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 группы ИВТ-б-о-20-1

Плотников Д. В. « »_____20__г.

Подпись студента_____

Работа защищена « »_____20__г.

Проверил Воронкин Р. А. _____

(подпись)

Ставрополь 2022

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

Ход работы

1. Создал новый собственный репозиторий. Ссылка на репозиторий: https://github.com/Dmitry-15/4.2_laba.

2. Ознакомившись с теорией выполнил для начала пример.

```
C:\Users\Plotnikov\PycharmProjects\pythonProject22\venv\Scripts\python.exe C:/Users/Plotnikov/PycharmProjects/pythonProject22/primer.py
r1 = 3 / 4
r2 = 5 / 6
r1 + r2 = 19 / 12
r1 - r2 = -1 / 12
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 – Выполнение примера

3. Приступил к выполнению индивидуальных заданий.

Индивидуальные задания

Вариант 18

Задание 1

1. Условие задания: выполнить индивидуальное задание 1 лабораторной работы 4.1, максимально задействовав имеющиеся в Python средства перегрузки операторов.

```
C:\Users\Plotnikov\PycharmProjects\pythonProject22\venv\Scripts\python.exe
Начисленная сумма за 2 месяца: 62277.85393899204

Process finished with exit code 0
```

Рисунок 2 – Выполнение первого индивид. задания

Задание 2

1. Условие задания: создать класс Octal для работы с беззнаковыми целыми восьмеричными числами, используя для представления числа список из 100 элементов типа int, каждый элемент которого является восьмеричной цифрой. Младшая цифра имеет меньший индекс (единицы — в нулевом элементе списка). Реальный размер списка задается как аргумент

конструктора инициализации. Реализовать арифметические операции, аналогичные встроенным для целых и операции сравнения.

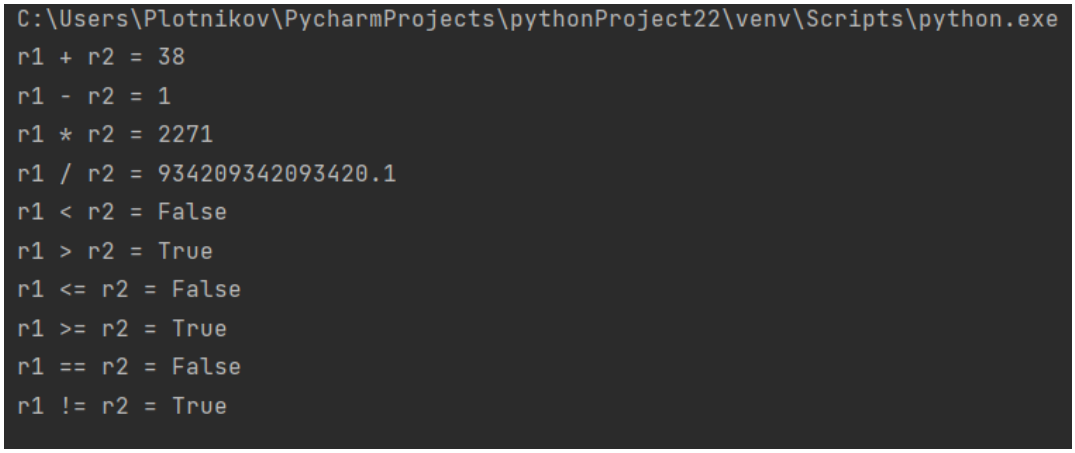

Рисунок 3 – Выполнение второго индивид. задания

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) .

__sub____(self, other) - вычитание (x - y).

__mul____(self, other) - умножение (x * y).

__truediv____(self, other) - деление (x / y).

__floordiv____(self, other) - целочисленное деление (x // y).

__mod____(self, other) - остаток от деления (x % y).

__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 >> 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) ,

___rfloordiv__(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 длялевого операнда не определён соответствующий метод.

__iadd__(self, other) - += .

___isub__(self, other) - -= .

___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) - &= .

__ixor_(self, other) - ^= .

__ior____(self, other) - |= .

3. В каких случаях будут вызваны следующие методы:__add__, __iadd__и____radd__?

- __add_____- a + b
- __iadd_____- a += b
- __radd_____- Если не получилось вызвать метод___add___

4. Для каких целей предназначен метод new? Чем он отличается от метода init?

Метод new используется, когда нужно управлять процессом создания нового экземпляра, a_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 выполнения лабораторной работы успешно приобрел навыки по перегрузке операторов при написании программ с помощью языка программирования Python версии 3.x.