

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ**
**Федеральное государственное автономное образовательное
учреждение высшего образования**
«СЕВЕРОКАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
Кафедра инфокоммуникаций
Институт цифрового развития

ОТЧЁТ

по лабораторной работе №4.1

Дисциплина: «Объектно-ориентированное программирование»

Тема: «Элементы объектно-ориентированного программирования в
языке Python»

Вариант 2

Выполнил: студент 3 курса, группы

ИВТ-б-о-21-1

Богдан Александр Анатольевич

Ставрополь 2023

Цель работы: приобретение навыков по работе с классами и объектами при написании программ с помощью языка программирования Python версии 3.x.

Практическая часть:

1. Создал общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и язык программирования Python.

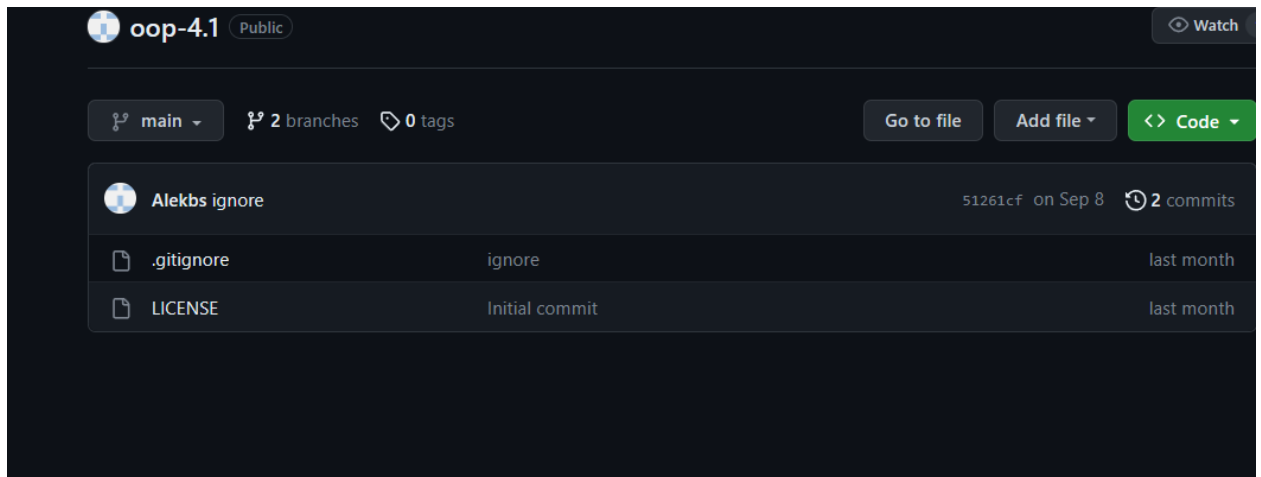


Рисунок 1. Создание репозитория

2. Выполнил клонирование созданного репозитория.

