Unit Testing 1 - Exercise 2 Solution

1. Rectangle.py

```
class Rectangle:
def __init__(self, width, length):
    if not isinstance(width, int) or not isinstance(length, int):
        raise TypeError('Invalid Type Entered, Int type Expected')
    if (isinstance(width, int) and width <= 0) or \
        (isinstance(length, int) and length <= 0):
        raise ValueError('Invalid Value Entered, Value greater than 0 Expected')
    self.width = width
    self.length = length

def get_area(self):
    return self.width * self.length
```

test_rectangle.py

```
from Rectangle import Rectangle
import pytest


# The test functions to be executed by PyTest

def test_can_calc_area():
    rectangle = Rectangle(2, 3)
    assert rectangle.get_area() == 6, "incorrect area"

# edge case
def test_can_calc_area_for_big_numbers():
    rectangle = Rectangle(20000, 30000)
    assert rectangle.get_area() == 600000000, "incorrect area for big numbers"

def test_can_calc_area_after_width_change():
    rectangle = Rectangle(2, 3)
    rectangle.width = 10
    assert rectangle.get_area() == 30, "incorrect area after width change"

def test_type_error_for_non_int_arg():
    with pytest.raises(TypeError):
        Rectangle('1', 2)

def test_type_error_for_non_positive_arg():
    with pytest.raises(ValueError):
        Rectangle(-1, 2)
```

Bonus

Rectangle.py

```
class Rectangle:
    def __init__(self, width, length):
        self.width = width
        self.length = length

    def get_area(self):
        return self.width * self.length

    @property
    def width(self):
        return self._width

    @width.setter
    def width(self, width):
        if not isinstance(width, int):
            raise TypeError('Invalid Type Entered, Int type Expected')
        if isinstance(width, int) and width <= 0:
            raise ValueError('Invalid Value Entered, Value greater than 0 Expected')
        self._width = width

    @property
    def length(self):
        return self._length

    @length.setter
    def length(self, length):
        if not isinstance(length, int):
            raise TypeError('Invalid Type Entered, Int type Expected')
        if isinstance(length, int) and length <= 0:
            raise ValueError('Invalid Value Entered, Value greater than 0 Expected')
        self._length = length
```

test_rectangle.py

```
# add these tests to the end of above test file
def test_type_error_with_attr_change_to_non_int_value():
    with pytest.raises(ValueError):
        rectangle = Rectangle(2, 3)
        rectangle.width = '1'

def test_type_error_with_attr_change_to_non_positive_value():
    with pytest.raises(ValueError):
        rectangle = Rectangle(2, 3)
        rectangle.width = -3
```