NSEREKO SARAH A96409
JAMUGISA PETER PAUL A96402
SOUL SOLOMON SEKAMATTE A96920

STEP ONE

Writing unit tests

```
import unittest
from calculator import Calculator

class TestCalculator(unittest.TestCase):
    def test_addition(self):
        calc = Calculator()
        result = calc.add(2, 4)
        self.assertEqual(result, 6)

    def test_subtraction(self):
        calc = Calculator()
        result = calc.subtract(10, 8)
        self.assertEqual(result, 2)

    def test_multiplication(self):
        calc = Calculator()
        result = calc.multiply(3, 3)
        self.assertEqual(result,9)

    def test_division(self):
        calc = Calculator()
        result = calc.divide(10, 10)
        self.assertEqual(result, 1)

    def test_division_by_zero(self):
        calc = Calculator()
        with self.assertRaises(ValueError):
            calc.divide(10, 0)

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

STEP 2

Realizing there is no calculator class functionality which is why the tests are failing
Creating functionality for the calculator class

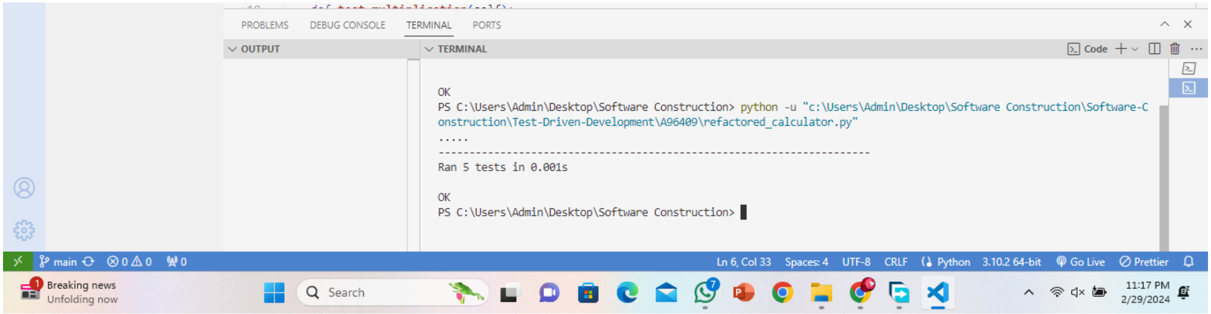
```
class Calculator:
    def add(self, x, y):
        return x + y

    def subtract(self, x, y):
        return x - y

    def multiply(self, x, y):
        return x * y

    def divide(self, x, y):
        if y == 0:
            raise ValueError("Cannot divide by zero")
        return x / y
```

STEP 3

Refactoring the code
Changes:
1.Creating the calculator instance once
2. Being more descriptive with our methods by adding doc strings

```
import unittest
from calculator import Calculator

class TestCalculator(unittest.TestCase):
    def setUp(self):
        self.calc = Calculator()

    def test_addition(self):
        """Test addition operation."""
        result = self.calc.add(2, 4)
        self.assertEqual(result, 6)

    def test_subtraction(self):
        """Test subtraction operation."""
        result = self.calc.subtract(10, 8)
        self.assertEqual(result, 2)

    def test_multiplication(self):
        """Test multiplication operation."""
        result = self.calc.multiply(3, 3)
        self.assertEqual(result, 9)

    def test_division(self):
        """Test division operation."""
        result = self.calc.divide(10, 10)
        self.assertEqual(result, 1)

    def test_division_by_zero(self):
        """Test division by zero."""
        with self.assertRaises(ValueError):
            self.calc.divide(10, 0)

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

Now all tests are passed



```
OK
PS C:\Users\Admin\Desktop\Software Construction> python -u "c:\Users\Admin\Desktop\Software Construction\Software-C
onstruction\Test-Driven-Development\A96409\refactored_calculator.py"
.....
----------------------------------------------------------------------
Ran 5 tests in 0.001s

OK
PS C:\Users\Admin\Desktop\Software Construction> 
```