/corey/Desktop/SoftwareTesting/BMI_test.py
Python 3.3.5 Shell
Ln: 35 Col: 4
Ln: 37 Col: 29

```python
import math



def square(value):
    return value * value

def ft_to_in(value):
    return value * 12
```

```
Python 3.3.5 (v3.3.5:62cf4e77f785, Mar  9 2014, 01:12:57)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
EEE.......
======================================================================
ERROR: test_d1 (__main__.Divide_testing)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "/Users/corey/Desktop/SoftwareTesting/BMI_test.py", line 39, in test_d1
    self.assertEqual(divide(4 , 2), 2)
NameError: global name 'divide' is not defined

======================================================================
ERROR: test_d2 (__main__.Divide_testing)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "/Users/corey/Desktop/SoftwareTesting/BMI_test.py", line 42, in test_d2
    self.assertEqual(divide(20 , 5), 4)
NameError: global name 'divide' is not defined
```

```python
import math
import unittest
import sys
from BMI import *
from types import *

#done by Corey Henry

class Squaretesting(unittest.TestCase):

    def test_square1(self):
        self.assertEqual(square(3), 9)

    def test_square2(self):
        self.assertEqual(square(-2), 4)

    def test_square3(self):
        self.assertEqual(square(0), 0)

    def test_square4(self):
        self.assertEqual(square(1), 1)

#done by Corey Henry

class Feet_to_inches_testing(unittest.TestCase):
    def test_fti1(self):
        self.assertEqual(ft_to_in(1), 12)

    def test_fti2(self):
        self.assertEqual(ft_to_in(2), 24)

    def test_fti3(self):
        self.assertEqual(ft_to_in(4), 48)

#done by Corey Henry

class Divide_testing(unittest.TestCase):
    def test_d1(self):
        self.assertEqual(divide(4 , 2), 2)

    def test_d2(self):
        self.assertEqual(divide(20 , 5), 4)

    def test_d3(self):
        self.assertEqual(divide(75 , 3), 25)
```

Figure 4b: *This image shows a **passed** division test during the **BMI** section*


Python 3.3.5: BMI.py - /Users/corey/Desktop/SoftwareTesting/BMI.py
Python 3.3.5: BMI_test.py - /Users/corey/Desktop/SoftwareTesting/BMI_test.py
Python 3.3.5 Shell
Ln: 52 Col: 4
Ln: 24 Col: 0

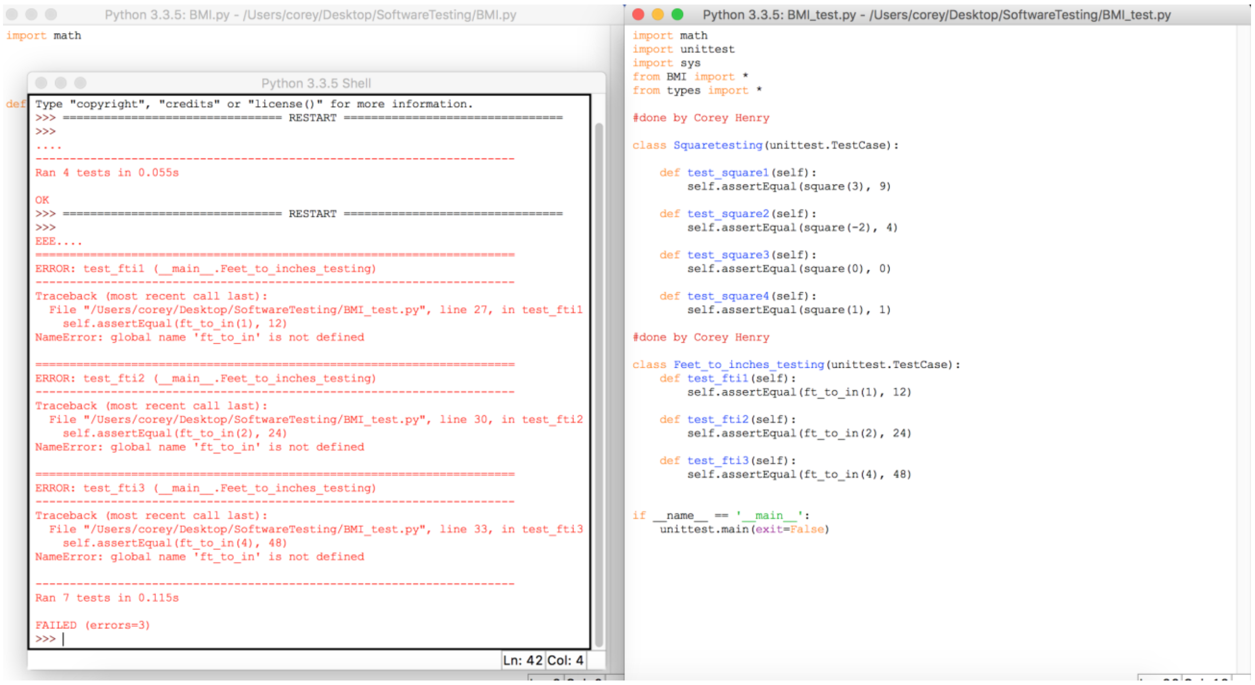
```python
import math

#evan
def convert_lb_kg(lb):
    return (lb * 0.45)
#corey
def square(value):
    return value * value
#corey
def ft_to_in(value):
    return value * 12

def divide(value1,value2):
    return value1/value2
```

```
----------------------------------------------------------------------
Ran 10 tests in 0.182s

FAILED (errors=3)
>>>
>>> ================================ RESTART ================================
>>>
Traceback (most recent call last):
  File "/Users/corey/Desktop/SoftwareTesting/BMI_test.py", line 4, in <module>
    from BMI import *
  File "/Users/corey/Desktop/SoftwareTesting/BMI.py", line 13
    def divide(value1,value2)
                            ^
SyntaxError: invalid syntax
>>> ================================ RESTART ================================
>>>
..........
----------------------------------------------------------------------
Ran 10 tests in 0.173s

OK
>>>
```

```python
import math
import unittest
import sys
from BMI import *
from types import *

#done by Corey Henry

class Squaretesting(unittest.TestCase):

    def test_square1(self):
        self.assertEqual(square(3), 9)

    def test_square2(self):
        self.assertEqual(square(-2), 4)

    def test_square3(self):
        self.assertEqual(square(0), 0)

    def test_square4(self):
        self.assertEqual(square(1), 1)

#done by Corey Henry

class Feet_to_inches_testing(unittest.TestCase):
    def test_fti1(self):
        self.assertEqual(ft_to_in(1), 12)

    def test_fti2(self):
        self.assertEqual(ft_to_in(2), 24)

    def test_fti3(self):
        self.assertEqual(ft_to_in(4), 48)

#done by Corey Henry

class Divide_testing(unittest.TestCase):
    def test_d1(self):
        self.assertEqual(divide(4 , 2), 2)

    def test_d2(self):
        self.assertEqual(divide(20 , 5), 4)

    def test_d3(self):
        self.assertEqual(divide(75 , 3), 25)
```

Test-Driven Development (TDD) Screenshots Continued…

Figure 5a: *This image shows a **failed** conversion from feet to inches test during the **BMI** section*

