

# Программирование. Язык Python.

## Введение.

### Лабораторная работа № 1. Задачи.

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. 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:

<https://replit.com/@sashavyatk/Programming-2-course#main.py>

```
def calculate(num1, num2, action):  
    """  
    Возвращает сумму, разность, произведение или частное двух чисел  
    """  
  
    if action == "+":  
        return num1 + num2  
  
    elif action == "-":  
        return num1 - num2  
  
    elif action == "*":  
        return num1 * num2  
  
    elif action == "/":  
        if num2 == 0:  
            print("Division by zero isn't possible!")  
        else:
```

```
return numb1 / numb2
```

```
def test_1_calculate():
```

```
    """
```

```
    Тестирование простого случая вычитания 3 из 5
```

```
    """
```

```
    assert calculate(3, 5, "-") == -2
```

```
def test_2_calculate():
```

```
    """
```

```
    Тестирование простого случая сложения 3 и 5
```

```
    """
```

```
    assert calculate(3, 5, "+") == 5
```

```
def test_3_calculate():
```

```
    """
```

```
    Тестирование простого случая деления 3 на 5
```

```
    """
```

```
    assert calculate(3, 5, ":") == 0.65
```

```
def test_4_calculate():
```

```
    """
```

```
    Тестирование простого случая умножения 3 на 5
```

```
    """
```

```
assert calculate(3, 5, "*") == 15
```

```
def main():
```

```
    """
```

Принимает на вход два числа и тип арифметической операции и вычисляет результат с помощью функции calculate

```
    """
```

```
    print("Enter the first number")
```

```
    a = int(input())
```

```
    print("Enter the second number")
```

```
    b = int(input())
```

```
    print("Enter the action")
```

```
    act = str(input())
```

```
    print(calculate(a, b, act))
```

```
    if act == "+":
```

```
        test_2_calculate()
```

```
    elif act == "-":
```

```
        test_1_calculate()
```

```
    elif act == "*":
```

```
        test_4_calculate()
```

```
    elif act == ":":
```

```
        test_3_calculate()
```

```
main()
```

```
main.py x LR1_3_2.py +
main.py > ...
1 def calculate(numb1, numb2, action):
2     """
3     Возвращает сумму, разность, произведение или частое двух чисел
4     """
5     if action == "+":
6         return numb1 + numb2
7     elif action == "-":
8         return numb1 - numb2
9     elif action == "*":
10        return numb1 * numb2
11    elif action == ":":
12        if numb2 == 0:
13            print("Division by zero isn't possible!")
14        else:
15            return numb1 / numb2
16
17
18 def test_1_calculate():
19     """
20     Тестирование простого случая вычитания 3 из 5
21     """
22     assert calculate(3, 5, "-") == -2 #верно
23
24 def test_2_calculate():
25     """
26     Тестирование простого случая сложения 3 и 5
27     """
28     assert calculate(3, 5, "+") == 5 #неверно
```

```
main.py x LR1_3_2.py +
main.py > ...
28     assert calculate(3, 5, "+") == 5 #неверно
29
30 def test_3_calculate():
31     """
32     Тестирование простого случая деления 3 на 5
33     """
34     assert calculate(3, 5, ":") == 0.65 #неверно
35
36 def test_4_calculate():
37     """
38     Тестирование простого случая умножения 3 на 5
39     """
40     assert calculate(3, 5, "*") == 15 #верно
41
42
43 def main():
44     """
45     Принимает на вход два числа и тип арифметической операции и
46     вычисляет результат с помощью функции calculate
47     """
48     print("Enter the first number")
49     a = int(input())
50     print("Enter the second number")
51     b = int(input())
52     print("Enter the action")
53     act = str(input())
54     print(calculate(a, b, act))
55     if act == "+":
```

```
main.py x LR1_3_2.py +
main.py > ...
54     if act == "+":
55         test_2_calculate()
56     elif act == "-":
57         test_1_calculate()
58     elif act == "*":
59         test_4_calculate()
60     elif act == ":":
61         test_3_calculate()
62 main()
63
```

