Exercise 3

Finger Exercises due Aug 5, 2020 20:30 -03 *Completo*

Exercise 3

1/1 point (graded)
ESTIMATED TIME TO COMPLETE: 3 minutes

Consider the following function definition:

```
def maxOfThree(a,b,c) :
    """
    a, b, and c are numbers

    returns: the maximum of a, b, and c
    """
    if a > b:
        bigger = a

    else:
        bigger = b

    if c > bigger:
        bigger = c

    return bigger
```

Assume that `maxOfThree` is called with numbers as arguments.

Which of the following test suites would make a path-complete glass box test suite for `maxOfThree`?

Review: Glass Box Test Suites

Recall from the lecture that a path-complete glass box test suite would find test cases that go through every possible path in the code. This is different from black-box testing, because in black-box testing you only have the function specification. For glass-box testing, you actually know how the function you are testing is defined. Thus you can use this definition to figure out how many different paths through the code exist, and then pick a test suite based on that knowledge.

Undoubtably many - if not all - of the listed tests look like they would be pretty good for testing the function `maxOfThree`. However, we want you to think critically about the way `maxOfThree` is defined - including possible boundary cases - and pick a set of tests that adequately and fully tests all paths and boundary conditions. A good first step will be to look at the function definition and figure out what paths through the code exist. Then, look through the suggested test suites one test at a time and see if the suite tests every single path.

[✓ Test Suite A: `maxOfThree(2, -10, 100)`, `maxOfThree(7, 9, 10)`, `maxOfThree(6, 1, 5)`, `maxOfThree(0, 40, 20)`]

Test Suite B: `maxOfThree(10, 100, -20)`, `maxOfThree(99, 0, 20)`, `maxOfThree(1, 60, 300)`

Test Suite C: `maxOfThree(0, 0, 0)`, `maxOfThree(-3, -10, -1)`, `maxOfThree(10, 30, 100)`, `maxOfThree(0, -9, 11)`, `maxOfThree(-10, 0, 30)`

✓

Explanation:
Recall from the lecture that a path-complete glass box test suite would find test cases that go through every possible path in the code. In this case, that means finding all possibilities for the conditional tests `a > b` and `c > bigger`. So, we end up with four possible paths that correspond to Test Suite A:

- Case 1: `a > b` and `c > bigger` - this corresponds to test `maxOfThree(2, -10, 100)`.
- Case 2: `a > b` and `c <= bigger` - this corresponds to test `maxOfThree(6, 1, 5)`
- Case 3: `a <= b` and `c > bigger` - this corresponds to test `maxOfThree(7, 9, 10)`.
- Case 4: `a <= b` and `c <= bigger` - this corresponds to test `maxOfThree(0, 40, 20)`

Test Suite B seems to explicitly test each of `a`, `b`, and `c` being the max of the three numbers, but this does not go through all possible paths in the code.
Test Suite C seems to be a good sampling of the space of input numbers, but it does not go through all possible paths in the code.

Enviar

Answers are displayed within the problem

Exercise 3

Ocultar discussão

Topic: Lecture 7 / Exercise 3

Show all posts

por atividade recente

Don't Understand Why Are Test Suite B and C Wrong?
Hi, I ran the function maxOfThree for all the Test Suite A, B, & C and all gave me the right max numb... 1

Why is the answer the one it is?
I would have thought it would be C, because of the inclusion of a test where all integers are the sam... 3 new_

Test Suite C
I understand why Test Suite C is incorrect. However, wouldn't it be wise to have a test in which all int... 1 new_