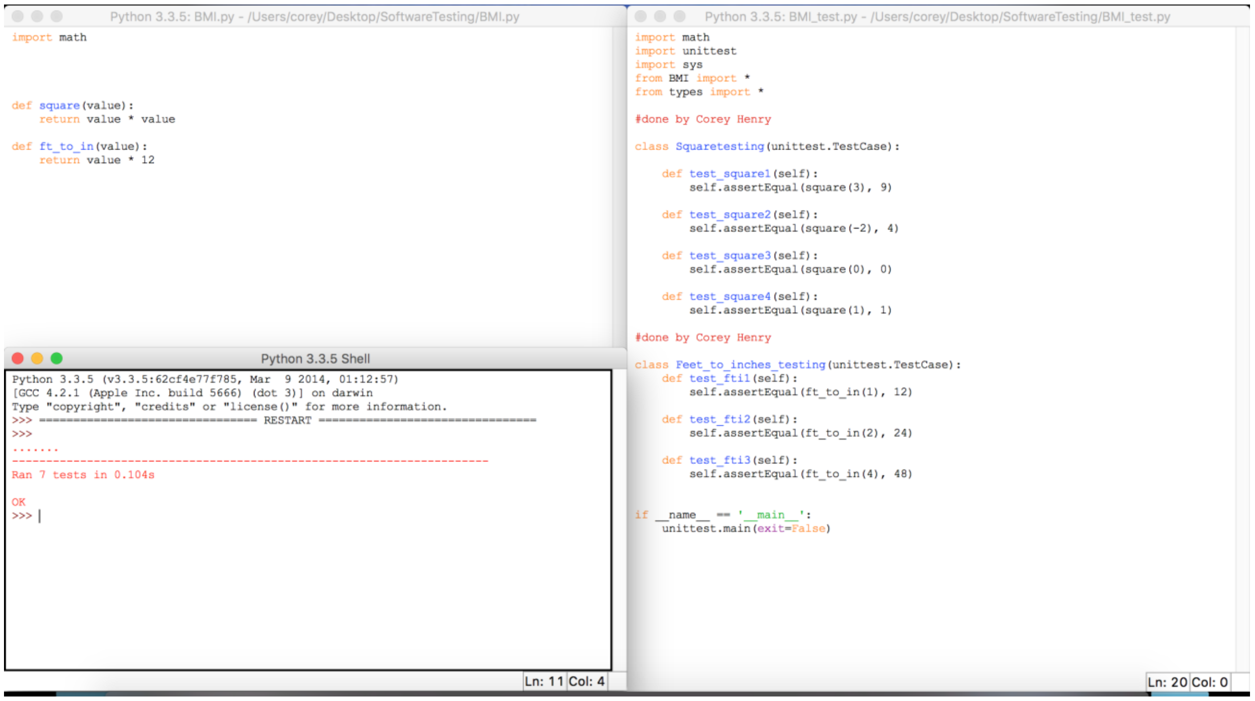

Python 3.3.5: BMI.py - /Users/corey/Desktop/SoftwareTesting/BMI.py
Python 3.3.5 Shell
Ln: 42 Col: 4
Python 3.3.5: BMI_test.py - /Users/corey/Desktop/SoftwareTesting/BMI_test.py

```
import math

def
```

```
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
....
----------------------------------------------------------------------
Ran 4 tests in 0.055s

OK
>>> ================================ RESTART ================================
>>>
EEE....
======================================================================
ERROR: test_fti1 (__main__.Feet_to_inches_testing)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "/Users/corey/Desktop/SoftwareTesting/BMI_test.py", line 27, in test_fti1
    self.assertEqual(ft_to_in(1), 12)
NameError: global name 'ft_to_in' is not defined

======================================================================
ERROR: test_fti2 (__main__.Feet_to_inches_testing)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "/Users/corey/Desktop/SoftwareTesting/BMI_test.py", line 30, in test_fti2
    self.assertEqual(ft_to_in(2), 24)
NameError: global name 'ft_to_in' is not defined

======================================================================
ERROR: test_fti3 (__main__.Feet_to_inches_testing)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "/Users/corey/Desktop/SoftwareTesting/BMI_test.py", line 33, in test_fti3
    self.assertEqual(ft_to_in(4), 48)
NameError: global name 'ft_to_in' is not defined

----------------------------------------------------------------------
Ran 7 tests in 0.115s

FAILED (errors=3)
>>> |
```

```
import math
import unittest
import sys
from BMI import *
from types import *

#done by Corey Henry

class Squaretesting(unittest.TestCase):

    def test_square1(self):
        self.assertEqual(square(3), 9)

    def test_square2(self):
        self.assertEqual(square(-2), 4)

    def test_square3(self):
        self.assertEqual(square(0), 0)

    def test_square4(self):
        self.assertEqual(square(1), 1)

#done by Corey Henry

class Feet_to_inches_testing(unittest.TestCase):
    def test_fti1(self):
        self.assertEqual(ft_to_in(1), 12)

    def test_fti2(self):
        self.assertEqual(ft_to_in(2), 24)

    def test_fti3(self):
        self.assertEqual(ft_to_in(4), 48)


if __name__ == '__main__':
    unittest.main(exit=False)
```

Figure 5b: *This image shows a **passed** conversion from feet to inches test during the **BMI** section*


Python 3.3.5: BMI.py - /Users/corey/Desktop/SoftwareTesting/BMI.py
Python 3.3.5 Shell
Ln: 11 Col: 4
Python 3.3.5: BMI_test.py - /Users/corey/Desktop/SoftwareTesting/BMI_test.py
Ln: 20 Col: 0

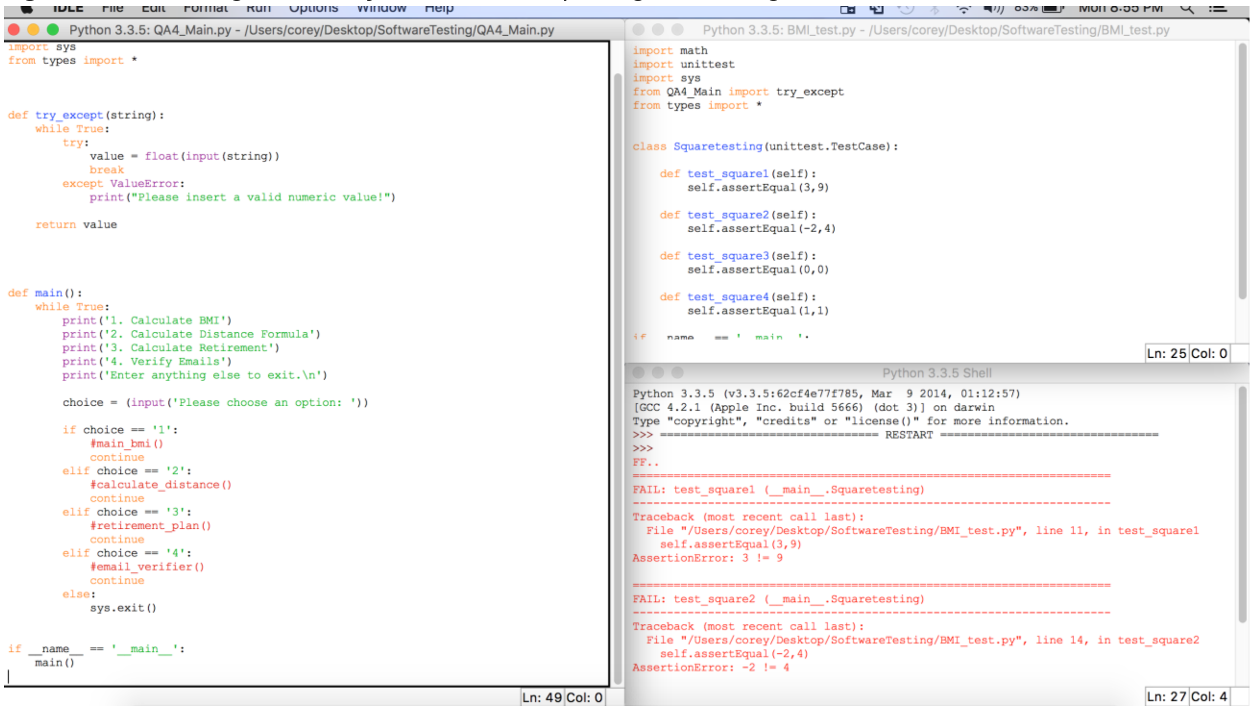
```
import math


def square(value):
    return value * value

def ft_to_in(value):
    return value * 12
```

```
Python 3.3.5 (v3.3.5:62cf4e77f785, Mar  9 2014, 01:12:57)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
.......
----------------------------------------------------------------------
Ran 7 tests in 0.104s

OK
>>> |
```

```
import math
import unittest
import sys
from BMI import *
from types import *

#done by Corey Henry

class Squaretesting(unittest.TestCase):

    def test_square1(self):
        self.assertEqual(square(3), 9)

    def test_square2(self):
        self.assertEqual(square(-2), 4)

    def test_square3(self):
        self.assertEqual(square(0), 0)

    def test_square4(self):
        self.assertEqual(square(1), 1)

#done by Corey Henry

class Feet_to_inches_testing(unittest.TestCase):
    def test_fti1(self):
        self.assertEqual(ft_to_in(1), 12)

    def test_fti2(self):
        self.assertEqual(ft_to_in(2), 24)

    def test_fti3(self):
        self.assertEqual(ft_to_in(4), 48)


if __name__ == '__main__':
    unittest.main(exit=False)
```

Test-Driven Development (TDD) Screenshots Continued…

Figure 6a: *This image shows a **failed** number squaring test during the **BMI** section*



Python 3.3.5: QA4_Main.py - /Users/corey/Desktop/SoftwareTesting/QA4_Main.py

```
import sys
from types import *


def try_except(string):
    while True:
        try:
            value = float(input(string))
            break
        except ValueError:
            print("Please insert a valid numeric value!")

    return value



def main():
    while True:
        print('1. Calculate BMI')
        print('2. Calculate Distance Formula')
        print('3. Calculate Retirement')
        print('4. Verify Emails')
        print('Enter anything else to exit.\n')

        choice = (input('Please choose an option: '))

        if choice == '1':
            #main_bmi()
            continue
        elif choice == '2':
            #calculate_distance()
            continue
        elif choice == '3':
            #retirement_plan()
            continue
        elif choice == '4':
            #email_verifier()
            continue
        else:
            sys.exit()


if __name__ == '__main__':
    main()
```

Ln: 49 Col: 0

Python 3.3.5: BMI_test.py - /Users/corey/Desktop/SoftwareTesting/BMI_test.py

```
import math
import unittest
import sys
from QA4_Main import try_except
from types import *


class Squaretesting(unittest.TestCase):

    def test_square1(self):
        self.assertEqual(3,9)

    def test_square2(self):
        self.assertEqual(-2,4)

    def test_square3(self):
        self.assertEqual(0,0)

    def test_square4(self):
        self.assertEqual(1,1)

if __name__ == '__main__':
```

