

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

Институт цифрового развития
Кафедра инфокоммуникаций

«Работа с файлами в языке Python»

ОТЧЕТ
по лабораторной работе №18
дисциплины
«Основы программной инженерии»

Выполнила:

Кувшин Ирина Анатольевна
2 курс, группа ПИЖ-б-о-21-1,
011.03.04 «Программная инженерия»,
направленность (профиль) «Разработка
и сопровождение программного
обеспечения», очная форма обучения

(подпись)

Проверил:

(подпись)

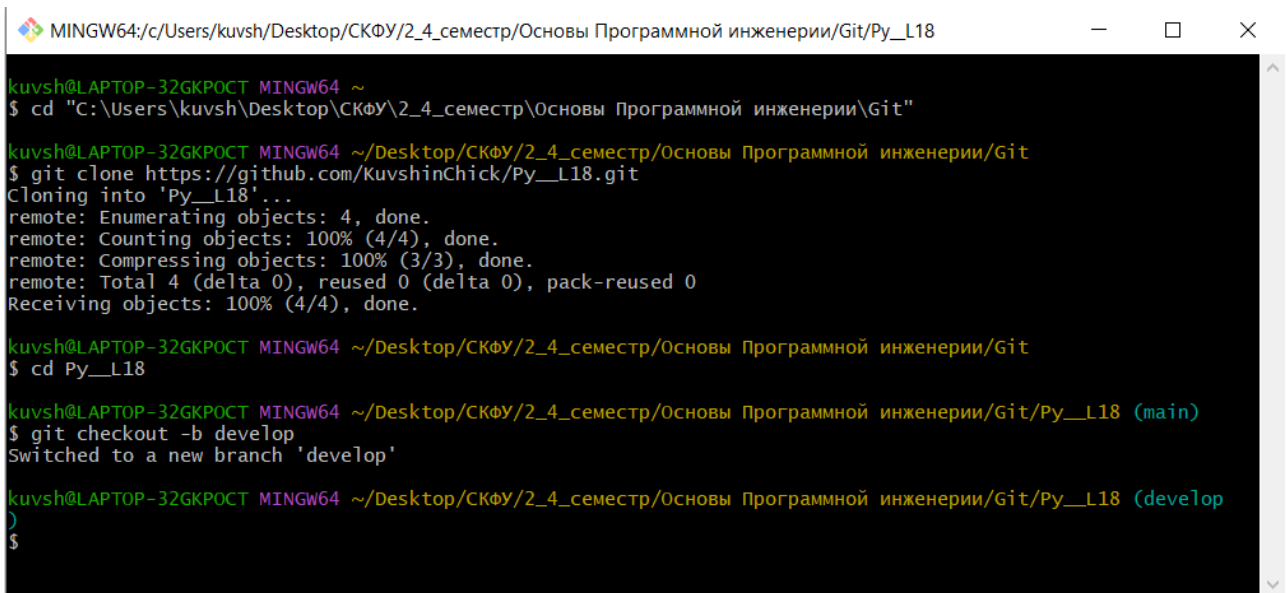
Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2023 г.

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.

Ход работы:

1. Изучить теоретический материал работы.
2. Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и язык программирования Python.
3. Выполните клонирование созданного репозитория.
4. Дополните файл .gitignore необходимыми правилами для работы с IDE PyCharm.
5. Организуйте свой репозиторий в соответствии с моделью ветвления git-flow.



```
MINGW64:/c:/Users/kuvsh/Desktop/СКФУ/2_4_семестр/Основы Программной инженерии/Git/Py_L18
kuvsh@LAPTOP-32GKPOCT MINGW64 ~
$ cd "C:\Users\kuvsh\Desktop\СКФУ\2_4_семестр\Основы Программной инженерии\Git"
kuvsh@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_4_семестр/Основы Программной инженерии/Git
$ git clone https://github.com/KuvshinChick/Py_L18.git
Cloning into 'Py_L18'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (4/4), done.
kuvsh@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_4_семестр/Основы Программной инженерии/Git
$ cd Py_L18
kuvsh@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_4_семестр/Основы Программной инженерии/Git/Py_L18 (main)
$ git checkout -b develop
Switched to a new branch 'develop'
kuvsh@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_4_семестр/Основы Программной инженерии/Git/Py_L18 (develop)
$
```

