Coverage for **triangle.py** : 100%

18 statements 18 run 0 missing 0 excluded

```python
# -*- coding: utf-8 -*-
"""
Created on Thu Jan 14 13:44:00 2016
Updated Sep 20, 2020

The primary goal of this file is to demonstrate a simple python program to classify triangle

@author: jrr
@author: rk
@author: Srikanth Uppada
"""

def classify_triangle(side_a,side_b,side_c):
    """
    Your correct code goes here...  Fix the faulty logic below until the code passes all of
    you test cases.

    This function returns a string with the type of triangle from three integer values
    corresponding to the lengths of the three sides of the Triangle.

    return:
        If all three sides are equal, return 'Equilateral'
        If exactly one pair of sides are equal, return 'Isoceles'
        If no pair of  sides are equal, return 'Scalene'
        If not a valid triangle, then return 'NotATriangle'
        If the sum of any two sides equals the squate of the third side, then return 'Right'

      BEWARE: there may be a bug or two in this code
    """

    # verify that all 3 inputs are integers
    # Python's "isinstance(object,type) returns True if the object is of the specified type
    if not(isinstance(side_a,int) and isinstance(side_b,int) and isinstance(side_c,int)):
        return 'InvalidInput'

    # require that the input values be >= 0 and <= 200
    if side_a > 200 or side_b > 200 or side_c > 200:
        return 'InvalidInput'

    if side_a <= 0 or side_b <= 0 or side_c <= 0:
        return 'InvalidInput'

    # This information was not in the requirements spec but
    # is important for correctness
    # the sum of any two sides must be strictly less than the third side
    # of the specified shape is not a triangle
    if ((side_a + side_b) <= side_c) or\
       ((side_b + side_c) <= side_a) or\
       ((side_a + side_c) <= side_b):
        return 'NotATriangle'

    # now we know that we have a valid triangle
    if side_a == side_b and side_a == side_c:
        ret_string = 'Equilateral'
    elif ((side_a**2 + side_b**2) == side_c**2) or\
         ((side_a**2 + side_c**2) == side_b**2) or\
         ((side_b**2 + side_c**2) == side_a**2):
        ret_string =  'Right'
    elif (side_a != side_b) and  (side_b != side_c) and (side_c != side_a):
        ret_string =  'Scalene'
    else:
        ret_string =  'Isoceles'

    return ret_string
```

« index   *coverage.py v5.3, created at 2020-10-05 23:54 -0400*