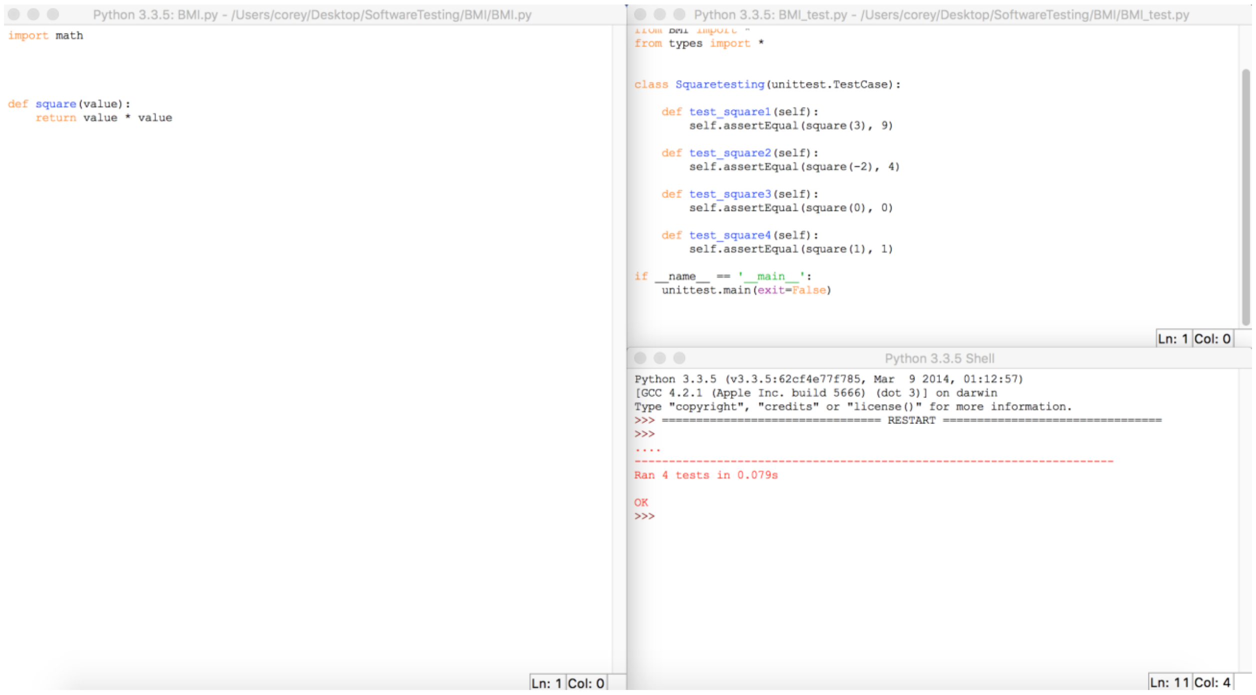
Ln: 25 Col: 0

Python 3.3.5 Shell

```
Python 3.3.5 (v3.3.5:62cf4e77f785, Mar  9 2014, 01:12:57)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
FF..
======================================================================
FAIL: test_square1 (__main__.Squaretesting)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "/Users/corey/Desktop/SoftwareTesting/BMI_test.py", line 11, in test_square1
    self.assertEqual(3,9)
AssertionError: 3 != 9

======================================================================
FAIL: test_square2 (__main__.Squaretesting)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "/Users/corey/Desktop/SoftwareTesting/BMI_test.py", line 14, in test_square2
    self.assertEqual(-2,4)
AssertionError: -2 != 4
```

Ln: 27 Col: 4

Figure 6b: *This image shows a **passed** number squaring test during the **BMI** section*



Python 3.3.5: BMI.py - /Users/corey/Desktop/SoftwareTesting/BMI/BMI.py

```
import math


def square(value):
    return value * value
```

Ln: 1 Col: 0

Python 3.3.5: BMI_test.py - /Users/corey/Desktop/SoftwareTesting/BMI/BMI_test.py

```
from BMI import *
from types import *


class Squaretesting(unittest.TestCase):

    def test_square1(self):
        self.assertEqual(square(3), 9)

    def test_square2(self):
        self.assertEqual(square(-2), 4)

    def test_square3(self):
        self.assertEqual(square(0), 0)

    def test_square4(self):
        self.assertEqual(square(1), 1)

if __name__ == '__main__':
    unittest.main(exit=False)
```

Ln: 1 Col: 0

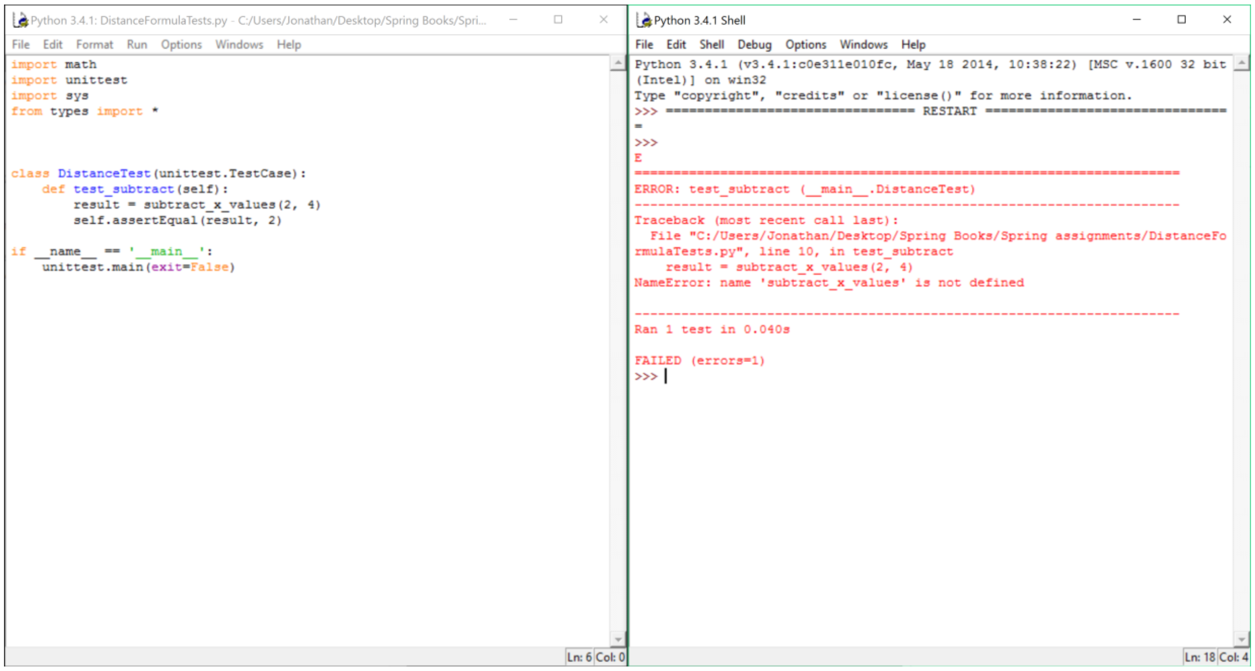
Python 3.3.5 Shell

```
Python 3.3.5 (v3.3.5:62cf4e77f785, Mar  9 2014, 01:12:57)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
....
----------------------------------------------------------------------
Ran 4 tests in 0.079s

OK
>>>
```

Ln: 11 Col: 4

Test-Driven Development (TDD) Screenshots Continued…

Figure 7a: *This image shows a **failed** number subtracting test during the **Distance Formula** section*

