МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Кафедра інформаційних систем та мереж

Лабораторна робота №6

з дисципліни Спеціалізовані мови програмування

на тему

Розробка та Unit тестування Python додатку

Виконав:
студент групи РІ-21сп
Владислав ДМИТРЕНКО

Львів – 2024

Мета: Створення юніт-тестів для додатка-калькулятора на основі класів

План роботи

Завдання 1: Напишіть юніт-тест, щоб перевірити, що операція додавання в вашому додатку-калькуляторі працює правильно. Надайте тестові випадки як для позитивних, так і для негативних чисел.

Завдання 2: Створіть юніт-тести для переконання, що операція віднімання працює правильно. Тестуйте різні сценарії, включаючи випадки з від'ємними результатами.

Завдання 3: Напишіть юніт-тести, щоб перевірити правильність операції множення в вашому калькуляторі. Включіть випадки з нулем, позитивними та від'ємними числами.

Завдання 4: Розробіть юніт-тести для підтвердження точності операції ділення. Тести повинні охоплювати ситуації, пов'язані з діленням на нуль та різними числовими значеннями.

Завдання 5: Створіть юніт-тести, щоб перевірити, як ваш додаток-калькулятор обробляє помилки. Включіть тести для ділення на нуль та інших потенційних сценаріїв помилок. Переконайтеся, що додаток відображає відповідні повідомлення про помилки.

Результати тестування:

```
PS C:\Users\Blxxd\Documents\GitHub\DegraCalc\tests> python .\test_calculator.py
test_addition_negative_numbers (__main__.TestCalculator.test_addition_negative_numbers) ... ok
✓Test 'test_memory_add (__main__.TestCalculator.test_memory_add)' passed successfully.
test_memory_clear (__main__.TestCalculator.test_memory_clear) ... ok
✓Test 'test_memory_clear (__main__.TestCalculator.test_memory_clear)' passed successfully.
test_memory_subtract (__main__.TestCalculator.test_memory_subtract) ... ok
✓Test 'test_memory_subtract (__main__.TestCalculator.test_memory_subtract)' passed successfully.
test_multiplication_negative_numbers (__main__.TestCalculator.test_multiplication_negative_numbers) ... ok
✓Test 'test_multiplication_negative_numbers (__main__.TestCalculator.test_multiplication_negative_numbers)' passed successfully.
test_multiplication_positive_numbers (__main__.TestCalculator.test_multiplication_positive_numbers) ... ok
✓Test 'test_multiplication_positive_numbers (__main__.TestCalculator.test_multiplication_positive_numbers)' passed successfully.
test_multiplication_with_zero (__main__.TestCalculator.test_multiplication_with_zero) ... ok
✓Test 'test_multiplication_with_zero (__main__.TestCalculator.test_multiplication_with_zero)' passed successfully.
test_subtraction_negative_result (__main__.TestCalculator.test_subtraction_negative_result) ... ok
✓Test 'test_subtraction_negative_result (__main__.TestCalculator.test_subtraction_negative_result)' passed successfully.
test_subtraction_positive_numbers (__main__.TestCalculator.test_subtraction_positive_numbers) ... ok
✓Test 'test_subtraction_positive_numbers (__main__.TestCalculator.test_subtraction_positive_numbers)' passed successfully.

----------------------------------------------------------------------
Ran 17 tests in 0.015s

OK
PS C:\Users\Blxxd\Documents\GitHub\DegraCalc\tests>
```

Рис. 1. Тестування проєкту

```
PS C:\Users\Blxxd\Documents\GitHub\DegraCalc\tests> coverage report
Name                                                                                    Stmts   Miss  Cover
-----------------------------------------------------------------------------------------------------------
C:\Users\Blxxd\Documents\GitHub\DegraCalc\classes\calculator.py                             57     11    81%
C:\Users\Blxxd\Documents\GitHub\DegraCalc\functions\ConvertNumberType.py                     9      3    67%
C:\Users\Blxxd\Documents\GitHub\DegraCalc\functions\ErrorHandler.py                         18     16    11%
test_calculator.py                                                                          70     24    66%
-----------------------------------------------------------------------------------------------------------
TOTAL                                                                                      154     54    65%
PS C:\Users\Blxxd\Documents\GitHub\DegraCalc\tests>
```

Рис. 2. Покриття тестами проєкту

Текст тест функцій

```python
class TestCalculator(unittest.TestCase):

    def setUp(self):
        self.calculator = Calculator()
        self.memory = Memory()

    def test_addition_positive_numbers(self):
        self.assertEqual(self.calculator.Add(3, 5), 8)

    def test_addition_negative_numbers(self):
        self.assertEqual(self.calculator.Add(-3, -5), -8)

    def test_addition_positive_and_negative(self):
        self.assertEqual(self.calculator.Add(-3, 5), 2)

    def test_subtraction_positive_numbers(self):
        self.assertEqual(self.calculator.Subtract(10, 5), 5)
```

```
    def test_subtraction_negative_result(self):
        self.assertEqual(self.calculator.Subtract(5, 10), -5)

    def test_multiplication_with_zero(self):
        self.assertEqual(self.calculator.Multiply(5, 0), 0)

    def test_multiplication_positive_numbers(self):
        self.assertEqual(self.calculator.Multiply(3, 5), 15)

    def test_multiplication_negative_numbers(self):
        self.assertEqual(self.calculator.Multiply(-3, -5), 15)

    def test_division_positive_numbers(self):
        self.assertEqual(self.calculator.Divide(10, 2), 5)

    def test_division_with_zero(self):
        self.assertEqual(self.calculator.Divide(10, 0), "Error: x/0")

    def test_division_negative_result(self):
        self.assertEqual(self.calculator.Divide(-10, 2), -5)

    def test_division_by_zero_error_handling(self):
        self.assertEqual(self.calculator.Divide(5, 0), "Error: x/0")

    def test_invalid_number_conversion(self):
        with self.assertRaises(ValueError):
            ConvertNumberType("invalid")

    def test_get_history(self):
        self.calculator.Add(1, 1)
        with open("History.txt", "r") as file:
            self.assertIn("1 + 1 = 2", file.read())

    def test_memory_add(self):
        self.memory.Add(10)
        self.assertEqual(self.memory.Read(), 10)

    def test_memory_subtract(self):
        self.memory.Add(10)
        self.memory.Subtract(5)
        self.assertEqual(self.memory.Read(), 5)

    def test_memory_clear(self):
        self.memory.Add(10)
        self.memory.Clear()
        self.assertEqual(self.memory.Read(), 0)

class CustomTextTestResult(unittest.TextTestResult):
    def addSuccess(self, test):
        super().addSuccess(test)
        print(f"✔ Test '{test}' passed successfully.")

    def addFailure(self, test, err):
        super().addFailure(test, err)
        print(f"✘ Test '{test}' failed. Error: {err}")
```

```
    def addError(self, test, err):
        super().addError(test, err)
        print(f"⚠ Test '{test}' encountered an error. Error: {err}")

class CustomTextTestRunner(unittest.TextTestRunner):
    def _makeResult(self):
        return CustomTextTestResult(self.stream, self.descriptions,
self.verbosity)

if __name__ == '__main__':
    runner = CustomTextTestRunner(verbosity=2)
    unittest.main(testRunner=runner)
```

Висновки: В ході виконання лабораторної роботи було створено юніт-тестування на основі лабораторної роботи 2, який перевіряє правильність основних арифметичних операцій.