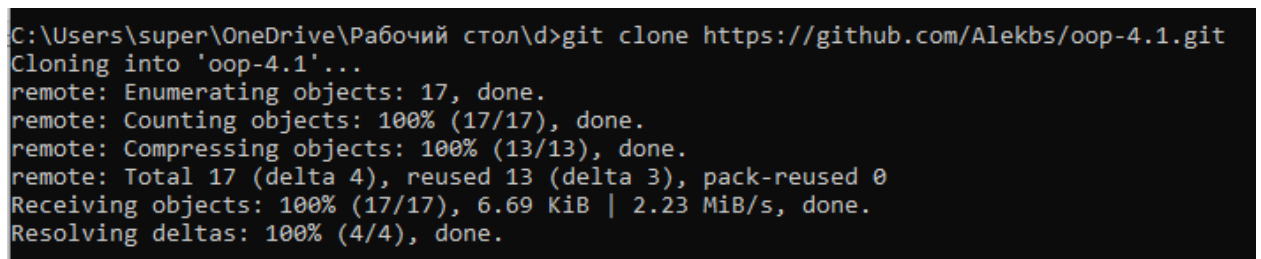
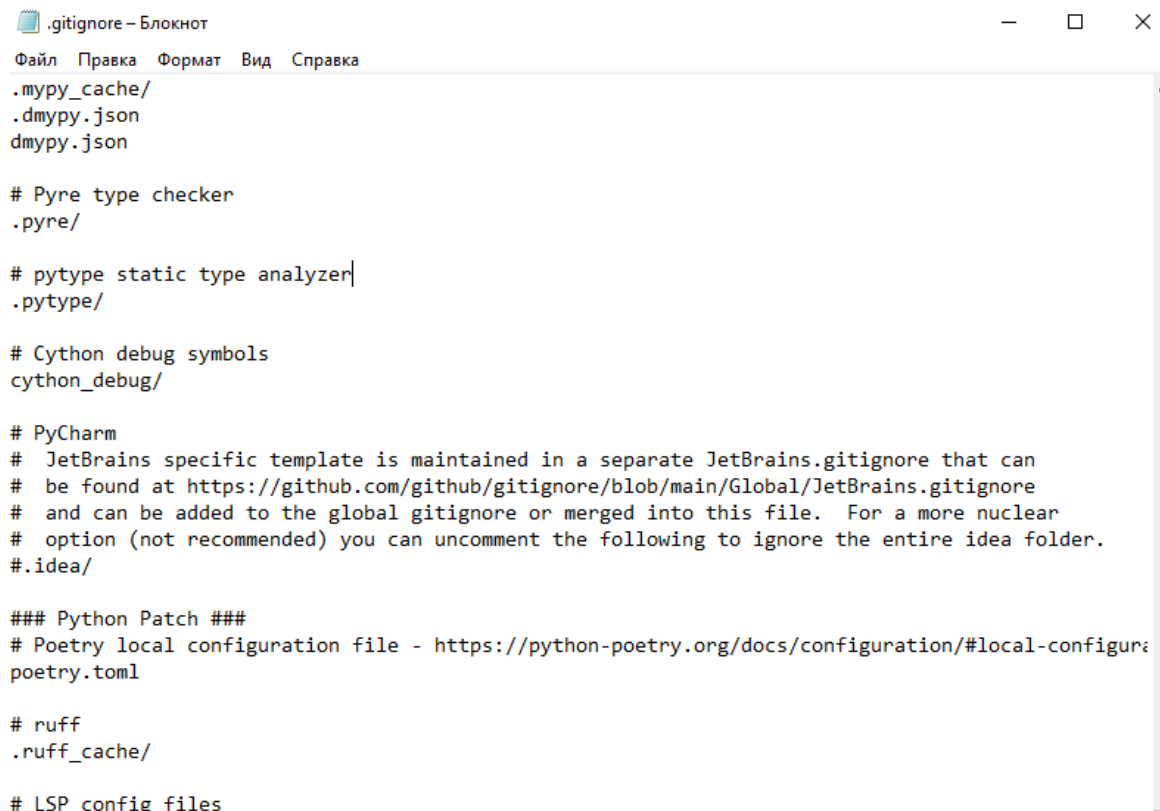


Рисунок 2. Клонирование репозитория

3. Дополнил файл .gitignore необходимыми правилами для работы с IDE PyCharm.

A screenshot of a Notepad window titled ".gitignore - Блокнот". The window contains a .gitignore file with the following content:

```
Файл  Правка  Формат  Вид  Справка
.муру_cache/
.dmpуру.json
dmpуру.json

# Pyre type checker
.pyre/

# pytype static type analyzer
.pytype/

# Cython debug symbols
cython_debug/

# PyCharm
# JetBrains specific template is maintained in a separate JetBrains.gitignore that can
# be found at https://github.com/github/gitignore/blob/main/Global/JetBrains.gitignore
# and can be added to the global gitignore or merged into this file.  For a more nuclear
# option (not recommended) you can uncomment the following to ignore the entire idea folder.
#.idea/

### Python Patch ###
# Poetry local configuration file - https://python-poetry.org/docs/configuration/#local-configuration
poetry.toml

# ruff
.ruff_cache/

# LSP config files
```

Рисунок 3. Изменение файла gitignore

4. Организовал свой репозиторий в соответствии с моделью ветвления git-flow.

