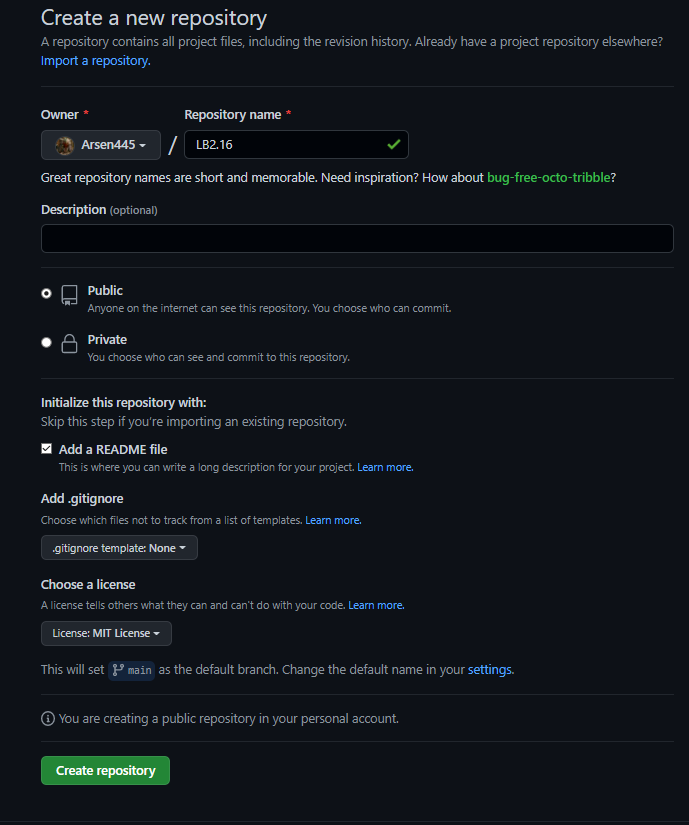
Лабораторная работа №11

Выполнил Эсеналиев Арсен

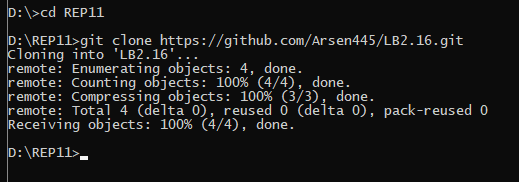
ИВТ-б-о-21-1

**Цель:** приобретение навыков по работе с данными формата JSON с помощью языка программирования Python версии 3.x.

1. Создал общедоступный репозиторий на GitHub c MIT



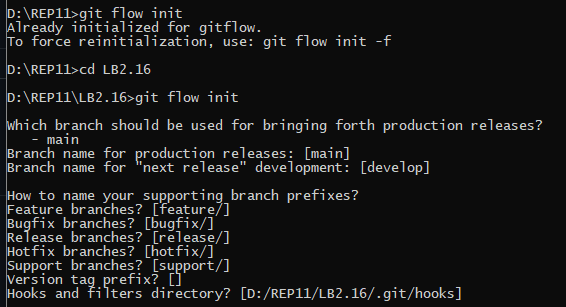
1. Выполнил клонирование созданного репозитория.



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

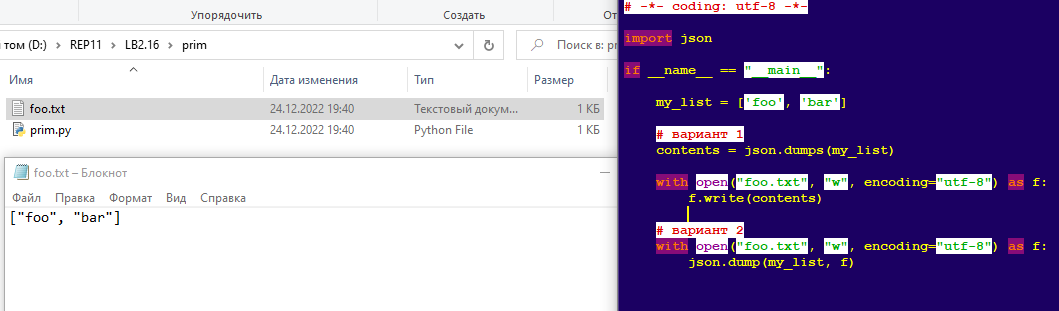


1. Организовал репазиторий в соответствии с моделью ветвления git-flow.

