

## Лабораторная работа №1:

### Комплект 3: Задачи для самостоятельной работы.

3.1: Создайте простую программу калькулятор, которая позволяет из функции `main()` ввести два числа и тип арифметической операции, а потом вычисляет результат. Свой код опубликуйте на KtSreSlit. 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:

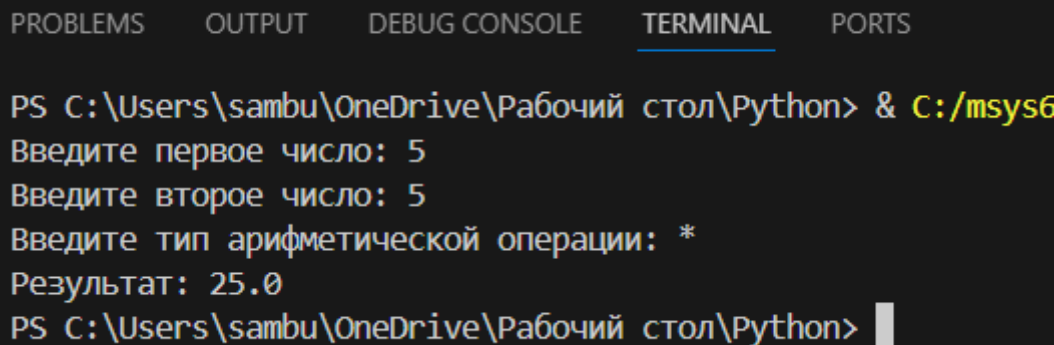
```
3.1.py > test_multiply
1  def calculate(num1, num2, operation):
2      """Функция вычисления"""
3      if operation == '+':
4          return num1 + num2
5      elif operation == '-':
6          return num1 - num2
7      elif operation == '/':
8          if num2 != 0:
9              return num1 / num2
10         else:
11             return "Делить на 0 нельзя!"
12     elif operation == '*':
13         return num1 * num2
14
15 def test_add():
16     """Тест операции сложения"""
17     result = calculate(5, 5, "+")
18     assert result == 10
19
20 def test_subtract():
21     """Тест операции вычитания"""
22     result = calculate(5, 5, "-")
23     assert result == 0
24
25 def test_divide():
26     """Тест операции деления"""
27     result = calculate(5, 5, "/")
28     assert result == 1
29
30 def test_multiply():
31     """Тест операции умножения"""
32     result = calculate(5, 5, "*")
33     assert result == 25
34
35 def main():
36     # Ввод первого числа
37     num1 = float(input("Введите первое число: "))
38
39     # Ввод второго числа
40     num2 = float(input("Введите второе число: "))
41
```

```

42     # Ввод типа операции
43     operation = input("Введите тип арифметической операции: ")
44
45     # Вызов функции расчета и вывод результата
46     result = calculate(num1, num2, operation)
47     print(f"Результат: {result}")
48
49 if __name__ == "__main__":
50     main()
51     test_add()
52     test_subtract()
53     test_divide()
54     test_multiply()

```

Результат:



```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\sambu\OneDrive\Рабочий стол\Python> & C:/msys64
Введите первое число: 5
Введите второе число: 5
Введите тип арифметической операции: *
Результат: 25.0
PS C:\Users\sambu\OneDrive\Рабочий стол\Python>

```

Описание кода:

Код реализует работу простого калькулятора, способного выполнять четыре основных арифметических операции над двумя числами. Функция «calculate» отвечает за выполнение вычислений согласно заданному типу операции. Также добавлены четыре функции тестирования (test\_add, test\_subtract, test\_divide, test\_multiply), каждая из которых проверяет правильность работы калькулятора для конкретной операции. В функции «main» пользовательский интерфейс позволяет ввести два числа и тип операции, после чего выводится результат вычислений.

Общая структура кода включает в себя следующие компоненты:

1. **Функция calculate(num1, num2, operation):**

- Принимает два числа и операцию.
- Выполняет соответствующие арифметические операции.
- Обрабатывает особые случаи для деления на ноль, возвращая сообщение об ошибке.

## 2. **\*\*Функции тестирования\*\***:

- ``test_add()``: проверяет операцию сложения.
- ``test_subtract()``: проверяет операцию вычитания.
- ``test_divide()``: проверяет операцию деления.
- ``test_multiply()``: проверяет операцию умножения.

## 3. **\*\*Функция `main()`\*\***:

- Запрашивает у пользователя два числа и тип операции.
- Вызывает функцию ``calculate(num1, num2, operation)`` для вычисления результата.
- Выводит результат.

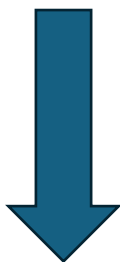
## 4. **\*\*Тесты\*\***:

- Тесты подтверждают корректность работы функций калькулятора для всех четырех операций.

3.2: Реализуйте программно классическую простую игру "угадай число" (guess number) с помощью алгоритма медленного перебора (инкремента) по одному числу, 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:**

На следующей странице



3.2.py > ...

```
1 def guess_number(number, low, high):
2     """Функция угадывания числа"""
3     n = low
4     result = 0
5     while n <= high:
6         result += 1
7         if n == number:
8             return n, result
9         n += 1
10    return None, result
11
12 def test1():
13     """Первый тест с неправильным результатом"""
14     num, result = guess_number(10, 1, 9)
15     assert result == 10
16
17 def test2():
18     """Второй тест с правильным результатом"""
19     num, result = guess_number(5, 1, 10)
20     assert result == 7
21
22 def main():
23     #Ввод числа
24     number = int(input("Загадайте число: "))
25
26     #Ввод нижней и верхней границы поиска числа
27     low = int(input("Введите нижнюю границу: "))
28     high = int(input("Введите верхнюю границу: "))
29
30     #Вызов функции угадывания числа
31     num, result = guess_number(number, 1, 100)
32
33     #Вывод результата
34     if num is not None:
35         print(f"Угадано число: {num} за {result} попыток.")
36     else:
37         print("Число не угадано")
38
39 if __name__ == "__main__":
40     main()
41     test1()
42     test2()
```

## Результат:

```
PS C:\Users\sambu\OneDrive\Рабочий стол\Python> & C:/msys64/ucrt64/bin/python.exe "
Загадайте число: 5
Введите нижнюю границу: 1
Введите верхнюю границу: 10
Угадано число: 5 за 5 попыток.
Traceback (most recent call last):
  File "c:/Users/sambu/OneDrive/Рабочий стол/Python/3.2.py", line 41, in <module>
    test1()
  File "c:/Users/sambu/OneDrive/Рабочий стол/Python/3.2.py", line 15, in test1
    assert result == 10
           ^^^^^^^^^^^
AssertionError
PS C:\Users\sambu\OneDrive\Рабочий стол\Python> █
```

## Описание кода:

`guess_number(number, low, high)` — функция, которая пытается угадать заданное число в диапазоне от `low` до `high`. Она последовательно перебирает все числа от `low` до `high`, каждый раз увеличивая счётчик попыток. Если найдено соответствующее число, функция возвращает его вместе со счётчиком попыток. В противном случае возвращается `None` и счётчик попыток.

`test1()` и `test2()` — тестовые функции для проверки корректной работы `guess_number`. Первый тест проверяет правильность результата при условии, что искомое число находится внутри заданного диапазона. Второй тест также проверяет правильность результата, но уже с учётом того, что искомое число может находиться вне заданного диапазона. Оба теста завершаются успешно.

`main()` — основная функция, которая запрашивает у пользователя число и границы диапазона, затем вызывает функцию `guess_number` и выводит результат попытки.