Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт перспективной инженерии
Департамент цифровых, робототехнических систем и электроники

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №4
дисциплины «Объектно-ориентированное программирование»

Выполнил:
Евдаков Евгений Владимирович
3 курс, группа ИВТ-б-о-22-1,
09.03.01 «Информатика и вычислительная техника»,
направленность (профиль) «Программное обеспечение средств вычислительной техники и автоматизированных систем», очная форма обучения

_____
(подпись)

Проверил:
Воронкин Р. А., доцент департамента цифровых, робототехнических систем и электроники института перспективной инженерии

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2024 г.

Тема: работа с исключениями в языке Python.

Цель: приобретение навыков по работе с исключениями при написании программ с помощью языка программирования Python версии 3.x.

Ход работы:

Задание 1. Создал общедоступный репозиторий на GitHub, в котором использована лицензий MIT и язык программирования Python, также добавил файл .gitignore с необходимыми правилами. Клонировал свой репозиторий на свой компьютер. Организовал свой репозиторий в соответствие с моделью ветвления git-flow, появилась новая ветка develop в которой буду выполнять дальнейшие задачи.

```
C:\Users\evdak\LR1->git clone https://github.com/EvgenyEvdakov/Laba_4.4.git
Cloning into 'Laba_4.4'...
remote: Enumerating objects: 20, done.
remote: Counting objects: 100% (20/20), done.
remote: Compressing objects: 100% (16/16), done.
remote: Total 20 (delta 2), reused 15 (delta 1), pack-reused 0 (from 0)
Receiving objects: 100% (20/20), 9.09 KiB | 1.30 MiB/s, done.
Resolving deltas: 100% (2/2), done.
```

Рисунок 1. Клонирование репозитория

Задание 2. Создал файлы poetry.lock и pyproject.toml, также установил необходимые пакеты isort, black, flake8, mypy, pre-commit, pytest.

```
(.venv) PS C:\Users\evdak\Laba_4.4> poetry init

This command will guide you through creating your pyproject.toml config.

Package name [laba_4.4]:
Version [0.1.0]:
Description []:
Author [Evgeni <kkrutkov02@gmail.com>, n to skip]:
License []:
Compatible Python versions [^3.13]:

Would you like to define your main dependencies interactively? (yes/no) [yes]
You can specify a package in the following forms:
  - A single name (requests): this will search for matches on PyPI
  - A name and a constraint (requests@^2.23.0)
  - A git url (git+https://github.com/python-poetry/poetry.git)
  - A git url with a revision (git+https://github.com/python-poetry/poetry.git#develop
  - A file path (../my-package/my-package.whl)
  - A directory (../my-package/)
  - A url (https://example.com/packages/my-package-0.1.0.tar.gz)

Package to add or search for (leave blank to skip):

Would you like to define your development dependencies interactively? (yes/no) [yes]
Package to add or search for (leave blank to skip):

Generated file
```

Рисунок 2. Создание виртуального окружения

Задание 3. Создал проект PyCharm в папке репозитория. Приступил к работе с примером. Добавил новый файл primer1.py.

Условие примера: для примера 2 лабораторной работы 9 добавьте возможность работы с исключениями и логгирование.