Рисунок 18.1 – Клонирование репозитория и создание ветки develop

```
MINGW64:/c/Users/kuvsh/Desktop/СКФУ/2_4_семестр/Основы Программной и...
kuvsh@LAPTOP-32GKPOCT MINGW64 ~
$ cd "C:\Users\kuvsh\Desktop\СКФУ\2_4_семестр\Основы Программной инженерии\Git\Py__L18"

kuvsh@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_4_семестр/Основы Программной инженерии/Git/Py__L18 (develop)
$ git status
On branch develop
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   README.md

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .gitignore

no changes added to commit (use "git add" and/or "git commit -a")

kuvsh@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_4_семестр/Основы Программной инженерии/Git/Py__L18 (develop)
$ git add .

kuvsh@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_4_семестр/Основы Программной инженерии/Git/Py__L18 (develop)
$ git commit -m "modified .gitignore & readme"
[develop d4a8d63] modified .gitignore & readme
2 files changed, 132 insertions(+), 1 deletion(-)
create mode 100644 .gitignore

kuvsh@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_4_семестр/Основы Программной инженерии/Git/Py__L18 (develop)
$ |
```

Рисунок 18.2 – Обновление .gitignore и readme

6. Создайте проект PyCharm в папке репозитория.
7. Проработать примеры лабораторной работы.