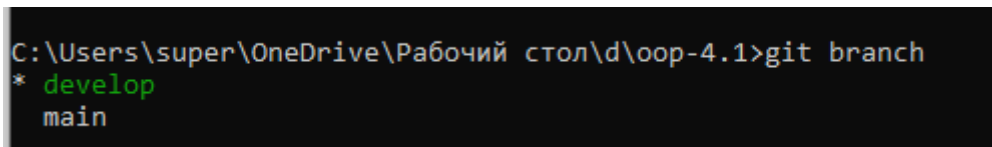
A screenshot of a terminal window showing the output of the 'git branch' command. The prompt is 'C:\Users\super\OneDrive\Рабочий стол\d\oop-4.1>'. The output shows two branches: 'develop' (highlighted in green) and 'main'.

Рисунок 4. Организация репозитория в соответствии с git flow

4. Проработал примеры лабораторной работы.

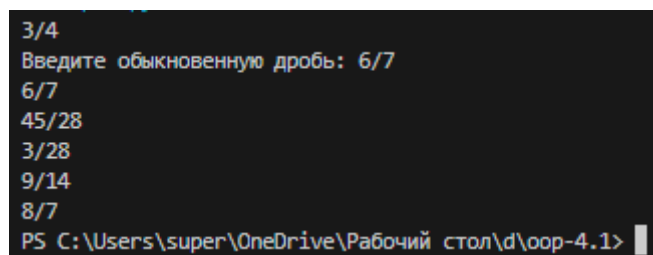
A screenshot of a terminal window showing a sequence of fraction calculations. The prompt is 'PS C:\Users\super\OneDrive\Рабочий стол\d\oop-4.1>'. The output shows: '3/4', 'Введите обыкновенную дробь: 6/7', '6/7', '45/28', '3/28', '9/14', '8/7', and the prompt again.

Рисунок 5. Выполнение примера

6. Выполнил индивидуальные задания.

Задание 1

Парой называется класс с двумя полями, которые обычно имеют имена `first` и `second`. Требуется реализовать тип данных с помощью такого класса. Во всех заданиях обязательно должны присутствовать: метод инициализации `__init__`; метод должен контролировать значения аргументов на корректность; ввод с клавиатуры `read`; вывод на экран `display`.

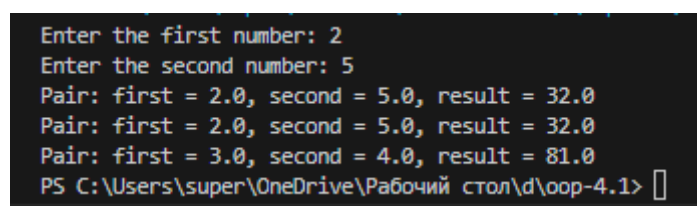
Реализовать внешнюю функцию с именем `make_тип()`, где `тип` — тип реализуемой структуры. Функция должна получать в качестве аргументов значения для полей структуры и возвращать структуру требуемого типа. При передаче ошибочных параметров следует выводить сообщение и заканчивать работу.

Номер варианта необходимо уточнить у преподавателя. В раздел программы, начинающийся после инструкции `if __name__ == '__main__':` добавить код, демонстрирующий возможности разработанного класса.

(вариант 5)

Поле `first` — дробное число; поле `second` — дробное число, показатель степени.

Реализовать метод `power()` — возведение числа `first` в `second` степень. Метод должен правильно работать при любых допустимых значениях `first` и `second`.



```
Enter the first number: 2
Enter the second number: 5
Pair: first = 2.0, second = 5.0, result = 32.0
Pair: first = 2.0, second = 5.0, result = 32.0
Pair: first = 3.0, second = 4.0, result = 81.0
PS C:\Users\super\OneDrive\Рабочий стол\d\oop-4.1>
```

Рисунок 6. Выполнение индивидуального задания 1

Задание 2

Составить программу с использованием классов и объектов для решения задачи. Во всех заданиях, помимо указанных в задании операций, обязательно должны быть реализованы следующие методы:

метод инициализации `__init__` ;

ввод с клавиатуры `read` ;

вывод на экран `display` .