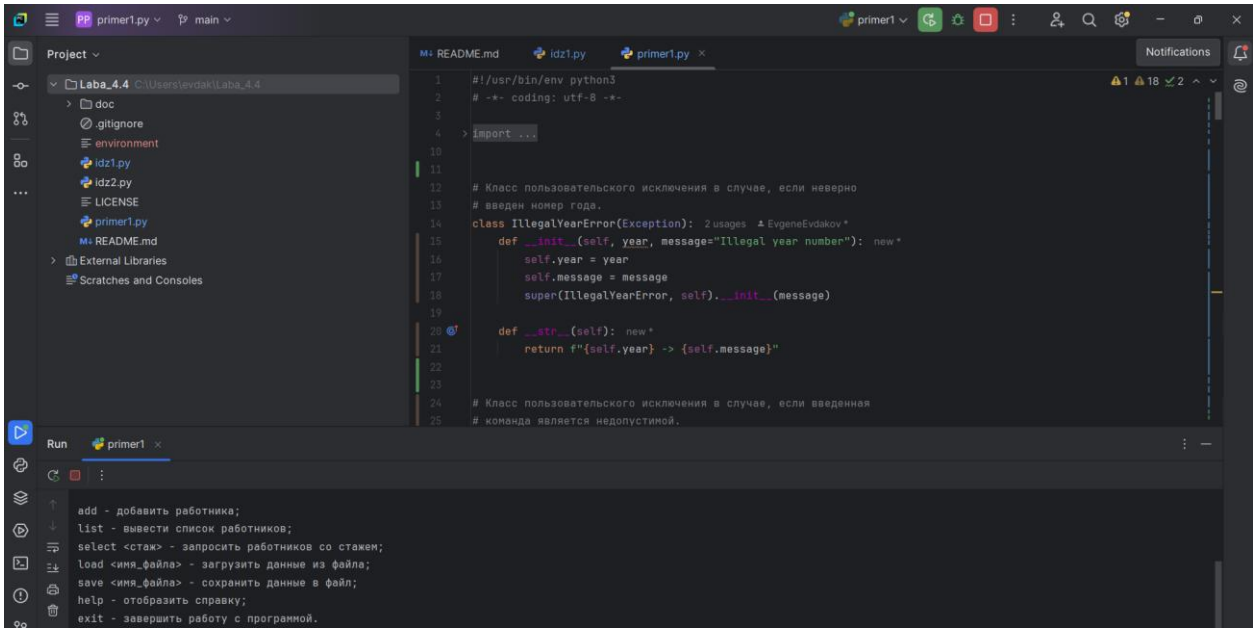

Рисунок 3. Выполнение первого примера

Задание 4.

Индивидуальное задание

Вариант 6

Создал новый файл под названием idz1.py.

Условие задания: необходимо выполнить индивидуальное задание 1 лабораторной работы 2.19, добавив возможность работы с исключениями и логгирование.