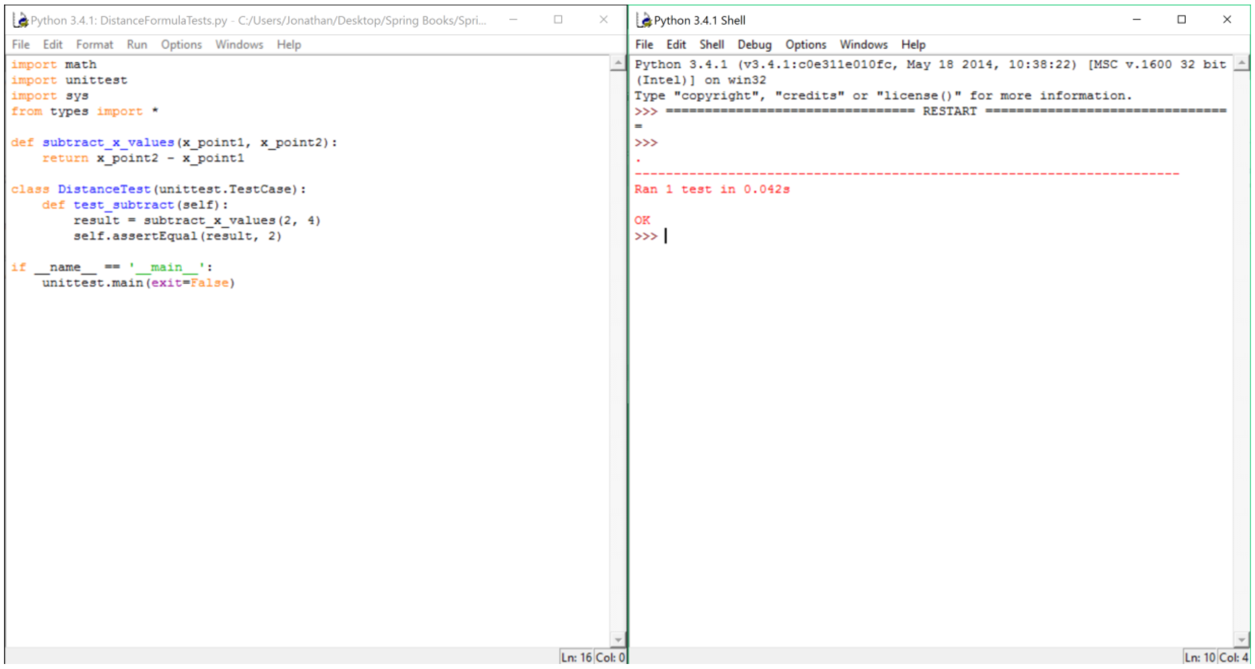


Python 3.4.1: DistanceFormulaTests.py - C:/Users/Jonathan/Desktop/Spring Books/Spri...
File Edit Format Run Options Windows Help
Ln: 6 Col: 0
Python 3.4.1 Shell
File Edit Shell Debug Options Windows Help
Ln: 18 Col: 4

```
import math
import unittest
import sys
from types import *


class DistanceTest(unittest.TestCase):
    def test_subtract(self):
        result = subtract_x_values(2, 4)
        self.assertEqual(result, 2)

if __name__ == '__main__':
    unittest.main(exit=False)
```

```
Python 3.4.1 (v3.4.1:c0e311e010fc, May 18 2014, 10:38:22) [MSC v.1600 32 bit
(Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
=
>>>
E
======================================================================
ERROR: test_subtract (__main__.DistanceTest)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "C:/Users/Jonathan/Desktop/Spring Books/Spring assignments/DistanceFo
rmulaTests.py", line 10, in test_subtract
    result = subtract_x_values(2, 4)
NameError: name 'subtract_x_values' is not defined

----------------------------------------------------------------------
Ran 1 test in 0.040s

FAILED (errors=1)
>>>
```

Figure 7b: *This image shows a **passed** number subtracting test during the **Distance Formula** section*



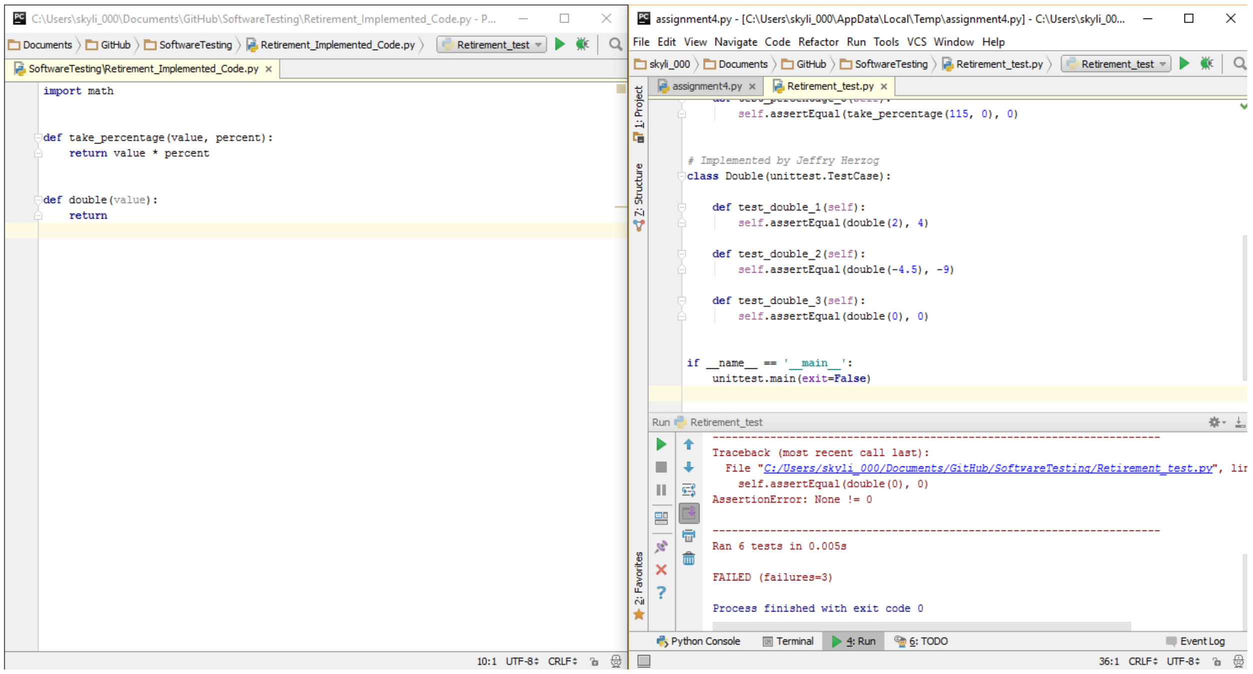
Python 3.4.1: DistanceFormulaTests.py - C:/Users/Jonathan/Desktop/Spring Books/Spri...
File Edit Format Run Options Windows Help
Ln: 16 Col: 0
Python 3.4.1 Shell
File Edit Shell Debug Options Windows Help
Ln: 10 Col: 4

```
import math
import unittest
import sys
from types import *

def subtract_x_values(x_point1, x_point2):
    return x_point2 - x_point1

class DistanceTest(unittest.TestCase):
    def test_subtract(self):
        result = subtract_x_values(2, 4)
        self.assertEqual(result, 2)

if __name__ == '__main__':
    unittest.main(exit=False)
```

```
Python 3.4.1 (v3.4.1:c0e311e010fc, May 18 2014, 10:38:22) [MSC v.1600 32 bit
(Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
=
>>>
.
----------------------------------------------------------------------
Ran 1 test in 0.042s

OK
>>>
```

Test-Driven Development (TDD) Screenshots Continued…

Figure 8a: *This image shows a* ***failed*** *number doubling test during the* ***Retirement*** *section*

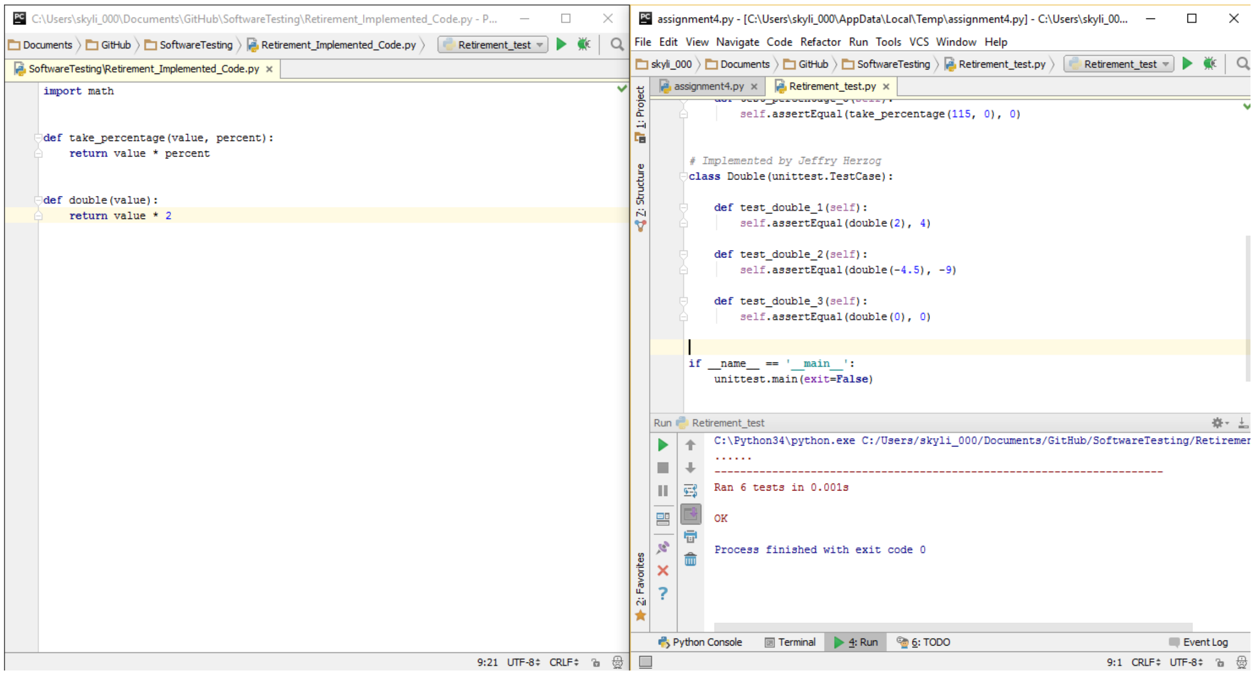

C:\Users\skyli_000\Documents\GitHub\SoftwareTesting\Retirement_Implemented_Code.py - P...
SoftwareTesting\Retirement_Implemented_Code.py
assignment4.py - [C:\Users\skyli_000\AppData\Local\Temp\assignment4.py] - C:\Users\skyli_00...
File Edit View Navigate Code Refactor Run Tools VCS Window Help
assignment4.py
Retirement_test.py
Run Retirement_test
Python Console
Terminal
4: Run
6: TODO
Event Log

```python
import math


def take_percentage(value, percent):
    return value * percent


def double(value):
    return
```

```python
        self.assertEqual(take_percentage(115, 0), 0)

# Implemented by Jeffry Herzog
class Double(unittest.TestCase):

    def test_double_1(self):
        self.assertEqual(double(2), 4)

    def test_double_2(self):
        self.assertEqual(double(-4.5), -9)

    def test_double_3(self):
        self.assertEqual(double(0), 0)

if __name__ == '__main__':
    unittest.main(exit=False)
```

```
----------------------------------------------------------------------
Traceback (most recent call last):
  File "C:/Users/skyli_000/Documents/GitHub/SoftwareTesting/Retirement_test.py", lir
    self.assertEqual(double(0), 0)
AssertionError: None != 0

----------------------------------------------------------------------
Ran 6 tests in 0.005s

FAILED (failures=3)

Process finished with exit code 0
```

