

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

Институт цифрового развития
Кафедра инфокоммуникаций

ОТЧЕТ
ПО РАБОТЕ №2.15
дисциплины «Программирование на языке Python»

Выполнил:

Хачатрян Владимир Владимирович
2 курс, группа ИТС-б-о-22-1,
11.03.02 «Инфокоммуникационные
технологии и системы связи»,
направленность (профиль)
«Инфокоммуникационные системы и
сети», очная форма обучения

(подпись)

Руководитель практики:

Воронкин Р.А., канд. тех. наук, доцент,
доцент кафедры инфокоммуникаций

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2023 г.

Тема: Работа с файлами в языке Python.

Цель работы: приобретение навыков по работе с текстовыми файлами при написании программ с помощью языка программирования Python версии 3.x, изучение основных методов модуля os для работы с файловой системой, получение аргументов командной строки.

Выполнения работы:

1. Изучил теоретический материал работы, создал общедоступный репозиторий на GitHub, в котором использована лицензий MIT и язык программирования Python, также добавил файл .gitignore с необходимыми правилами.

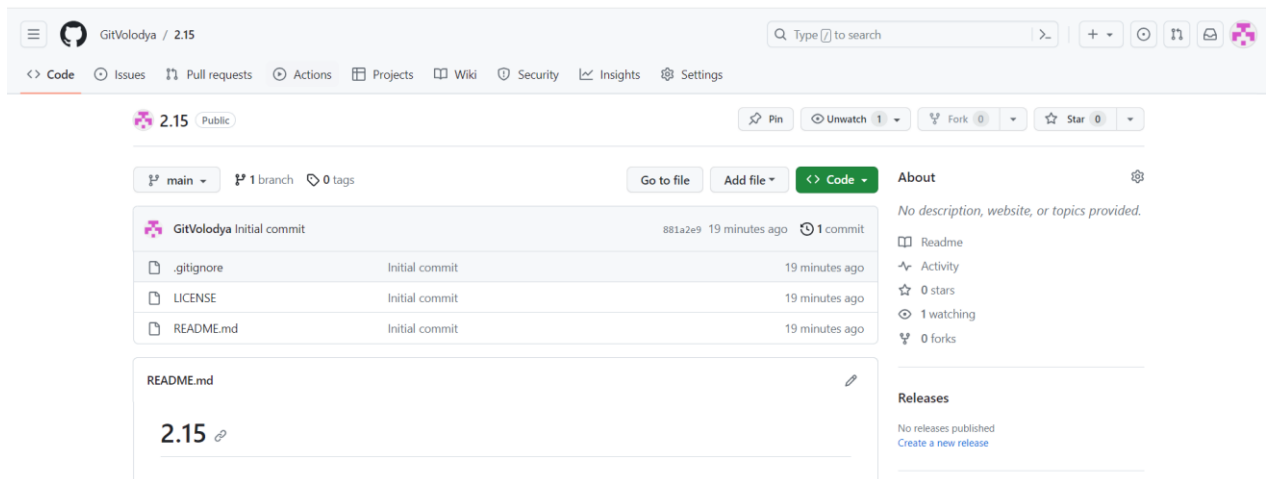


Рисунок 1. Репозиторий

2. Клонировал репозиторий на свой компьютер.

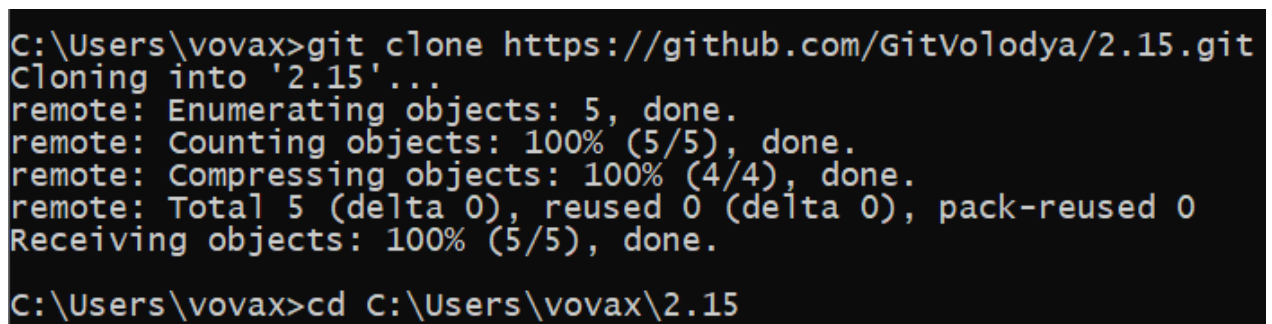


Рисунок 2. Клонирование репозитория

3. Создал виртуальное окружение miniconda.

```
(base) PS C:\Users\vovax\2.15> conda create -n 2.15 python=3.11
```

