## МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования «Северо-Кавказский федеральный университет»

Кафедра инфокоммуникаций

Отчет по лабораторной работе № 4.2 «Перегрузка операторов в языке Python»

по дисциплине «Объектно-ориентированное программирование»

Выполнил студент групп	ы ИЕ	ЗТ-б-6	o-20-	1
Дыбов Д.В. « »	_20_	_Γ.		
Подпись студента		_		
Работа защищена « »			_20_	_г.
Проверил Воронкин Р.А.				
	(подп	ись)		

**Цель работы:** приобретение навыков по перегрузке операторов при написании программ с помощью языка программирования Python версии 3.х.

#### Ход Работы

- 1. Создал новый репозиторий для лабораторной работы №4.2;
- 2. Клонировал созданный репозиторий на компьютер;
- 3. Создал новый РуСharm проект в папке репозитория;
- 4. Проработал пример:

```
r1 = 3 / 4
r2 = 5 / 6
r1 + r2 = 19 / 12
r1 - r2 = -1 / 4
r1 * r2 = 5 / 8
r1 / r2 = 9 / 10
r1 == r2: False
r1 != r2: True
r1 > r2: False
r1 < r2: True
r1 >= r2: False
r1 <= r2: True
```

Рисунок 1 – Результат выполнения примера

- 5. Выполнил первое индивидуальное задание;
- 6. Проверил задание на работоспособность:

```
Целочисленное деление: (2, 8)

Process finished with exit code 0
```

Рисунок 2 – Результат выполнения первого индивидуального задания

- 7. Выполнил второе индивидуальное задание;
- 8. Проверил задание на работоспособность:

```
n1 + n2 = 4d1x0
n1 - n2 = 01x0
n1 * n2 = 4a5dx0
n1 < n2 = False
n1 > n2 = True
n1 == n2 = False

Process finished with exit code 0
```

Рисунок 3 — Результат выполнения второго индивидуального задания

9. Проверил пример на наличие ошибок:

# PythonChecker Makes Your Code Great Again

```
100%
Guido
Lines: 124
Hints: 0
```

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

class Rational:

   def __init__(self, a=0, b=1):
        a = int(a)
        b = int(b)
```

Рисунок 4 – Проверка примера на наличие ошибок

10. Проверил первое индивидуальное задание на наличие ошибок:

# PythonChecker Makes Your Code Great Again

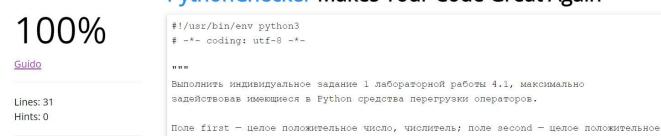


Рисунок 5 – Проверка примера на наличие ошибок

11. Проверил первое индивидуальное задание на наличие ошибок:

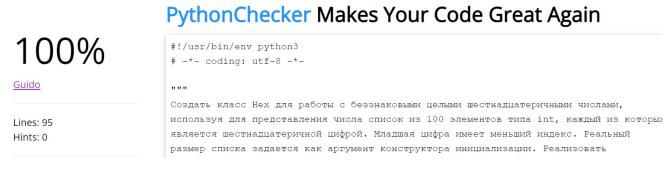


Рисунок 6 – Проверка примера на наличие ошибок

### Ответы на контрольные вопросы

1. Какие средства существуют в Python для перегрузки операций? Перегрузка осуществляется при помощи специальных методов. Методы группируются по следующим категориям:

- методы для всех видов операций;
- методы перегрузки операторов работы с коллекциями;
- методы для числовых операций в двоичной форме;
- методы для других операций над числами;
- методы для операций с дескрипторами;
- методы для операций, используемых с диспетчерами контекста.

2. Какие существуют методы для перегрузки арифметических операций и операций отношения в языке Python?

```
_{add}_{self}, other) - сложение. x + y вызывает x. add (y).
     _{\text{sub}} (self, other) - вычитание (x - y).
     _{\text{mul}} (self, other) - умножение (x * y).
     \_truediv\_(self, other) - деление (x / y).
     __floordiv__(self, other) - целочисленное деление (x // y).
     _{\rm mod} (self, other) - остаток от деления (х % у).
      \_divmod\_(self, other) - частное и остаток (divmod(x, y)).
     __pow__(self, other[, modulo]) - возведение в степень ( x ** y , pow(x,
     y[,modulo])).
     __lshift__(self, other) - битовый сдвиг влево (x << y).
     _rshift__(self, other) - битовый сдвиг вправо (x \gg y).
     and (self, other) - битовое И (x & y).
     __xor__(self, other) - битовое ИСКЛЮЧАЮЩЕЕ ИЛИ (x ^ y).
     __radd__(self, other),
     __rsub__(self, other) ,
     __rmul__(self, other),
     __rtruediv__(self, other),
     __rmod__(self, other),
     __rdivmod__(self, other),
      __rpow__(self, other),
     __rlshift__(self, other) ,
     __rrshift__(self, other) ,
     __rand__(self, other),
     __rxor__(self, other),
     __ror__(self, other) - делают то же самое, что и арифметические операторы,
перечисленные выше, но для аргументов, находящихся справа, и только в
случае, если для левого операнда не определён соответствующий метод.
     _{idd}(self, other) - += .
```

```
\_isub\_(self, other) - -= .
      _{inv} imul_ (self, other) - *= .
      __itruediv__(self, other) - /= .
      __ifloordiv__(self, other) - //= .
      __imod__(self, other) - %= .
      _{ipow}(self, other[, modulo]) - **= .
      __ilshift__(self, other) - <<= .
      __irshift__(self, other) - >>= .
      \_iand\_(self, other) - &= .
      \underline{\text{ixor}}_{\text{self}}(\text{self}, \text{other}) - ^= .
      _{\rm ior} (self, other) - |= .
      3. В каких случаях будут вызваны следующие методы: add , iadd
      и __radd___?
      1) add -a+b
      2) iadd -a += b
      3) __radd__ - Если не получилось вызвать метод add
      4. Для каких целей предназначен метод пеж ? Чем он отличается от
метода init ?
      Метод new используется, когда нужно управлять процессом создания
нового экземпляра, а init – когда контролируется его инициализация.
      5. Чем отличаются методы str и repr ?
      str должен возвращать строковый объект, тогда как repr может
возвращать любое выражение в Python
```

**Вывод:** в ходе занятия были приобретены навыки по перегрузке операторов при написании программ с помощью языка программирования Рython версии 3.х.