

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ  
Федеральное государственное автономное  
образовательное учреждение высшего образования  
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

**Кафедра инфокоммуникаций**

**Лабораторная работа 2**

Перегрузка операторов в языке Python

Выполнил студент группы ИВТ-б-о-20-1

Симанский М.Ю « » \_\_\_\_\_ 20\_\_ г.

Подпись студента \_\_\_\_\_

Работа защищена « » \_\_\_\_\_ 20\_\_ г.

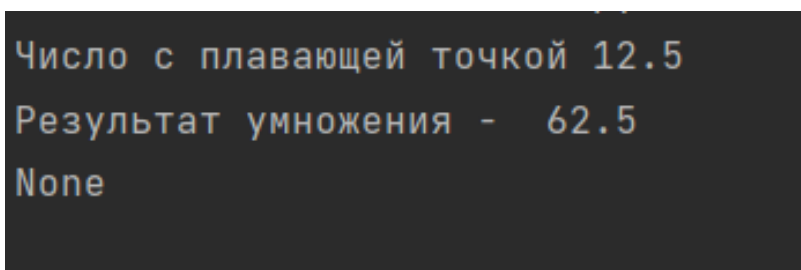
Проверил Воронкин Р.А. \_\_\_\_\_

(подпись)

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

### Задание 1

Выполнить индивидуальное задание 1 лабораторной работы 4.1, максимально задействовав имеющиеся в Python средства перегрузки операторов (рис. 1).



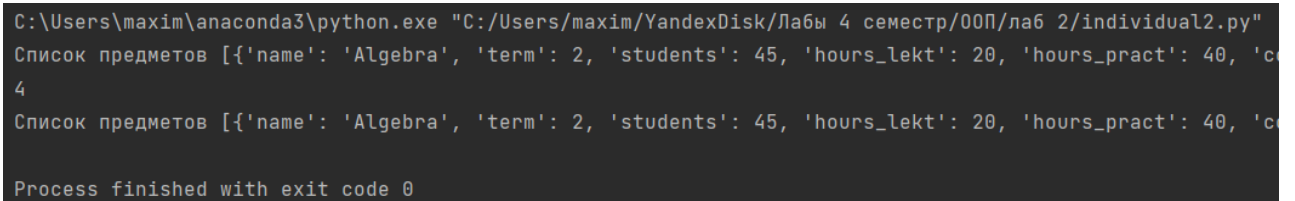
```
Число с плавающей точкой 12.5
Результат умножения - 62.5
None
```

Рисунок 1 – Результат выполнения

### Задание 2

Нагрузка преподавателя за учебный год представляет собой список дисциплин, преподаваемых им в течение года. Одна дисциплина представляется информационным словарем с ключами: название дисциплины, семестр проведения, количество студентов, количество часов аудиторных лекций, количество аудиторных часов практики, вид контроля (зачет или экзамен). Реализовать класс WorkTeacher, моделирующий бланк назначенной преподавателю нагрузки. Класс содержит фамилию преподавателя, дату утверждения, список преподаваемых дисциплин, объем полной нагрузки в часах и в ставках. Дисциплины в списке не должны повторяться. Объем в ставках вычисляется как частное от деления объема в часах на среднюю годовую ставку, одинаковую для всех преподавателей кафедры. Элемент списка преподаваемых дисциплин содержит дисциплину, количество часов,

выделяемых на зачет (0,35 ч на одного студента) или экзамен (0,5 ч на студента), сумму часов по дисциплине. Реализовать добавление и удаление дисциплин; вычисление суммарной нагрузки в часах и ставках. Должен осуществляться контроль за превышением нагрузки более полутора ставок. (рис. 2).



```
C:\Users\maxim\anaconda3\python.exe "C:/Users/maxim/YandexDisk/Лабы 4 семестр/00П/лаб 2/individual2.py"
Список предметов [{'name': 'Algebra', 'term': 2, 'students': 45, 'hours_lekt': 20, 'hours_pract': 40, 'credits': 4}
4
Список предметов [{'name': 'Algebra', 'term': 2, 'students': 45, 'hours_lekt': 20, 'hours_pract': 40, 'credits': 4}
Process finished with exit code 0
```

Рисунок 2 – Результат выполнения

## Ответы на контрольные вопросы

1. Перегрузка операторов — один из способов реализации полиморфизма, когда мы можем задать свою реализацию какого-либо метода в своём классе.

- 2. `__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 ( $x \wedge y$ ).

`__or__(self, other)` - битовое ИЛИ ( $x | y$ ).

3. операция  $x + y$  будет сначала пытаться вызвать `x.__add__(y)`, и только в том случае, если это не получилось, будет пытаться вызвать `y.__radd__(x)`. Аналогично для остальных методов.
4. `__new__(cls[, ...])` — управляет созданием экземпляра. В качестве обязательного аргумента принимает класс (не путать с экземпляром). Должен возвращать экземпляр класса для его последующей его передачи методу `__init__`
5. `__str__(self)` - вызывается функциями `str`, `print` и `format`. Возвращает строковое представление объекта.