Рисунок 3. Создание

4. Активировал виртуальное окружение miniconda.

```
(base) PS C:\Users\vovax\2.15> conda activate 2.15  
(2.15) PS C:\Users\vovax\2.15> conda list
```

Рисунок 4. Активация

5. Организовал репозиторий в соответствии с моделью ветвления git-flow. Появилась ветка develop.

```
C:\Users\vovax\2.15>git flow init
```

Рисунок 5. Модель ветвления git-flow

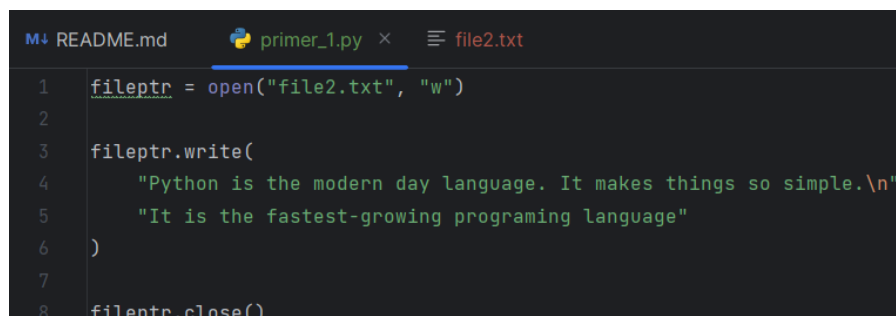
6. Сформировал файл environment.yml.

```
(2.15) PS C:\Users\vovax\2.15> conda env export > environment.yml
```

Рисунок 6. environment.yml

7. Выполнение примеров

Пример 1. Мы открыли файл в режиме w. Файл file2.txt не существует, он создал новый файл, и мы записали его содержимое с помощью функции write ().



```
1 fileptr = open("file2.txt", "w")  
2  
3 fileptr.write(  
4     "Python is the modern day language. It makes things so simple.\n"  
5     "It is the fastest-growing programing language"  
6 )  
7  
8 fileptr.close()
```

Рисунок 7. Код примера 1

```
M↓ README.md  primer_1.py  file2.txt ×
1 Python is the modern day language. It makes things so simple.
2 It is the fastest-growing programing language
```

Рисунок 8. Результат примера 1

Пример 3. Мы вызвали функцию `readline()` два раза, поэтому она считывает две строки из файла.

```
M↓ README.md  primer_1.py  primer_3.py ×  file2.txt
1 fileptr = open("file2.txt", "r")
2
3 content1 = fileptr.readline()
4 content2 = fileptr.readline()
5
6 print(content1)
7 print(content2)
8
9 fileptr.close()
```

Рисунок 9. Код примера 3

```
Python is the modern day language. It makes things so simple.

It is the fastest-growing programing language
```

Рисунок 10. Результат примера 3

Пример 5.

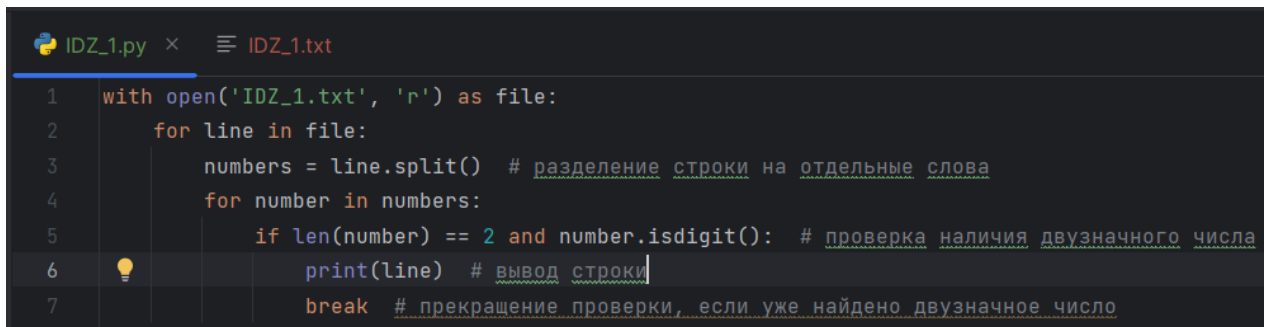
```
M↓ README.md  primer_1.py  primer_3.py  primer_5.py ×
1 fileptr = open("newfile.txt", "x")
2 print(fileptr)
3 if fileptr:
4     print("File created successfully")
5
6 fileptr.close()
```

Рисунок 11. Код примера 5

```
<_io.TextIOWrapper name='newfile.txt' mode='x' encoding='cp1251'>
File created successfully
```