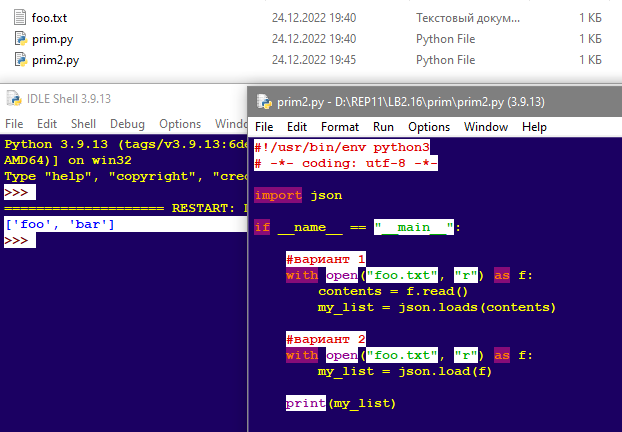


1. Проработал примеры лабораторной работы.

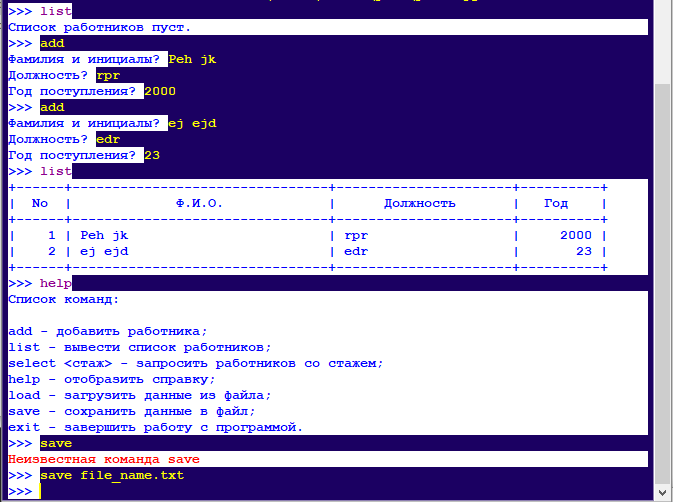
**Запись в файл**



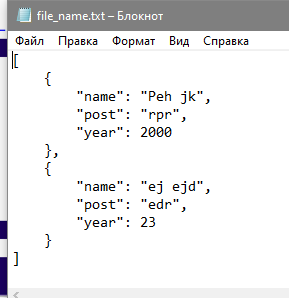
**Чтение из файла**

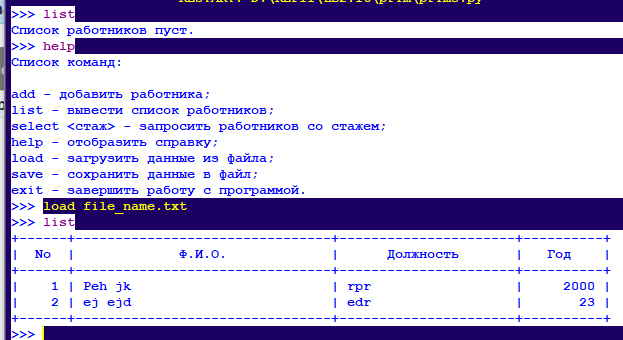


**Пример1**

****

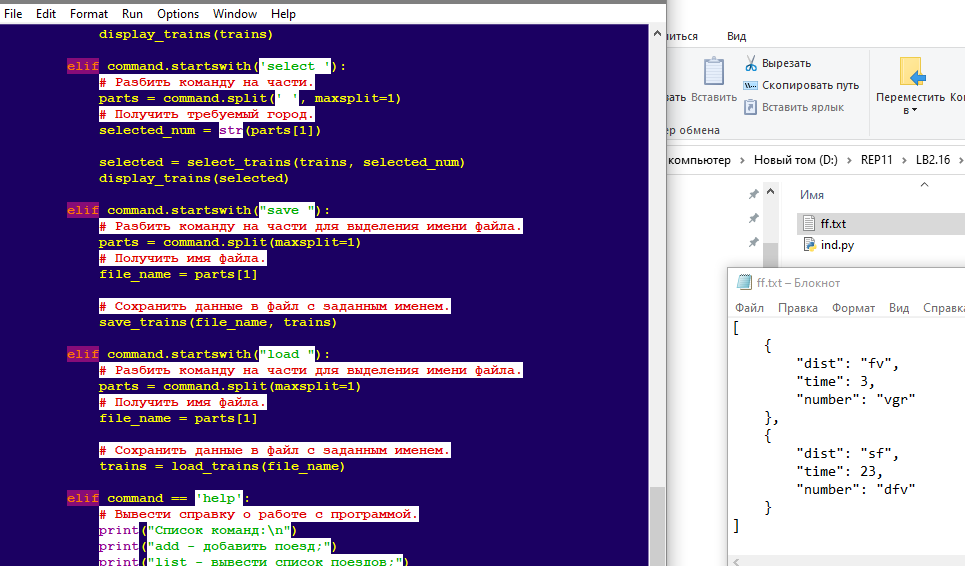
****

****

****

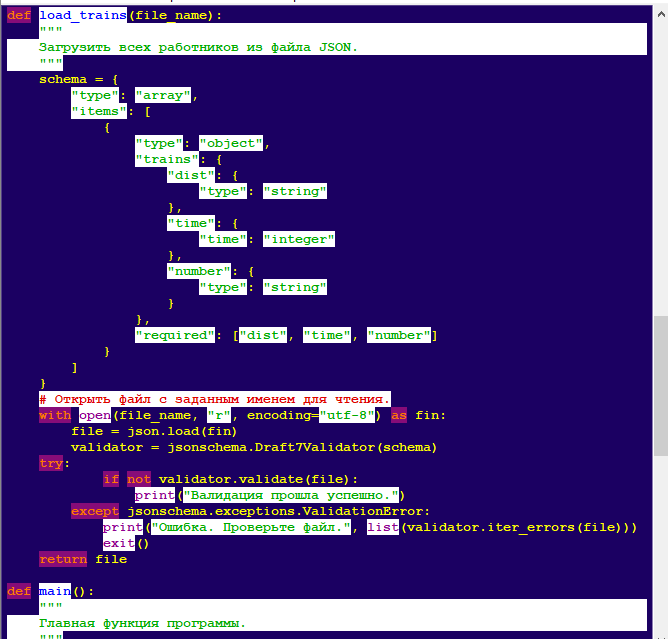
**Индивидуальное 1**

Для своего варианта лабораторной работы 2.8 необходимо дополнительно реализовать сохранение и чтение данных из файла формата JSON. Необходимо также проследить за тем, чтобы файлы генерируемый этой программой не попадали в репозиторий лабораторной работы.

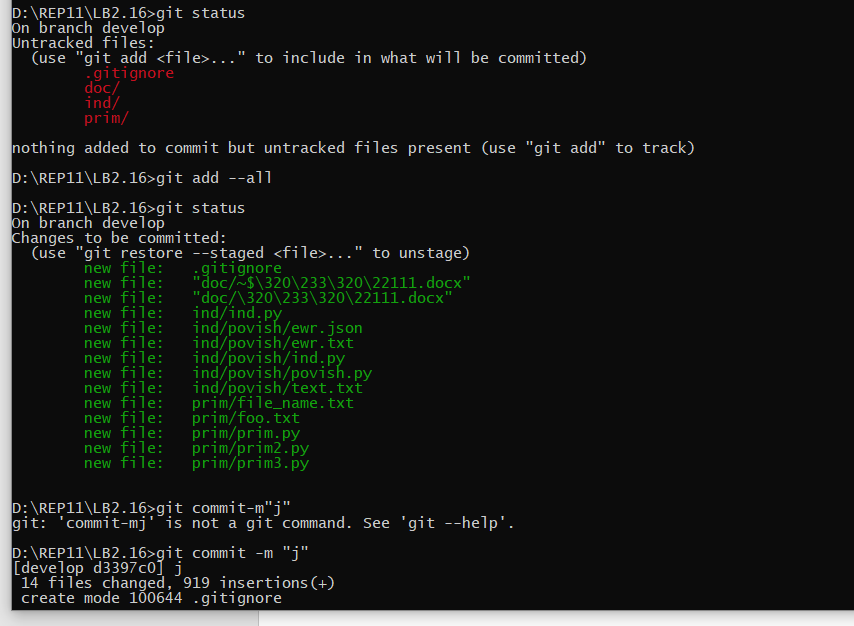