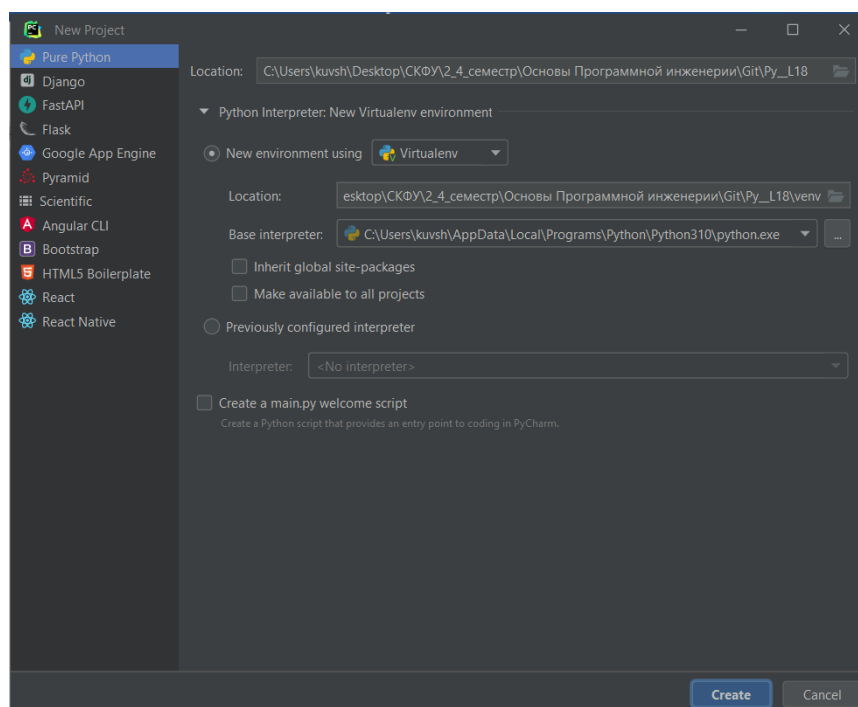


Рисунок 18.3 – Создание проекта и виртуального окружения

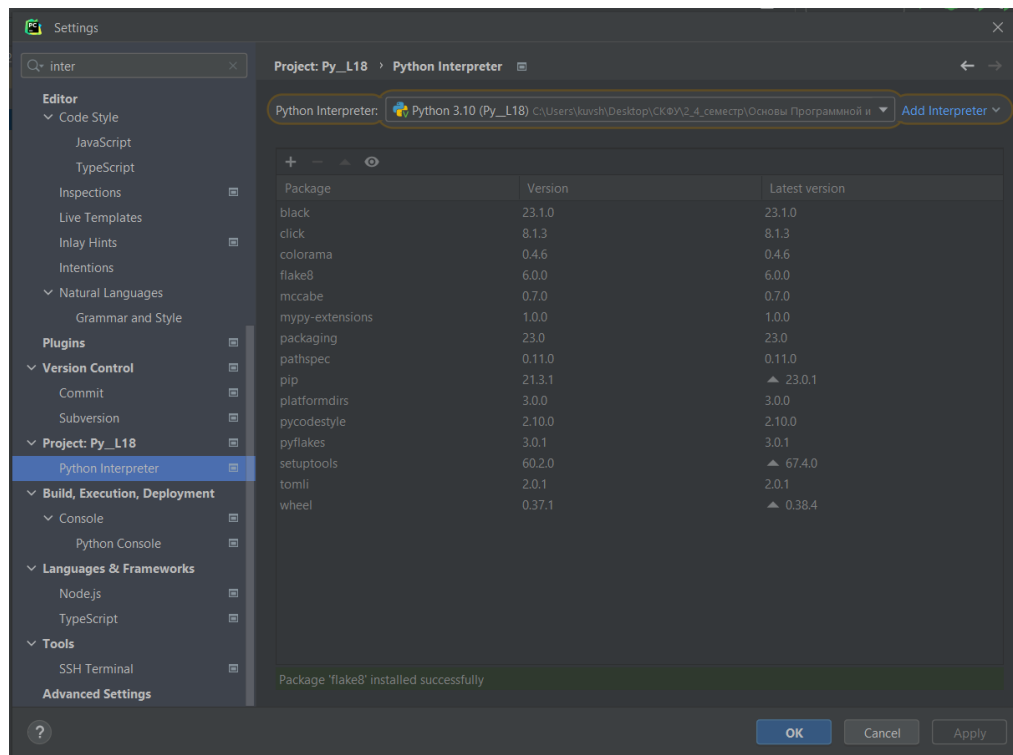


Рисунок 18.4 – Результат установки пакетов black и flake8

Сам Flake8 — инструмент, позволяющий просканировать код проекта и обнаружить в нем стилистические ошибки и нарушения различных конвенций кода на Python. Flake8 умеет работать не только с PEP 8, но и с другими правилами.

Black — это Python-пакет, который автоматически форматирует код, приводя его внешний вид к стандарту PEP 8.

