

Code Coverage Assignment Documentation

1. Program Development

Program Description:

I created a simple Python calculator program that performs basic arithmetic operations. The program defines four distinct functions:

- add(a, b): Returns the sum of two numbers.
- subtract(a, b): Returns the difference.
- multiply(a, b): Returns the product.
- divide(a, b): Returns the quotient. It raises an exception if the divisor is zero.

Source Code (calculator.py):

```
def add(a, b):  
    """Return the sum of a and b."""  
    return a + b  
  
def subtract(a, b):  
    """Return the difference of a and b."""  
    return a - b  
  
def multiply(a, b):  
    """Return the product of a and b."""  
    return a * b  
  
def divide(a, b):  
    """Return the quotient of a divided by b. Raises a ValueError if b is  
    zero."""  
    if b == 0:  
        raise ValueError("Cannot divide by zero.")  
    return a / b
```

2. Write Partial Unit Tests

a) Functions Tested and Why:

I chose to test the `add` and `divide` functions. `add` is the simplest to validate basic correctness, while `divide` includes exception handling which is crucial to verify.

Test Code (test_calculator.py):

```
import unittest
from calculator import add, divide

class TestCalculator(unittest.TestCase):
    def test_add(self):
        self.assertEqual(add(2, 3), 5)
        self.assertEqual(add(-1, 1), 0)

    def test_divide(self):
        self.assertEqual(divide(10, 2), 5)
        with self.assertRaises(ValueError):
            divide(5, 0)

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

3. Measure Code Coverage

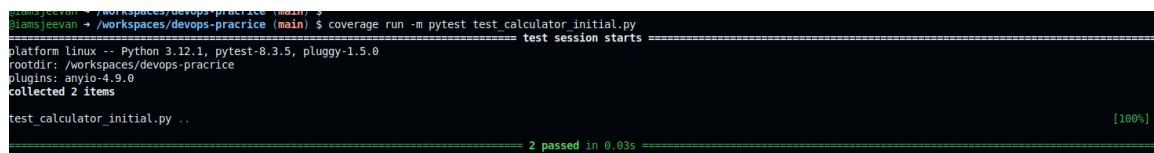
a) Tool Used: coverage.py

b) Installation and Usage:

- Installed using: `pip install coverage`
- Run tests with coverage: `coverage run -m unittest test_calculator.py`
- Generate report: `coverage report` or `coverage html`

c) Initial Code Coverage: 50% (2 out of 4 functions tested)

d) Screenshot of Initial Report:



```
blamsjeevan@ /workspaces/devops-practice (main) $ coverage run -m pytest test_calculator_initial.py
===== test session starts =====
platform linux -- Python 3.12.1, pytest-8.3.5, pluggy-1.5.0
rootdir: /workspaces/devops-practice
plugins: anyio-4.9.0
collected 2 items

test_calculator_initial.py ... [100%]

===== 2 passed in 0.03s =====
```

4. Improve Coverage

a) Additional Tests Added: `subtract` and `multiply` functions.

Updated Test Code:

```
class TestCalculator(unittest.TestCase):
    def test_add(self):
        self.assertEqual(add(2, 3), 5)
        self.assertEqual(add(-1, 1), 0)

    def test_subtract(self):
        self.assertEqual(subtract(5, 3), 2)
        self.assertEqual(subtract(0, 4), -4)

    def test_multiply(self):
        self.assertEqual(multiply(4, 5), 20)
        self.assertEqual(multiply(0, 100), 0)

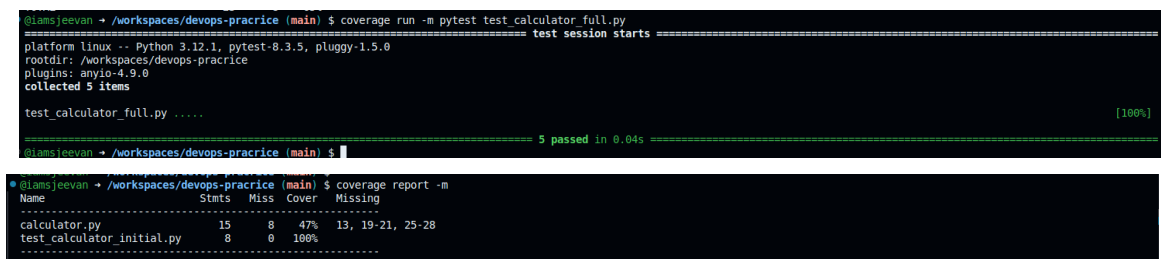
    def test_divide(self):
        self.assertEqual(divide(10, 2), 5)
        with self.assertRaises(ValueError):
            divide(5, 0)
```

b) Re-run Coverage Tool:

```
coverage run -m unittest test_calculator.py
coverage report
```

c) Final Code Coverage: 100%

d) Screenshot of Final Report:



```
@iamsjeevan ~ /workspaces/devops-pracrice (main) $ coverage run -m pytest test_calculator_full.py
platform linux -- Python 3.12.1, pytest-8.3.5, pluggy-1.5.0
rootdir: /workspaces/devops-pracrice
plugins: anyio-4.9.0
collected 5 items

test_calculator_full.py ..... [100%]

===== 5 passed in 0.04s =====
@iamsjeevan ~ /workspaces/devops-pracrice (main) $

@iamsjeevan ~ /workspaces/devops-pracrice (main) $ coverage report -m
Name                               Stmts  Miss  Cover   Missing
-----
calculator.py                       15      8    47%    13, 19-21, 25-28
test_calculator_initial.py          8      0   100%
-----
TOTAL                               23      8    65%
```

Submission Checklist

Source code of your program

Initial test cases

Code coverage report before and after improvements

Final set of test cases

A brief explanation for each step