Adam Zabloudil
CMSI 4072
3/19/2024

Homework 3

7.1
The issues in the code are that the ‘private’ is misspelled as ‘provate’, and the ‘if’ is capitalized as ‘If’.

7.2
If the person writing the comments misunderstands the logic being implemented, or if the comments are too vague and don’t provide enough context for someone who might be new to the code.

7.4
You could validate the inputs like checking that ‘a’ and ‘b; are not null, and consider edge cases like when either input is zero.

7.5
Yes, you should add error handling like checking for invalid inputs.

7.7
You would start by getting your keys and wallet and then entering your car. You then start the engine, and then use a GPS to find a route to the nearest supermarket. You then use the gas pedal and steering wheel to start driving to the supermarket. Once you arrive at the supermarket, you would pull into a spot at the parking lot and then put your car in park and then turn off your engine.
This assumes the person has a drivers license, has a functional car with gas, and has a supermarket somewhere near them.

8.1
You would write a test method that verifies the output of the ‘IsRelativelyPrime’ for pairs of numbers you know are either relatively prime or not. Then you can include edge cases like (0,1) and very large numbers.

8.3
This is black box testing because we do not know the exact code of the ‘IsRelativelyPrime’ method. White box testing could also be used if we know the internal logic and then could be used to test edge cases like the handling of zero.

8.5
In python:

```
def gcd(a, b):
	while b:
		a, b = b, a % b
	return a

def is_relatively_prime(a, b):
	return gcd(a, b) == 1



def test_is_relatively_prime():
	test_cases = [
		(21, 35, False),
		(8, 9, True),
		(0, 1, True),
		(1, 0, True),
		(-1, 0, True),
		(0, -1, True),
		(-3, -7, True),
		(14, 49, False),
	]

for a, b, expected in test_cases:
	result = is_relatively_prime(a, b)
	assert result == expected, f"Failed on {a}, {b}. Expected: {expected}, Got:
{result}"
	print("All tests passed")

test_is_relatively_prime()
```

This is the bug I got from testing the code

```
Traceback (most recent call last):
  File "/Users/adamzabloudil/Desktop/isRelative.py", line 30, in <module>
    test_is_relatively_prime()
  File "/Users/adamzabloudil/Desktop/isRelative.py", line 26, in test_is_relatively_prime
    assert result == expected, f"Failed on {a}, {b}. Expected: {expected}, Got: {result}"
           ^^^^^^^^^^^^^^^^^^
AssertionError: Failed on -1, 0. Expected: True, Got: False
```

8.9

Exhausting testing is a form of black box testing because it tests all possible inputs such as a brute force testing implementation. It's more focused on the behavior and result of the software rather than how the software is actually implemented.

8.11
We can use alice and bob who have an overlap of 2 bugs. Since alice found 5 bugs and bob found 4 bugs, the estimate is 5*4/2 = 10 bugs in total. You would then have to take into account the pairs including Carmen, which would be the bugs at large.

8.12
If the testers don’t find any bugs in common you would get undefined as an answer because you’d be dividing by zero. The number of bugs is at least as high as the sum of unique bugs found by each tester, even though it's probably an underestimate.