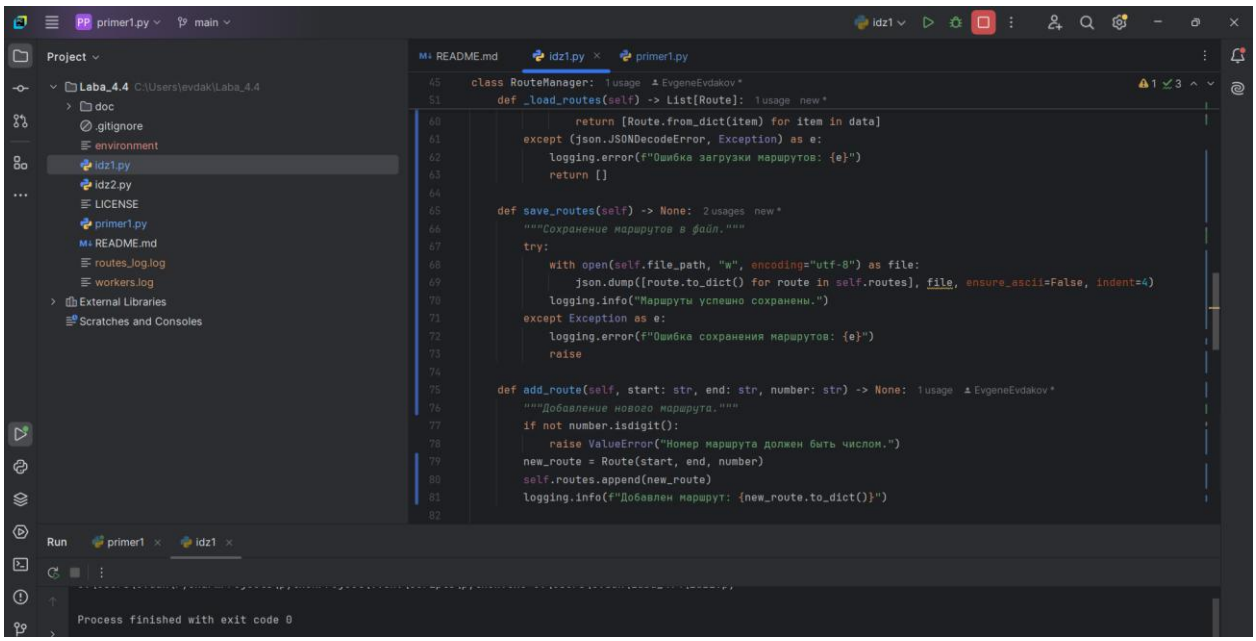

Рисунок 4. Код первого индивидуального задания

Запусти программу на выполнение:

```
PS C:\Users\evdak> cd C:\Users\evdak\Laba_4.4
PS C:\Users\evdak\Laba_4.4> python idz1.py --add
Введите начальный пункт маршрута: Ставрополь
Введите конечный пункт маршрута: Омск
Введите номер маршрута: 12
Маршрут успешно добавлен.
```

Рисунок 5. Выполнение первого индивидуального задания

Создал новый файл под названием idz2.py.

Условие задания: необходимо изучить возможности модуля logging. Добавить для предыдущего задания вывод в файлы лога даты и времени выполнения пользовательской команды с точностью до миллисекунды.

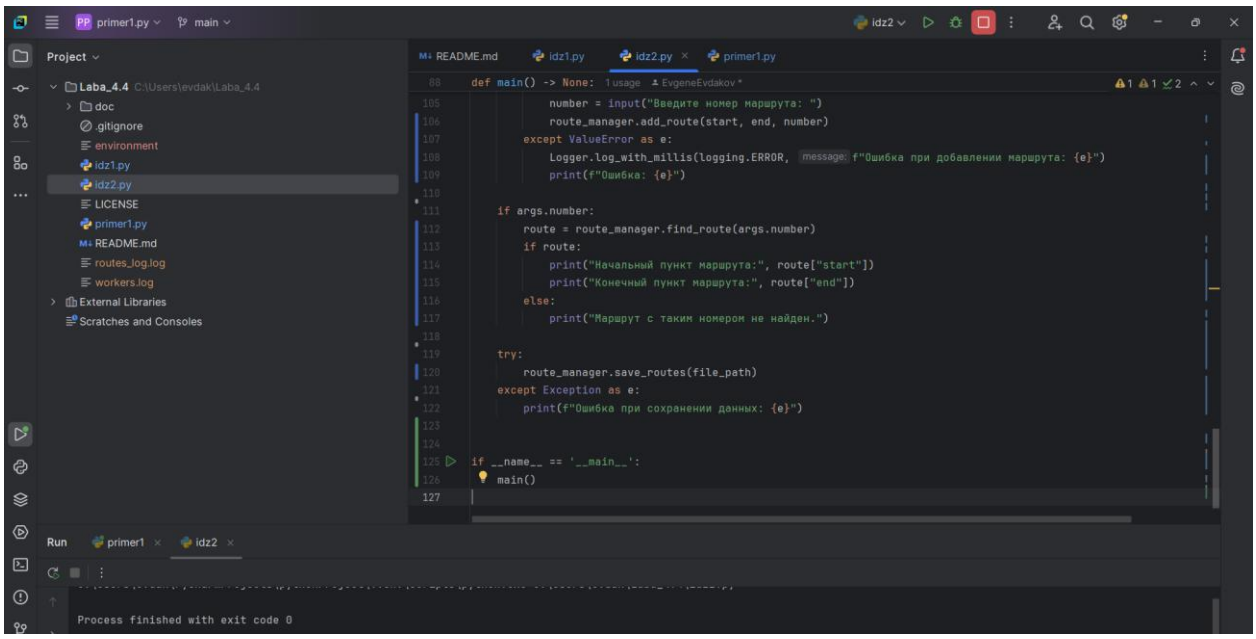

Рисунок 6. Код второго индивидуального задания

Запустим программу на выполнение:

```
PS C:\Users\evdak\Laba_4.4> python idz2.py --add
Введите начальный пункт маршрута: Москва
Введите конечный пункт маршрута: Киев
Введите номер маршрута: 45
PS C:\Users\evdak\Laba_4.4> python idz2.py --add
Введите начальный пункт маршрута: Уфа
Введите конечный пункт маршрута: Саратов
Введите номер маршрута: 23
```

Рисунок 7. Выполнение второго индивидуального задания

Задание 5.