Figure 8b: *This image shows a* ***passed*** *number doubling test during the* ***Retirement*** *section*


C:\Users\skyli_000\Documents\GitHub\SoftwareTesting\Retirement_Implemented_Code.py - P...
SoftwareTesting\Retirement_Implemented_Code.py
assignment4.py - [C:\Users\skyli_000\AppData\Local\Temp\assignment4.py] - C:\Users\skyli_00...
File Edit View Navigate Code Refactor Run Tools VCS Window Help
assignment4.py
Retirement_test.py
Run Retirement_test
Python Console
Terminal
4: Run
6: TODO
Event Log

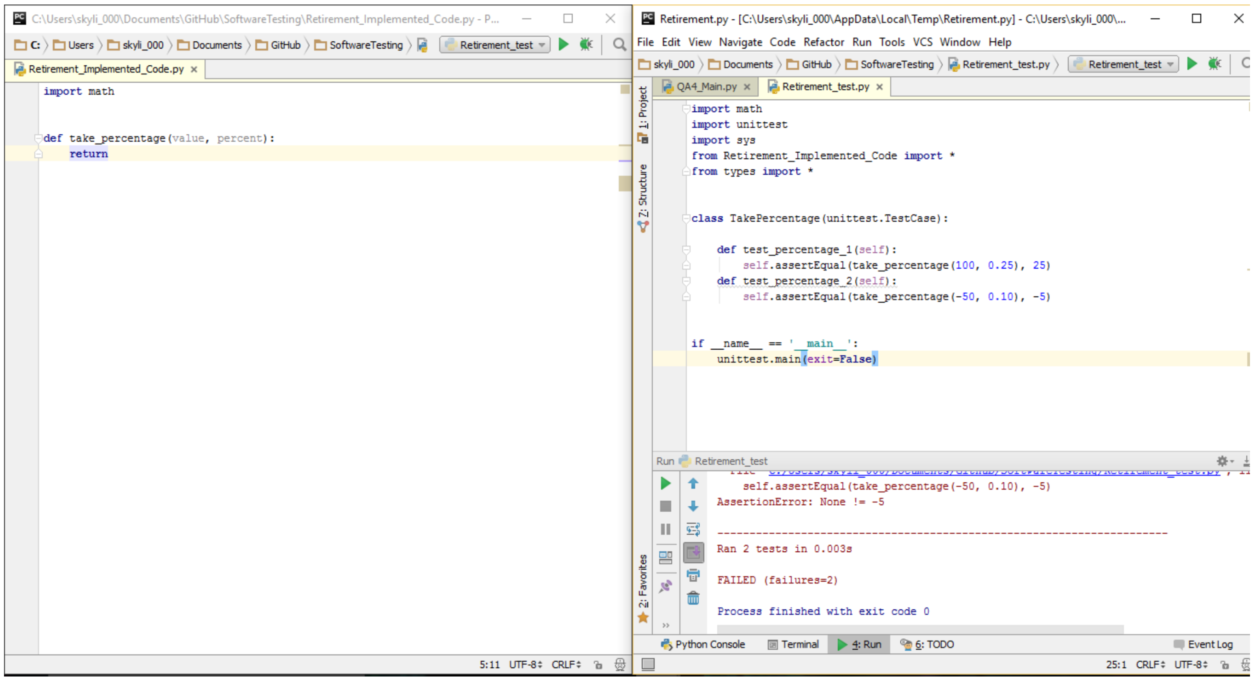
```python
import math


def take_percentage(value, percent):
    return value * percent


def double(value):
    return value * 2
```

```python
        self.assertEqual(take_percentage(115, 0), 0)

# Implemented by Jeffry Herzog
class Double(unittest.TestCase):

    def test_double_1(self):
        self.assertEqual(double(2), 4)

    def test_double_2(self):
        self.assertEqual(double(-4.5), -9)

    def test_double_3(self):
        self.assertEqual(double(0), 0)


if __name__ == '__main__':
    unittest.main(exit=False)
```

```
C:\Python34\python.exe C:/Users/skyli_000/Documents/GitHub/SoftwareTesting/Retiremer
......
----------------------------------------------------------------------
Ran 6 tests in 0.001s

OK

Process finished with exit code 0
```

Test-Driven Development (TDD) Screenshots Continued…

Figure 9a: *This image shows a* ***failed*** *percentage test during the* ***Retirement*** *section*



C:\Users\skyli_000\Documents\GitHub\SoftwareTesting\Retirement_Implemented_Code.py - P...

Retirement_Implemented_Code.py

```
import math


def take_percentage(value, percent):
    return
```

Retirement.py - [C:\Users\skyli_000\AppData\Local\Temp\Retirement.py] - C:\Users\skyli_000\...

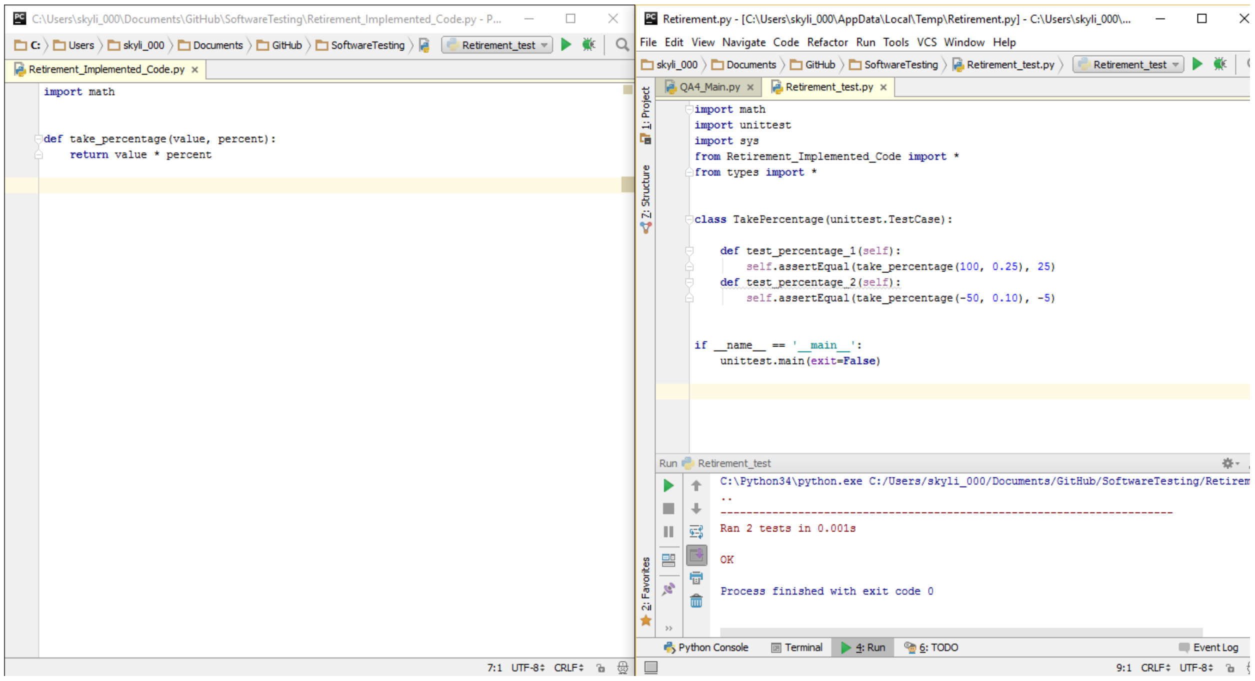
Retirement_test.py

```
import math
import unittest
import sys
from Retirement_Implemented_Code import *
from types import *


class TakePercentage(unittest.TestCase):

    def test_percentage_1(self):
        self.assertEqual(take_percentage(100, 0.25), 25)
    def test_percentage_2(self):
        self.assertEqual(take_percentage(-50, 0.10), -5)


if __name__ == '__main__':
    unittest.main(exit=False)
```

Run Retirement_test

```
    self.assertEqual(take_percentage(-50, 0.10), -5)
AssertionError: None != -5

----------------------------------------------------------------------
Ran 2 tests in 0.003s

FAILED (failures=2)

Process finished with exit code 0
```

Figure 9b: *This image shows a* ***passed*** *percentage test during the* ***Retirement*** *section*



C:\Users\skyli_000\Documents\GitHub\SoftwareTesting\Retirement_Implemented_Code.py - P...

Retirement_Implemented_Code.py

```
import math


def take_percentage(value, percent):
    return value * percent
```

Retirement.py - [C:\Users\skyli_000\AppData\Local\Temp\Retirement.py] - C:\Users\skyli_000\...

