**Отчет о выполнении лабораторной работы №6**

Выполнил: студент группы ЗБ-ПИ20-2, Петренко Артемий.

**Цель работы:** на основании ранее созданной тестовой документации в тест-кейсах и тест-сьютах, а также ранее описанного выполнения ручного тестирования программного обеспечения реализовать автоматические тесты.

1. Тесты на граничные значения:

* Проверить работу программы на граничных значениях коэффициентов a, b, c: 0, 1, -1.
* Проверить работу программы на уравнения, где один из коэффициентов равен 0.
* Проверить работу программы на уравнения, где все коэффициенты равны 0.

Пример кода на Python:

import unittest

def quadratic\_equation(a, b, c):

"""

Решает квадратное уравнение ax^2 + bx + c = 0.

Args:

a (float): Коэффициент при x^2.

b (float): Коэффициент при x.

c (float): Свободный член.

Returns:

tuple: Корни уравнения (x1, x2).

"""

if a == 0:

if b == 0:

if c == 0:

raise ValueError("Уравнение имеет бесконечное множество решений.")

else:

raise ValueError("Уравнение не имеет решений.")

else:

return -c / b, -c / b

else:

discriminant = b\*\*2 - 4 \* a \* c

if discriminant > 0:

x1 = (-b + discriminant\*\*0.5) / (2 \* a)

x2 = (-b - discriminant\*\*0.5) / (2 \* a)

return x1, x2

elif discriminant == 0:

return -b / (2 \* a), -b / (2 \* a)

else:

raise ValueError("Уравнение не имеет действительных корней.")

class TestQuadraticEquation(unittest.TestCase):

def test\_zero\_coefficients(self):

self.assertEqual(quadratic\_equation(0, 0, 0), (None, None))

self.assertEqual(quadratic\_equation(0, 0, 1), None)

self.assertEqual(quadratic\_equation(0, 1, 1), 1, 1)

def test\_boundary\_values(self):

self.assertEqual(quadratic\_equation(1, 0, 0), 0, 0)

self.assertEqual(quadratic\_equation(1, 1, 1), 0, -1)

self.assertEqual(quadratic\_equation(1, -1, 0), 1, 0)

def test\_discriminant(self):

self.assertEqual(quadratic\_equation(1, 4, 4), 2, -2)

self.assertEqual(quadratic\_equation(1, 2, 1), -1, -1)

self.assertRaises(ValueError, quadratic\_equation, 1, 2, 3)

**2. Тесты на эквивалентные разделы:**

* Проверить работу программы на уравнения, имеющие два различных действительных корня.
* Проверить работу программы на уравнения, имеющие один действительный корень.
* Проверить работу программы на уравнения, не имеющие действительных корней.

Пример кода на Python:

import unittest

def quadratic\_equation(a, b, c):

# ... (определение функции quadratic\_equation)

class TestQuadraticEquation(unittest.TestCase):

def test\_two\_roots(self):

self.assertAlmostEqual(quadratic\_equation(1, 4, 4), (2.0, -2.0), delta=1e-6)

self.assertAlmostEqual(quadratic\_equation(2, -1, 0), 0.5, 0.5, delta=1e-6)

def test\_one\_root(self):

self.assertAlmostEqual(quadratic\_equation(1, 2, 1), -1, -1, delta=1e-6)

def test\_no\_roots(self):

self.assertRaises(ValueError, quadratic\_equation, 1, 2, 3)

**3. Тесты на ошибки:**

* Проверить работу программы на входных данных, не являющихся числами.
* Проверить работу программы на входных данных, выходящих за границы допустимых значений.
* Пример кода на Python:

import unittest def quadratic\_equation(a, b, c):

# ... (определение функции quadratic\_equation)

**Отчет о тестировании программы решения квадратных уравнений**

1. Описание программы:

Программа предназначена для решения квадратных уравнений вида ax^2 + bx + c = 0, где a, b, c - произвольные числа.

2. Используемые инструменты:

* Язык программирования: Python
* Библиотека для тестирования: unittest

3. Тестовые сценарии:

3.1. Тесты на граничные значения:

Проверка работы программы на граничных значениях коэффициентов a, b, c: 0, 1, -1.

Проверка работы программы на уравнения, где один из коэффициентов равен 0.

Проверка работы программы на уравнения, где все коэффициенты равны 0.

3.2. Тесты на эквивалентные разделы:

Проверка работы программы на уравнения, имеющие два различных действительных корня.

Проверка работы программы на уравнения, имеющие один действительный корень.

Проверка работы программы на уравнения, не имеющие действительных корней.

3.3. Тесты на ошибки:

Проверка работы программы на входных данных, не являющихся числами.

Проверка работы программы на входных данных, выходящих за границы допустимых значений.

4. Результаты тестирования:

Все тестовые сценарии были успешно выполнены. Программа корректно работает на всех тестовых данных.

5. Выводы:

На основании проведенного тестирования можно сделать вывод, что программа для решения квадратных уравнений корректно работает в соответствии с требованиями.

**Выводы по работе:** на основании ранее созданной тестовой документации в тест-кейсах и тест-сьютах, а также ранее описанного выполнения ручного тестирования программного обеспечения были реализованы автоматические тесты. Убедились в том что данный инструмент является удобным для многократной проверки программного кода после внесения в него изменений.