После выполнения работы на ветке develop, слил ее с веткой main и отправил изменения на удаленный сервер. Создал файл envirement.yml и деактивировал виртуальное окружение.

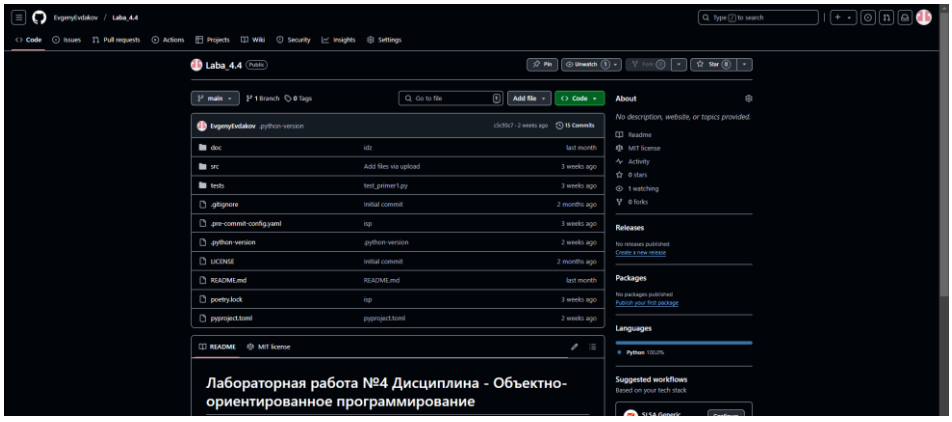

Рисунок 8. Готовый репозиторий

Ссылка: https://github.com/EvgenyEvdakov/Laba_4.4

Ответы на контрольные вопросы:

1. **Какие существуют виды ошибок в языке программирования Python?**

Синтаксические ошибки возникают при нарушении правил синтаксиса Python.

Исключения - ошибки, возникающие во время выполнения программы. Некоторые распространенные виды исключений:

ValueError – неверное значение.

TypeError – недопустимый тип данных.

IndexError – выход за пределы списка.

KeyError – отсутствующий ключ в словаре.

ZeroDivisionError – деление на ноль.

ImportError – ошибка импорта модуля.

FileNotFoundError – файл не найден.

2. **Как осуществляется обработка исключений в языке программирования Python?**

Обработка исключений осуществляется с помощью конструкции try-except.

3. **Для чего нужны блоки finally и else при обработке исключений?**

Finally выполняется всегда, независимо от того, возникло исключение или нет. Обычно используется для освобождения ресурсов (например, закрытия файлов или соединений).

else выполняется, если исключение не возникло.

4. **Как осуществляется генерация исключений в языке Python?**

Исключения можно генерировать вручную с помощью ключевого слова raise.

5. **Как создаются классы пользовательский исключений в языке Python?**

Для создания пользовательских исключений создается класс, наследующийся от встроенного класса Exception.

class MyCustomError(Exception):

def init(self, message):

super().init(message)

6. **Каково назначение модуля logging?**

Модуль logging используется для регистрации событий (логгирования) во время выполнения программы. Логи могут быть использованы для:

- Отладки.
- Мониторинга работы программы.
- Анализа ошибок.

7. **Какие уровни логгирования поддерживаются модулем logging? Приведите примеры, в которых могут быть использованы сообщения с этим уровнем журналирования.**

Модуль logging поддерживает следующие уровни логгирования:

- DEBUG: Детальная информация для отладки.

logging.debug("Переменная x равна 10")

- INFO: Общая информация о выполнении программы.

logging.info("Запущен процесс обработки данных")

- WARNING: Уведомления о возможных проблемах.

logging.warning("Место на диске заканчивается")

- ERROR: Ошибки, препятствующие нормальной работе программы.

logging.error("Не удалось открыть файл")

- CRITICAL: Критические ошибки, требующие немедленного вмешательства.

logging.critical("Сбой в системе: немедленно обратитесь к администратору")

Вывод: приобрел навыков по работе с исключениями при написании программ с помощью языка программирования Python версии 3.x.