Отчет по лабораторной работе 1 по программированию

Задание 3,1

Условие:

Создайте простую программу калькулятор, которая позволяет из функции main() ввести два числа и тип арифметической операции, а потом вычисляет результат. Свой код опубликуйте на KttSs reSlit. cRm и предоставьте ссылку в ответах на лабораторную работу в Moodle в документе-отчёте. Реализацию арифметических действий и вычисление результата с его возвратом сделайте в отдельной функции calculate(...). Протестируйте свой калькулятор с помощью вызова нескольких своих простых функций test_*() с ключевым словом assert внутри. Обязательно напишите хорошую документацию к своему коду.

Код программы:

```
def calculate(num1, num2, operator):
    if operator == "+":
        return num1 + num2
    elif operator == "-":
        return num1 - num2
    elif operator == "*":
        return num1 * num2
    elif operator == "/":
        if num2 == 0:
            return "Ошибочка!"
        else:
            return num1 / num2
    else:
        return "Ошибочка!"
def test_1():
    assert calculate(2, 3, "+") == 5,  "ошибочка"
def test_2():
    assert calculate(5, 2, "-") == 3,  "ошибочка"
def test_3():
    assert calculate(4, 5, "*") == 20,  "ошибочка"
def test_4():
    assert calculate(10, 3, "/") == 5.0,  "ошибочка"
def main():
    num1 = float(input("Введите первое число: "))
    num2 = float(input("Введите второе число: "))
    operator = input("Введите оператор: ")

    res = calculate(num1, num2, operator)
    print(res)
main()
test_1()
test_2()
test_3()
test_4()
```

Результат:

```
Введите первое число: 10
Введите второе число: 10
Введите оператор: *
100.0
Traceback (most recent call last):
  File "C:\Users\Michael\PycharmProjects\pythonProject\venv\ЛР 1.1.py", line 34, in <module>
    test_4()
  File "C:\Users\Michael\PycharmProjects\pythonProject\venv\ЛР 1.1.py", line 22, in test_4
    assert calculate(10, 3, "/") == 5.0,  "ошибочка"
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
AssertionError: ошибочка
```

Описание кода:

Этот код реализует простой калькулятор, который может выполнять четыре основные арифметические операции: сложение, вычитание, умножение и деление.

1. def calculate(num1, num2, operator): - Функция calculate() принимает три параметра: два числа (num1 и num2) и оператор (operator). Она проверяет, какая именно операция должна быть выполнена, и возвращает результат.

- Если operator равен "+", функция возвращает сумму num1 и num2.
- Если operator равен "-", функция возвращает разность num1 и num2.
- Если operator равен "*", функция возвращает произведение num1 и num2.
- Если operator равен "/", функция проверяет, не является ли num2 равным 0. Если так, она возвращает строку "Ошибочка!". В противном случае, она возвращает частное num1 и num2.

2. def test_1(), test_2(), test_3(), test_4(): - Эти функции представляют тесты, которые проверяют корректность работы функции calculate(). Каждая из них вызывает calculate() с определенными входными данными и сравнивает результат с ожидаемым значением. Если результат не соответствует ожидаемому, тест выводит сообщение "ошибочка".

3. def main(): - Эта функция запрашивает у пользователя два числа и оператор, вызывает функцию calculate() с этими параметрами и выводит результат.

Задание 3,2

Условие: Реализуйте программно классическую простую игру "угадай число" (guess number) с помощью алгоритма медленного перебора (инкремента) по одному числа, либо с помощью алгоритма бинарного поиска. Алгоритм принимает на вход само число, которое он должен угадать, интервал значений в котором оно загадано и в цикле делает угадывания тем или иным выбранным вами способом. После угадывания из функции алгоритма возвращается угаданное число и число угадываний/сравнений, которые пришлось проделать. Обязательно напишите хорошую документацию к своему коду.

Код программы:

```
def main(number, low, high):
    n = low
    res = 0
    while n <= high:
        res += 1
        if n == number:
            return n, res
        n += 1
    return None, res

number = int(input("Введите число: "))
num, res = main(number, 1, 100)
if num is not None:
    print(f"Угадано число: {num} за {res} попыток.")
else:
    print("Число не найдено.")

def test1():
    assert main(50, 1, 100) == (50, 50),  "все верно"
def test2():
    assert main(101, 1, 100) == (None, 101), "Ошибочка"
test1()
test2()
```

Результат:

```
Введите число: 75
Угадано число: 75 за 75 попыток.
Traceback (most recent call last):
  File "C:\Users\Michael\PycharmProjects\pythonProject\ЛР 1,2.py", line 23, in <module>
    test1()
  File "C:\Users\Michael\PycharmProjects\pythonProject\ЛР 1,2.py", line 19, in test1
    assert main(50, 1, 100) == (51, 50),  "ошибочка"
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^
AssertionError: ошибочка
```

Описание кода:

1. def main(number, low, high): - Функция main() принимает три аргумента: number (число, которое необходимо найти), low (нижняя граница диапазона поиска) и high (верхняя граница диапазона поиска).

2. Внутри функции main() происходит поиск числа number в указанном диапазоне. Счетчик res увеличивается на 1 с каждой итерацией цикла, пока число не будет найдено или диапазон поиска не будет исчерпан.

3. Если число number найдено, функция возвращает его и количество итераций res. Если число не найдено, функция возвращает None и количество итераций res.

4. В основной части кода пользователь вводит число, которое необходимо найти, и вызывается функция main() с этим числом в качестве аргумента.

5. Результат работы функции main() (найденное число и количество попыток) сохраняется в переменные num и res.

6. Если число найдено (num не равно None), выводится соответствующее сообщение. Если число не найдено, выводится сообщение об этом.

7. Далее идут два теста: test1() и test2(). Тест test1() проверяет, что функция main() правильно находит число 50 в диапазоне 1-100 за 50 попыток. Тест test2() проверяет, что функция main() правильно возвращает None и количество попыток, если число 101 не найдено в диапазоне 1-100.