**Повышенная сложность**

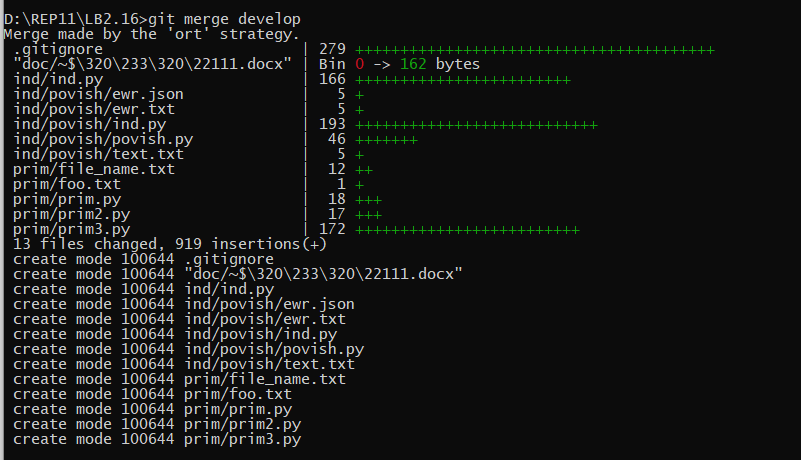
Очевидно, что программа в примере 1 и в индивидуальном задании никак не проверяет правильность загружаемых данных формата JSON. В следствие чего, необходимо после загрузки из файла JSON выполнять валидацию загруженных данных. Валидацию данных необходимо производить с использованием спецификации JSON Schema, описанной на сайте https://json-sch ema.org/. Одним из возможных вариантов работы с JSON Schema является использование пакета jsonschema , который не является частью стандартной библиотеки Python. Таким образом, необходимо реализовать валидацию загруженных данных с помощью спецификации JSON Schema.