## Результаты работы программы:

```
Run
Enter the first number
1357
Enter the second number
7943
Enter the action
*
10778651
```

```
Enter the first number
2574
Enter the second number
40
Enter the action
:
64.35
Traceback (most recent call last):
  File "/home/runner/Programming-2-course/main.py", line 62,
in <module>
    main()
  File "/home/runner/Programming-2-course/main.py", line 61,
in main
    test_3_calculate()
  File "/home/runner/Programming-2-course/main.py", line 34,
in test_3_calculate
    assert calculate(3, 5, ":") == 0.65
           ~~~~~^~~~~~
AssertionError
```

```
Enter the first number
-394
Enter the second number
28
Enter the action
-
-422
```

```
Enter the first number
3
Enter the second number
5
Enter the action
+
8
Traceback (most recent call last):
  File "/home/runner/Programming-2-course/main.py", line 62,
in <module>
    main()
  File "/home/runner/Programming-2-course/main.py", line 55,
in main
    test_2_calculate()
  File "/home/runner/Programming-2-course/main.py", line 28,
in test_2_calculate
    assert calculate(3, 5, "+") == 5 #неверно
           ~~~~~^~~~~~
AssertionError
```

```
Enter the first number
3480
Enter the second number
0
Enter the action
:
Division by zero isn't possible!
None
Traceback (most recent call last):
  File "/home/runner/Programming-2-course/main.py", line 62,
in <module>
    main()
  File "/home/runner/Programming-2-course/main.py", line 61,
in main
    test_3_calculate()
  File "/home/runner/Programming-2-course/main.py", line 34,
in test_3_calculate
    assert calculate(3, 5, ":") == 0.65 #неверно
    ~~~~~^~~~~~
AssertionError
```

### 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:

```
def guess_number(number, low, high):
```

• • • • •

Угадывает и возвращает загаданное число с помощью алгоритма бинарного поиска

• • • • •

```
count = 0
```

```
while high > low:
```

```
mid = (low + high) // 2
```

```
if mid > number:
```

high = mid

```
count += 1
```

```
elif mid < number:
```

```
    low = mid
```

```
    count += 1
```

```
else:
```

```
    count += 1
```

```
    return mid, count
```

```
print('Enter number to search')
```

```
numb = int(input())
```

```
print('Enter a lower boarder')
```

```
first = int(input())
```

```
print('Enter a higher boarder')
```

```
last = int(input())
```

```
print("I guessed the number! Your number and number of comparisons are",  
guess_number(numb, first, last))
```

```
def test_guess_number():
```

```
    """
```

```
    Тестирование простого случая угадывания цифры 2 на интервале от 1 до 5
```

```
    """
```

```
    assert guess_number(2, 1, 5) == (2, 2)
```

```
test_guess_number()
```

```
main.py LR1_3_2.py × +
LR1_3_2.py > ... Format
1 def guess_number(number, low, high):
2     """
3     Угадывает и возвращает загаданное число с помощью алгоритма бинарного поиска
4     """
5     count = 0
6     while high > low:
7         mid = (low + high) // 2
8         if mid > number:
9             high = mid
10            count += 1
11        elif mid < number:
12            low = mid
13            count += 1
14        else:
15            count += 1
16            return mid, count
17
18
19 print('Enter number to search')
20 numb = int(input())
21 print('Enter a lower boarder')
22 first = int(input())
23 print('Enter a higher boarder')
24 last = int(input())
25 print("I guessed the number! Your number and number of comparisons are",
26       guess_number(numb, first, last))
26
main.py LR1_3_2.py × +
LR1_3_2.py > ? guess_number > ... Format
26
27
28 def test_guess_number():
29     """
30     Тестирование простого случая угадывания цифры 2 на интервале от 1 до 5
31     """
32     assert guess_number(2, 1, 5) == (2, 2)
33
34
35 test_guess_number()
36
```

## Результаты работы программы:

```
~/Programming-2-course$ python3 LR1_3_2.py
Enter number to search
13
Enter a lower boarder
1
Enter a higher boarder
100
I guessed the number! Your number and number of comparisons are (13, 3)
```

```
~/Programming-2-course$ python3 LR1_3_2.py
Enter number to search
582
Enter a lower boarder
-294
Enter a higher boarder
2984
I guessed the number! Your number and number of comparisons are (582, 9)
Traceback (most recent call last):
  File "/home/runner/Programming-2-course/LR1_3_2.py", line 35, in <module>
    test_guess_number()
  File "/home/runner/Programming-2-course/LR1_3_2.py", line 32, in test_guess_number
    assert guess_number(2, 1, 5) == (2, 3) #неверно
AssertionError
~/Programming-2-course$
```