Retirement_test.py

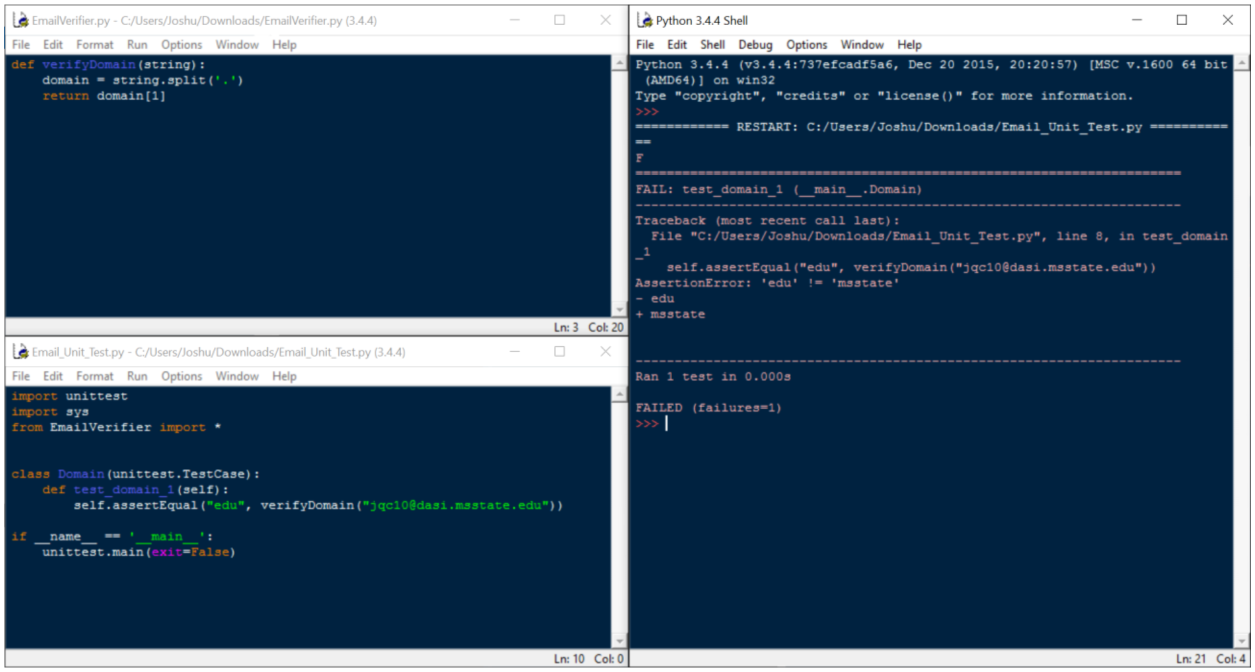
```
import math
import unittest
import sys
from Retirement_Implemented_Code import *
from types import *


class TakePercentage(unittest.TestCase):

    def test_percentage_1(self):
        self.assertEqual(take_percentage(100, 0.25), 25)
    def test_percentage_2(self):
        self.assertEqual(take_percentage(-50, 0.10), -5)


if __name__ == '__main__':
    unittest.main(exit=False)
```

Run Retirement_test

```
C:\Python34\python.exe C:/Users/skyli_000/Documents/GitHub/SoftwareTesting/Retirem
..
----------------------------------------------------------------------
Ran 2 tests in 0.001s

OK

Process finished with exit code 0
```

Test-Driven Development (TDD) Screenshots Continued…

Figure 10a: *This image shows a* ***failed*** *domain check during the* ***Verify Email*** *section*

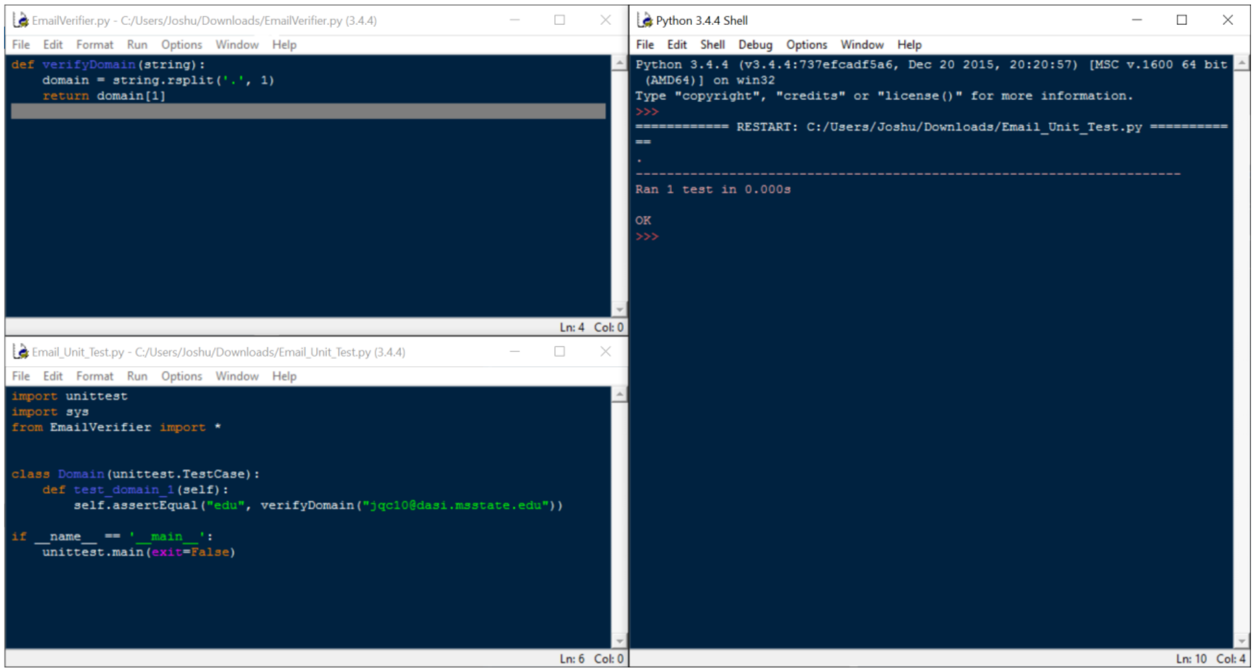

EmailVerifier.py - C:/Users/Joshu/Downloads/EmailVerifier.py (3.4.4)
File Edit Format Run Options Window Help
Ln: 3 Col: 20
Email_Unit_Test.py - C:/Users/Joshu/Downloads/Email_Unit_Test.py (3.4.4)
File Edit Format Run Options Window Help
Ln: 10 Col: 0
Python 3.4.4 Shell
File Edit Shell Debug Options Window Help
Ln: 21 Col: 4

```
def verifyDomain(string):
    domain = string.split('.')
    return domain[1]
```

```
import unittest
import sys
from EmailVerifier import *


class Domain(unittest.TestCase):
    def test_domain_1(self):
        self.assertEqual("edu", verifyDomain("jqc10@dasi.msstate.edu"))

if __name__ == '__main__':
    unittest.main(exit=False)
```

```
Python 3.4.4 (v3.4.4:737efcadf5a6, Dec 20 2015, 20:20:57) [MSC v.1600 64 bit
 (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> 
=========== RESTART: C:/Users/Joshu/Downloads/Email_Unit_Test.py =========
==
F
======================================================================
FAIL: test_domain_1 (__main__.Domain)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "C:/Users/Joshu/Downloads/Email_Unit_Test.py", line 8, in test_domain
_1
    self.assertEqual("edu", verifyDomain("jqc10@dasi.msstate.edu"))
AssertionError: 'edu' != 'msstate'
- edu
+ msstate


----------------------------------------------------------------------
Ran 1 test in 0.000s

FAILED (failures=1)
>>> 
```

Figure 10b: *This image shows a* ***passed*** *domain check during the* ***Verify Email*** *section*


EmailVerifier.py - C:/Users/Joshu/Downloads/EmailVerifier.py (3.4.4)
File Edit Format Run Options Window Help
Ln: 4 Col: 0
Email_Unit_Test.py - C:/Users/Joshu/Downloads/Email_Unit_Test.py (3.4.4)
File Edit Format Run Options Window Help
Ln: 6 Col: 0
Python 3.4.4 Shell
File Edit Shell Debug Options Window Help
Ln: 10 Col: 4