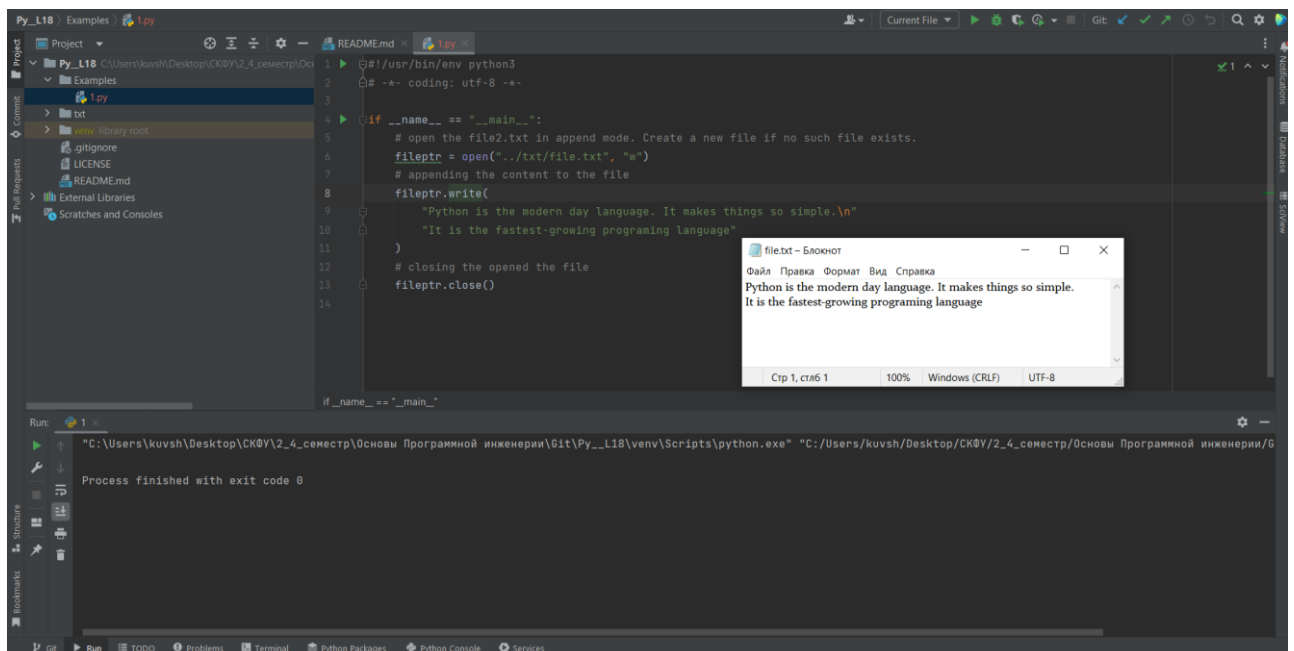


Рисунок 18.5 – Результат проработки первого примера

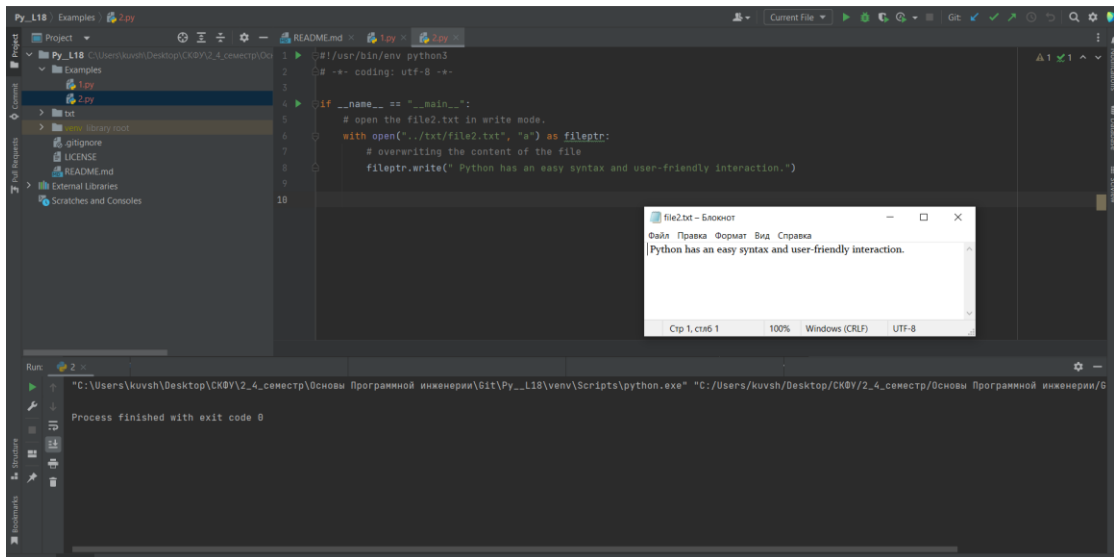


Рисунок 18.6 – Результат проработки второго примера