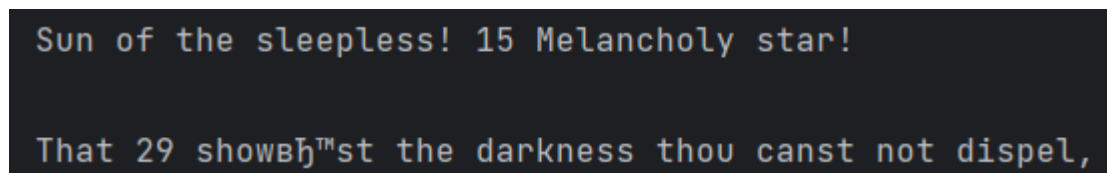
Рисунок 12. Результат примера 5

8. Выполнение индивидуального задания.



```
1 with open('IDZ_1.txt', 'r') as file:
2     for line in file:
3         numbers = line.split() # разделение строки на отдельные слова
4         for number in numbers:
5             if len(number) == 2 and number.isdigit(): # проверка наличия двузначного числа
6                 print(line) # вывод строки
7                 break # прекращение проверки, если уже найдено двузначное число
```

Рисунок 13. Код ИДЗ1.

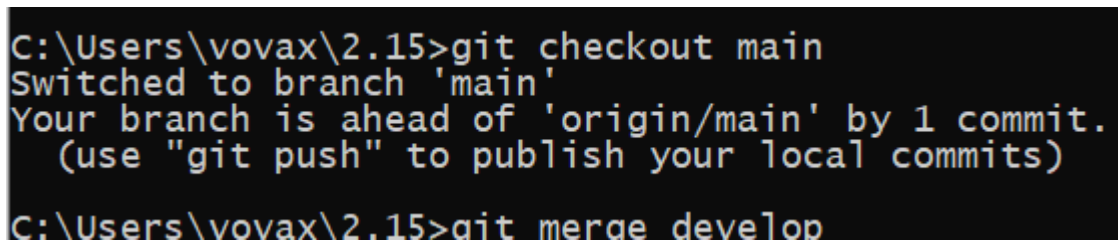


```
Sun of the sleepless! 15 Melancholy star!

That 29 showbħ™st the darkness thou canst not dispel,
```

Рисунок 14. Результат ИДЗ1.

9. Слил ветку develop с веткой main и отправил на удаленный сервер.



```
C:\Users\vovax\2.15>git checkout main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 1 commit.
(use "git push" to publish your local commits)

C:\Users\vovax\2.15>git merge develop
```

Рисунок 15. Слияние веток

Ссылка на репозиторий: <https://github.com/GitVolodya/2.15.git>

Контрольные вопросы:

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

Для открытия файла в режиме только для чтения в Python используется функция open с аргументом 'r'.

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

Для открытия файла в режиме только для записи в Python используется функция open с аргументом 'w'.

3. Как прочитать данные из файла в языке Python?

Чтение данных из файла в Python можно осуществить с помощью метода read.

4. Как записать данные в файл в языке Python?

Запись данных в файл в Python можно осуществить с помощью метода write.

5. Как закрыть файл в языке Python?

Чтобы закрыть файл в Python после работы с ним, используется метод close.

6. Изучите самостоятельно работу конструкции with ... as. Каково ее назначение в языке Python? Где она может быть использована еще, помимо работы с файлами?

Конструкция with ... as используется в Python для автоматического управления ресурсами. В случае работы с файлами, она гарантирует, что файл будет автоматически закрыт по окончании блока кода, независимо от того, произошло исключение или нет.

7. Изучите самостоятельно документацию Python по работе с файлами. Какие помимо рассмотренных существуют методы записи/чтения информации из файла?

Помимо методов read() и write(), существуют другие методы работы с файлами в Python, такие как readline() для чтения одной строки из файла, readlines() для чтения всех строк из файла в список, seek() для перемещения указателя чтения/записи по файлу, tell() для получения текущей позиции указателя и другие. Подробнее о методах работы с файлами можно узнать из документации Python.

8. Какие существуют, помимо рассмотренных, функции модуля os для работы с файловой системой?

Некоторые функции модуля os для работы с файловой системой в Python:

- os.rename() для переименования файла или директории.
- os.remove() для удаления файла.
- os.mkdir() для создания директории.

- `os.getcwd()` для получения текущей рабочей директории.
- `os.path.exists()` для проверки существования файла или директории.
- `os.path.isfile()` для проверки, является ли путь файлом.
- `os.path.isdir()` для проверки, является ли путь директорией.
- `os.path.join()` для объединения путей к файлам или директориям.

Вывод: приобрели навыки по работе с текстовыми файлами при написании программ с помощью языка программирования Python версии 3.x, изучили основных методов модуля `os` для работы с файловой системой, получение аргументов командной строки.