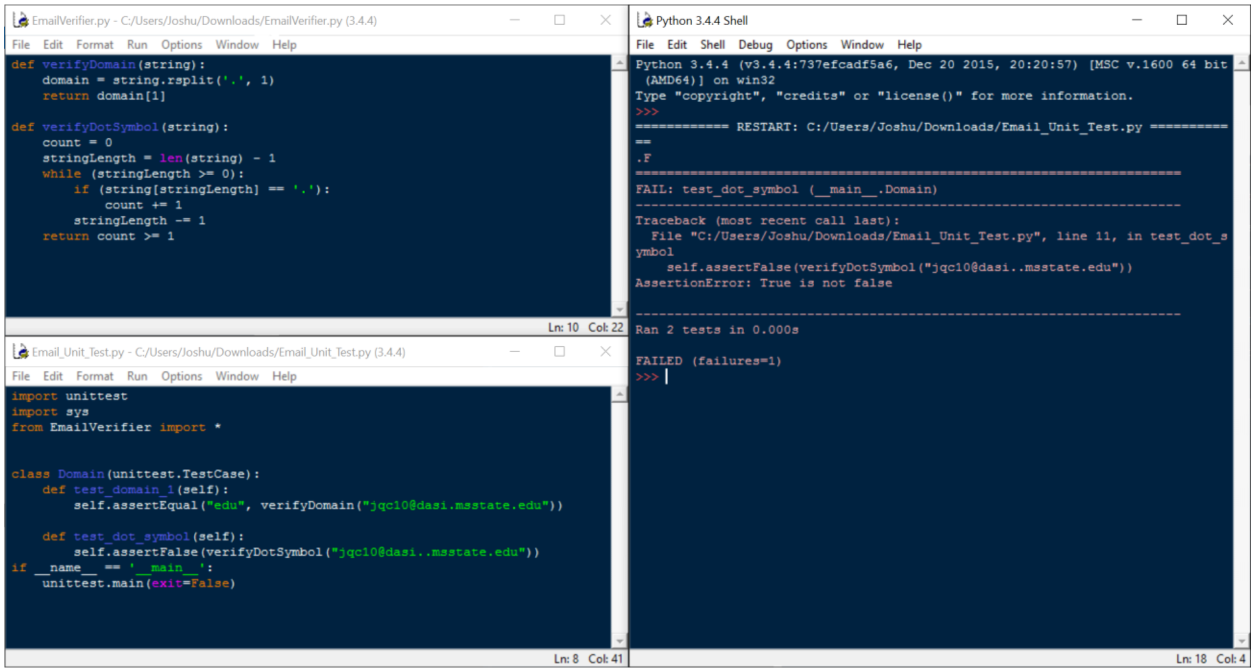
```
def verifyDomain(string):
    domain = string.rsplit('.', 1)
    return domain[1]
```

```
import unittest
import sys
from EmailVerifier import *


class Domain(unittest.TestCase):
    def test_domain_1(self):
        self.assertEqual("edu", verifyDomain("jqc10@dasi.msstate.edu"))

if __name__ == '__main__':
    unittest.main(exit=False)
```

```
Python 3.4.4 (v3.4.4:737efcadf5a6, Dec 20 2015, 20:20:57) [MSC v.1600 64 bit
 (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> 
=========== RESTART: C:/Users/Joshu/Downloads/Email_Unit_Test.py =========
==
.
----------------------------------------------------------------------
Ran 1 test in 0.000s

OK
>>> 
```

Test-Driven Development (TDD) Screenshots Continued…

Figure 11a: *This image shows a* ***failed*** *dot symbol check during the* ***Verify Email*** *section*



EmailVerifier.py - C:/Users/Joshu/Downloads/EmailVerifier.py (3.4.4)

```python
def verifyDomain(string):
    domain = string.rsplit('.', 1)
    return domain[1]

def verifyDotSymbol(string):
    count = 0
    stringLength = len(string) - 1
    while (stringLength >= 0):
        if (string[stringLength] == '.'):
            count += 1
        stringLength -= 1
    return count >= 1
```

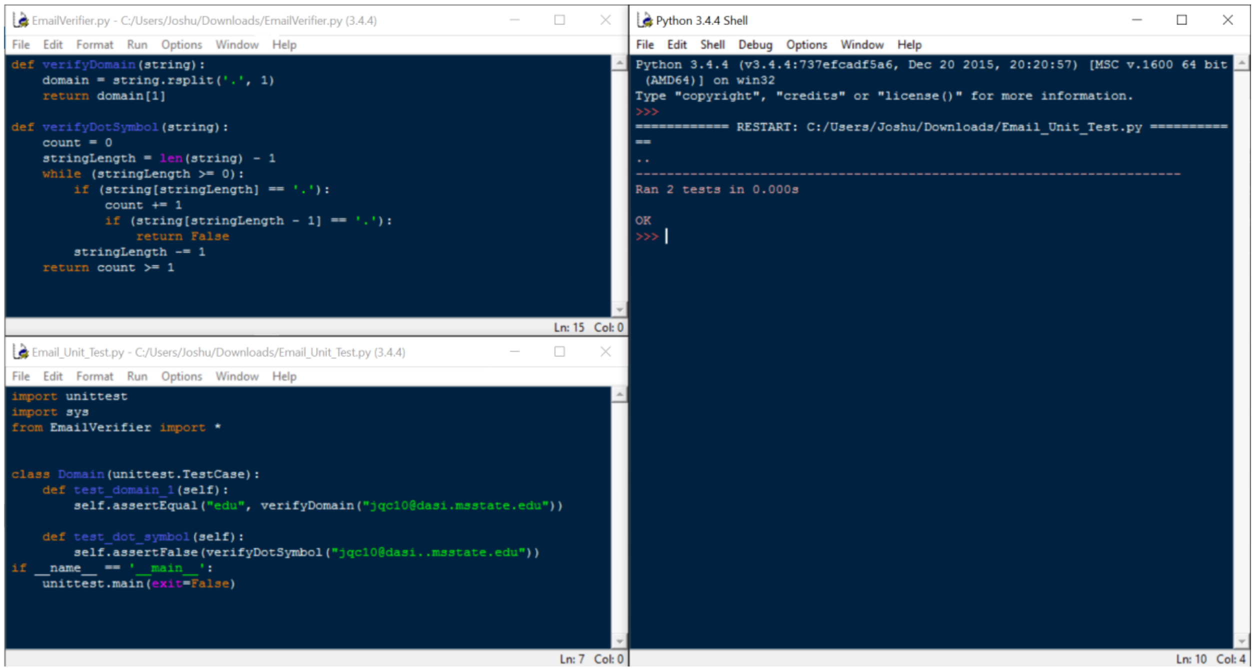
Email_Unit_Test.py - C:/Users/Joshu/Downloads/Email_Unit_Test.py (3.4.4)

```python
import unittest
import sys
from EmailVerifier import *


class Domain(unittest.TestCase):
    def test_domain_1(self):
        self.assertEqual("edu", verifyDomain("jqc10@dasi.msstate.edu"))

    def test_dot_symbol(self):
        self.assertFalse(verifyDotSymbol("jqc10@dasi..msstate.edu"))
if __name__ == '__main__':
    unittest.main(exit=False)
```

Python 3.4.4 Shell

```
Python 3.4.4 (v3.4.4:737efcadf5a6, Dec 20 2015, 20:20:57) [MSC v.1600 64 bit
 (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> 
============ RESTART: C:/Users/Joshu/Downloads/Email_Unit_Test.py ==========
==
.F
======================================================================
FAIL: test_dot_symbol (__main__.Domain)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "C:/Users/Joshu/Downloads/Email_Unit_Test.py", line 11, in test_dot_s
ymbol
    self.assertFalse(verifyDotSymbol("jqc10@dasi..msstate.edu"))
AssertionError: True is not false

----------------------------------------------------------------------
Ran 2 tests in 0.000s

FAILED (failures=1)
>>> 
```

Figure 11b: *This image shows a* ***passed*** *dot symbol check during the* ***Verify Email*** *section*



EmailVerifier.py - C:/Users/Joshu/Downloads/EmailVerifier.py (3.4.4)

```python
def verifyDomain(string):
    domain = string.rsplit('.', 1)
    return domain[1]

def verifyDotSymbol(string):
    count = 0
    stringLength = len(string) - 1
    while (stringLength >= 0):
        if (string[stringLength] == '.'):
            count += 1
            if (string[stringLength - 1] == '.'):
                return False
        stringLength -= 1
    return count >= 1
```

Email_Unit_Test.py - C:/Users/Joshu/Downloads/Email_Unit_Test.py (3.4.4)

```python
import unittest
import sys
from EmailVerifier import *


class Domain(unittest.TestCase):
    def test_domain_1(self):
        self.assertEqual("edu", verifyDomain("jqc10@dasi.msstate.edu"))

    def test_dot_symbol(self):
        self.assertFalse(verifyDotSymbol("jqc10@dasi..msstate.edu"))
if __name__ == '__main__':
    unittest.main(exit=False)
```

Python 3.4.4 Shell

```
Python 3.4.4 (v3.4.4:737efcadf5a6, Dec 20 2015, 20:20:57) [MSC v.1600 64 bit
 (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> 
============ RESTART: C:/Users/Joshu/Downloads/Email_Unit_Test.py ==========
==
..
----------------------------------------------------------------------
Ran 2 tests in 0.000s

OK
>>> 
```

Instructions for Setting Up Code and Unit Test

STEP 1: Download the latest version of Python.

STEP 2: Navigate to the following link:
https://github.com/JoshuaQChurch/SoftwareTesting

STEP 3: Download the following files:

- def_BMI.py
- def_DistanceFormula.py
- def_Retirement.py
- def_EmailVerifier.py
- Unit_Tests_BMI.py
- Unit_Tests_DistanceFormula.py
- Unit_Tests_Retirement.py
- Unit_Tests_EmailVerifier.py
- QA4_Main.py

STEP 4: Make sure all of the files above are located within the same file location

STEP 5: Open QA4_Main.py and execute the program

STEP 6: Follow the steps on screen within the main menu.