1. Зафиксируйте сделанные изменения в репозитории.(после создания веток не запушил, поэтому не работало)



1. Выполните слияние ветки для разработки с веткой main/master.



**Контрольные вопросы:**

**1. Как открыть файл в языке Python только для чтения?**

Чтобы открыть файл для чтения, мы используем режим r. Для чтения мы

воспользуемся функцией read(size), если параметр size не указан, функция вернет нам всю строку. file = open("text.txt", 'r', encoding = 'utf-8').

**2. Как открыть файл в языке Python только для записи?**

В Python открытие файлов выполняется с помощью функции open(), которой передается два аргумента - имя файла и режим. Файл может быть открыт

в режиме чтения, записи, добавления.

**3. Как прочитать данные из файла в языке Python?**

Чтение данных из файла осуществляется с помощью методов read(размер) и readline(). Метод read(размер) считывает из файла определенное количество символов, переданное в качестве аргумента.

**4. Как записать данные в файл в языке Python?**

Запись данных в файл. Записать данные в файл можно с помощью метода write().

**5. Как закрыть файл в языке Python?**

После того, как мы открыли файл, и выполнили все нужные операции,

нам необходимо его закрыть. Для закрытия файла используется функция

close().

**6. Изучите самостоятельно работу конструкции with ... as.** **Каково ее**

**назначение в языке?**

Конструкция with ... as используется для оборачивания выполнения

блока инструкций менеджером контекста. Если в конструкции with - as было

несколько выражений, то это эквивалентно нескольким вложенным конструкциям

**7. Изучите самостоятельно документацию Python по работе с файлами. Какие помимо рассмотренных существуют методы записи/чтения**

**информации из файла?**

Один из самых распространенных способов вывести данные в Python –

это напечатать их в консоли. Если вы находитесь на этапе изучения языка, такой способ является основным для того, чтобы быстро просмотреть результат

свой работы

**8. Какие существуют, помимо рассмотренных, функции модуля os**

**для работы с файловой системой?**

os.chdir(path) - смена текущей директории.

os.chmod (path, mode, \*, dir\_fd=None, follow\_symlinks=True) - смена прав

доступа к объекту (mode - восьмеричное число).

os.chown (path, uid, gid, \*, dir\_fd=None, follow\_symlinks=True) - меняет id

владельца и группы (Unix).

os.getcwd() - текущая рабочая директория.

os.link (src, dst, \*, src\_dir\_fd=None, dst\_dir\_fd=None,

follow\_symlinks=True) - создаѐт жѐсткую ссылку.

os.listdir (path=".") - список файлов и директорий в папке.

os.mkdir (path, mode=0o777, \*, dir\_fd=None) - создаѐт директорию.

OSError, если директория существует.

os.makedirs (path, mode=0o777, exist\_ok=False) - создаѐт директорию, создавая при этом промежуточные директории.

os.remove (path, \*, dir\_fd=None) - удаляет путь к файлу.

os.rename (src, dst, \*, src\_dir\_fd=None, dst\_dir\_fd=None) - переименовывает файл или директорию из src в dst.

os.renames (old, new) - переименовывает old в new, создавая промежуточные директории.

os.replace (src, dst, \*, src\_dir\_fd=None, dst\_dir\_fd=None) - переименовывает из src в dst с принудительной заменой.

os.rmdir (path, \*, dir\_fd=None) - удаляет пустую директорию.

os.removedirs (path) - удаляет директорию, затем пытается удалить родительские директории, и удаляет их рекурсивно, пока они пусты.

os.sync() - записывает все данные на диск (Unix).

os.truncate (path, length) - обрезает файл до длины length.

os.utime (path, times=None, \*, ns=None, dir\_fd=None,

follow\_symlinks=True) - модификация времени последнего доступа и изменения файла. Либо times - кортеж (время доступа в секундах, время изменения в секундах), либо ns - кортеж (время доступа в наносекундах, время изменения в наносекундах).

os.walk (top, topdown=True, onerror = None, followlinks=False) – генерация имѐн файлов в дереве каталогов, сверху вниз (если topdown равен True),

либо снизу вверх (если False). Для каждого каталога функция walk возвращает

кортеж (путь к каталогу, список каталогов, список файлов).