Номер варианта необходимо уточнить у преподавателя. В раздел программы, начинающийся после инструкции `if __name__ == '__main__':` добавить код, демонстрирующий возможности разработанного класса.

(вариант 2)

Создать класс `ModelWindow` для работы с моделями экранных окон. В качестве полей задаются: заголовок окна, координаты левого верхнего угла, размер по горизонтали, размер по вертикали, цвет окна, состояние «видимое/невидимое», состояние «с рамкой/без рамки». Координаты и размеры указываются в целых числах. Реализовать операции: передвижение окна по горизонтали, по вертикали; изменение высоты и/или ширины окна изменение цвета; изменение состояния, опрос состояния. Операции передвижения и изменения размера должны осуществлять проверку на пересечение границ экрана. Функция вывода на экран должна индуцировать состояние полей объекта.

```
{'title': 'Window 1', 'x': 100, 'y': 100, 'width': 200, 'height': 150, 'color': 'blue', 'visibility': 'visible', 'with_border': True}
{'title': 'Window 1', 'x': 150, 'y': 130, 'width': 250, 'height': 200, 'color': 'red', 'visibility': 'hidden', 'with_border': False}
```

Рисунок 7. Выполнение индивидуального задания 2

Контрольные вопросы:

1. Как осуществляется объявление класса в языке Python?

Классы объявляются с помощью ключевого слова `class` и имени класса:

```
# class syntax
class MyClass:
    var = ... # некоторая переменная

    def do_smt(self):
        # какой-то метод
```

2. Чем атрибуты класса отличаются от атрибутов экземпляра?

Атрибуты класса являются общими для всех объектов класса, а атрибуты экземпляра специфическими для каждого экземпляра. Более того, атрибуты класса определяются внутри класса, но вне каких-либо методов, а атрибуты экземпляра обычно определяются в методах, чаще всего в `__init__`.

3. Каково назначение методов класса?

Методы определяют функциональность объектов, принадлежащих конкретному классу.

4. Для чего предназначен метод `__init__()` класса?

Чтобы настроить начальное состояние экземпляра, используется метод `__init__`.

Метод `__init__` является конструктором. Конструкторы - это концепция объектно-ориентированного программирования. Класс может иметь один и только один конструктор. Если `__init__` определен внутри класса, он автоматически вызывается при создании нового экземпляра класса.

Метод `__init__` указывает, какие атрибуты будут у экземпляров нашего класса.

5. Каково назначение `self`?

Аргумент `self` представляет конкретный экземпляр класса и позволяет нам получить доступ к его атрибутам и методам.

Важно использовать параметр `self` внутри метода, если мы хотим сохранить значения экземпляра для последующего использования.

6. Как добавить атрибуты в класс?

Метод `__init__` указывает, какие атрибуты будут у экземпляров нашего класса.

Добавить атрибут можно следующим образом:

Объект.атрибут = значение

7. Как осуществляется управление доступом к методам и атрибутам в языке Python?

Хорошим тоном считается, что для чтения/изменения какого-то атрибута должны использоваться специальные методы, которые называются `getter/setter`, их можно реализовать, но ничего не мешает изменить атрибут напрямую. При этом есть соглашение, что метод или атрибут, который начинается с нижнего подчеркивания, является скрытым, и снаружи класса трогать его не нужно (хотя сделать это можно).

8. Каково назначение функции `isinstance` ?

Встроенная функция `isinstance(obj, Cls)` , используемая при реализации методов арифметических операций и операций отношения, позволяет узнать что некоторый объект `obj` является либо экземпляром класса `Cls` либо экземпляром одного из потомков класса `Cls`.

Вывод: в результате выполнения работы были приобретены навыки по работе с классами и объектами при написании программ с помощью языка программирования Python версии 3.x.