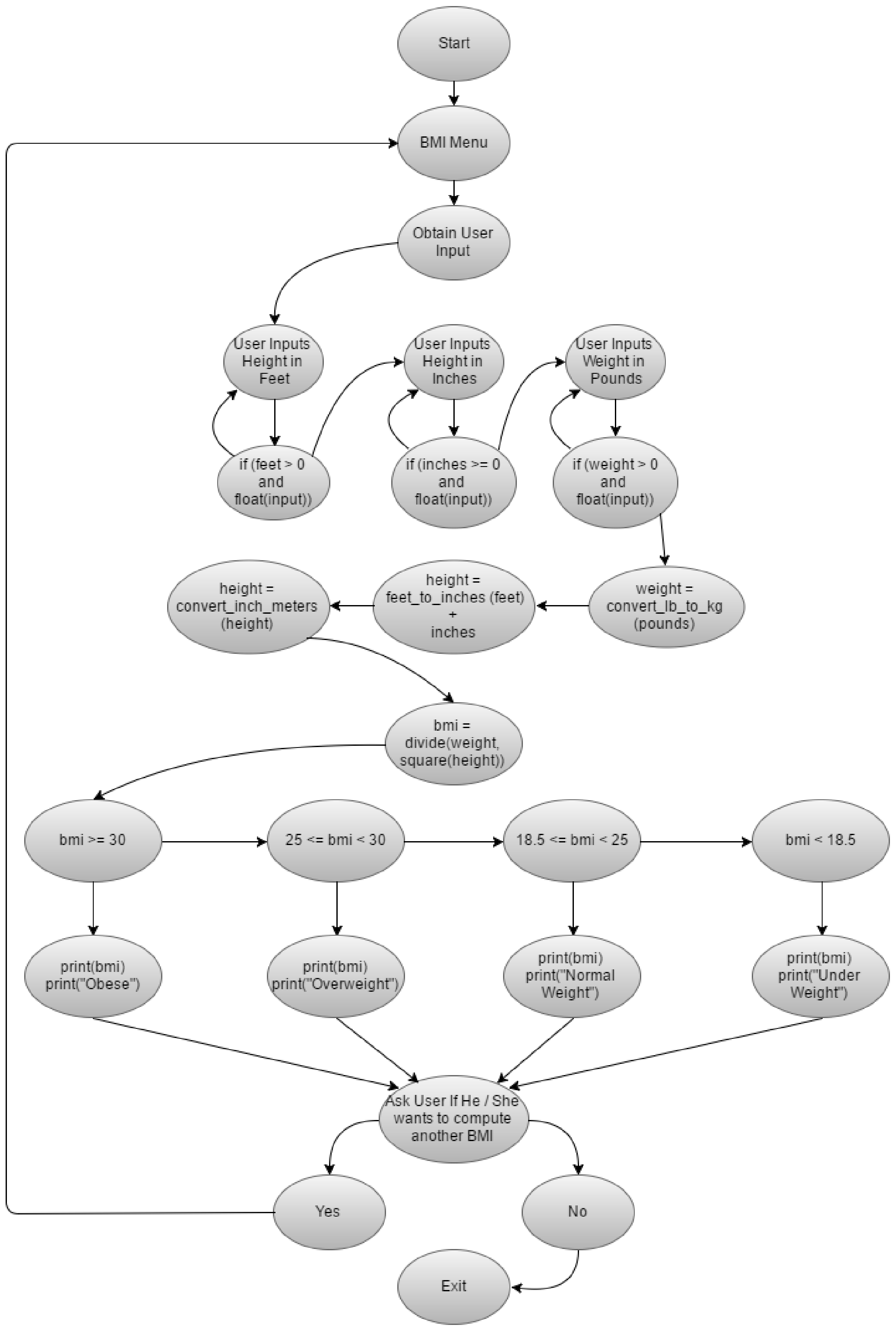
Python Unit Test Framework Report

#	Test Case	Control	Comparison	Value	Expected	Result
1	Square	Positive	Equal	3	9	pass
2	Square	Negative	Equal	-2	4	pass
3	Square	Zero	Equal	0	0	pass
4	Square	One	Equal	1	1	pass
5	Feet to Inches	One	Equal	1	12	pass
6	Feet to Inches	Two	Equal	2	24	pass
7	Feet to Inches	Four	Equal	4	48	pass
8	Feet to Inches	Decimal	Equal	1.5	18	pass
9	Feet to Inches	Negative	Raises	-1	error	pass
10	Inches to Meters	Negative	Raises	-1	error	pass
11	Pounds to Kilograms	Negative	Raises	-1	error	pass
12	Divide	Positive	Equal	(4, 2)	2	pass
13	Divide	Positive	Equal	(20, 5)	4	pass
14	Divide	Positive	Equal	(75, 3)	25	pass
15	Divide	Negative	Raises	(-1, 0)	error	pass
16	Divide	Negative	Raises	(0, -1)	error	pass
17	Divide	Double Negative	Raises	(-1, -1)	error	pass
18	Divide	Divide by Zero	Raises	(1, 0)	error	pass
19	Percentage	Positive	Equal	(100, 0.25)	25	pass
20	Percentage	Zero	Equal	(115, 0.0)	0	pass
21	Percentage	Negative	Raises	(-1, 0.10)	error	pass
22	Percentage	Negative	Raises	(1, -0.10)	error	pass
23	Double	Positive	Equal	2	4	pass
24	Double	Zero	Equal	0	0	pass
25	Double	Negative	Raises	-1	error	pass
26	Aging	Negative	Raises	-1	error	pass
27	Retirement	Negative	Raises	(1, 100, .1, -2)	error	pass
28	Retirement	Negative	Raises	(-1, 100, .1, -1)	error	pass
29	X Subtract	Positive	Equal	(2, 4)	2	pass
30	X Subtract	Double Negative	Equal	(-1, -1)	0	pass

Python Unit Test Framework Report Continued…

31	X Subtract	Negative	Equal	(1, -1)	-2	pass
32	Y Subtract	Positive	Equal	(2, 4)	2	pass
33	Y Subtract	Double Negative	Equal	(-1, -1)	0	pass
34	Y Subtract	Negative	Equal	(1, -1)	-2	pass
35	Square X	Positive	Equal	2	4	pass
36	Square X	Negative	Equal	-2	4	pass
37	Square X	Zero	Equal	0	0	pass
38	Square Y	Positive	Equal	2	4	pass
39	Square Y	Negative	Equal	-2	4	pass
40	Square Y	Zero	Equal	0	0	pass
41	Add Value	Positive	Equal	(4, 4)	8	pass
42	Add Value	Zero	Equal	(0, 0)	0	pass
43	Add Value	Negative	Equal	(-1, -2)	-3	pass
44	Get Distance	Positive	Equal	4	2	pass
45	Get Distance	Negative	Raises	-4	error	pass
46	Distance Formula	Positive	Equal	(2, 4, 5, 8)	5	pass
47	Distance Formula	Identical	Equal	(4, 4, 4, 4)	0	pass
48	Distance Formula	Negative	Equal	(4, 7, -2, 5)	sqrt(40)	pass
49	Distance Formula	Zero	Equal	(0, 0, 0, 0)	0	pass
50	Distance Formula	Negative X	Equal	(-1, 0, -2, 0)	1	pass
51	Distance Formula	All Negative	Equal	(-1, -1, -2, -1)	1	pass
52	Distance Formula	Negative Y	Equal	(1, -1, 2, -1)	1	pass
53	Distance Formula	Positive Y1	Equal	(-1, 1, -1, -1)	2	pass
54	Distance Formula	Positive X2	Equal	(-1, -1, 1, -1)	2	pass
55	Distance Formula	Negative X2 & Y2	Almost	(1, 1, -1, -1)	2.8284	pass
56	Distance Formula	Positive X1 & Y1	Almost	(1, 1, -1, -1)	2.8284	pass
57	Distance Formula	Inverse Negative	Almost	(-1, 1, 1, -1)	2.8284	pass
58	Domain	Dot Count 2	Not Equal	jqc10@dasi.ms.edu	edu	pass
59	Dot Symbol	Consecutive Dot	FALSE	jqc10@dasi..ms.edu	FALSE	pass
60	Dot Symbol	Dot Count 2	TRUE	jqc10@dasi.ms.edu	TRUE	pass
61	At Symbol	At Symbol Exists	TRUE	jjh258@ms.edu	TRUE	pass
62	At Symbol	At Missing	FALSE	jjh258@ms.edu	FALSE	pass
63	Verify Content	Domain Length	TRUE	randEmail@joshu.co	TRUE	pass
64	Verify Content	Dot to At	FALSE	shouldFail.@.com	FALSE	pass
65	Verify Content	Consecutive At	FALSE	WillFail@@test.com	FALSE	pass

Control Flow Graph for BMI


Start
BMI Menu
Obtain User Input
User Inputs Height in Feet
User Inputs Height in Inches
User Inputs Weight in Pounds
if (feet > 0 and float(input))
if (inches >= 0 and float(input))
if (weight > 0 and float(input))
height = convert_inch_meters (height)
height = feet_to_inches (feet) + inches
weight = convert_lb_to_kg (pounds)
bmi = divide(weight, square(height))
bmi >= 30
25 <= bmi < 30
18.5 <= bmi < 25
bmi < 18.5
print(bmi) print("Obese")
print(bmi) print("Overweight")
print(bmi) print("Normal Weight")
print(bmi) print("Under Weight")
Ask User If He / She wants to compute another BMI
Yes
No
Exit

Conclusion and Final Thoughts

The Test-Driven Development process was initially difficult to understand, but as our group developed more tests and developed new tactics for the process, we began to understand the overall importance of the TDD cycle. On smaller-scaled projects such as this, we began to realize that the TDD process wasn’t as beneficial as it was initially perceived to be. However, through much research and practice, we now understand that the TDD cycle would be immensely beneficial on larger-scaled projects and would ensure that the code functions as intended.

Overall, this project was challenging and required time to fully understand many of its specifications, but it taught us all what TDD can truly do and its benefits and drawbacks.