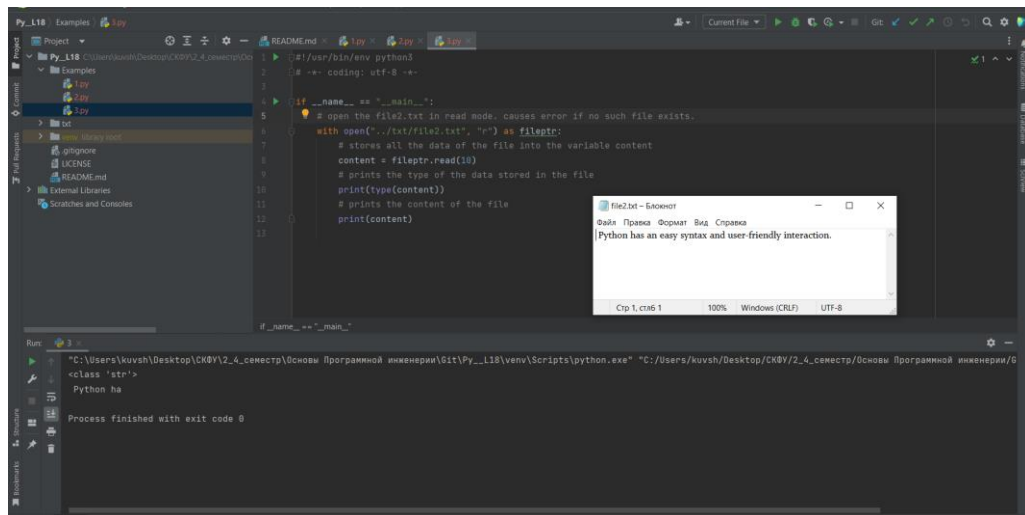


Рисунок 18.7 – Результат проработки третьего примера

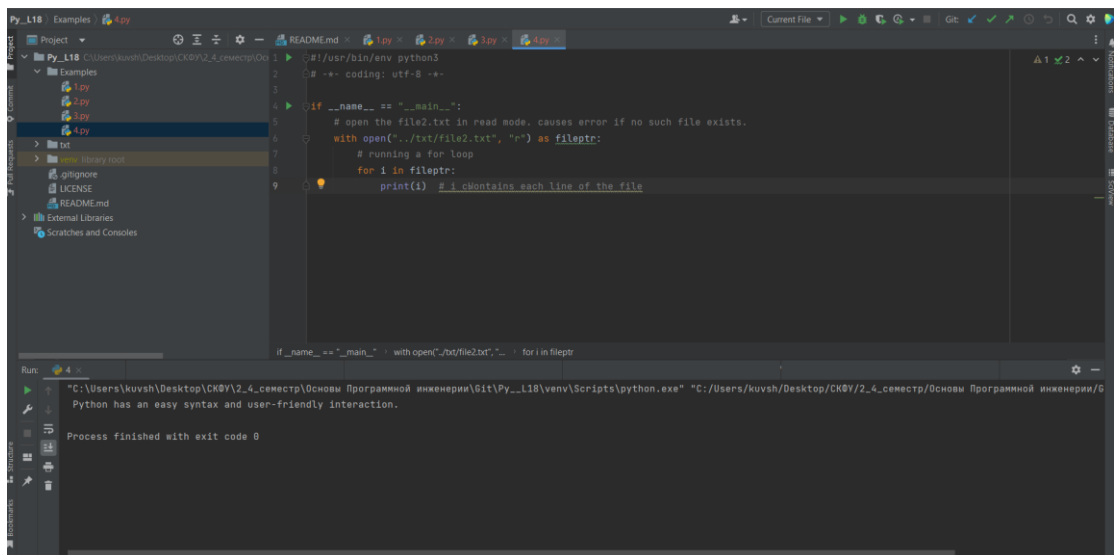


Рисунок 18.8 – Результат проработки четвертого примера

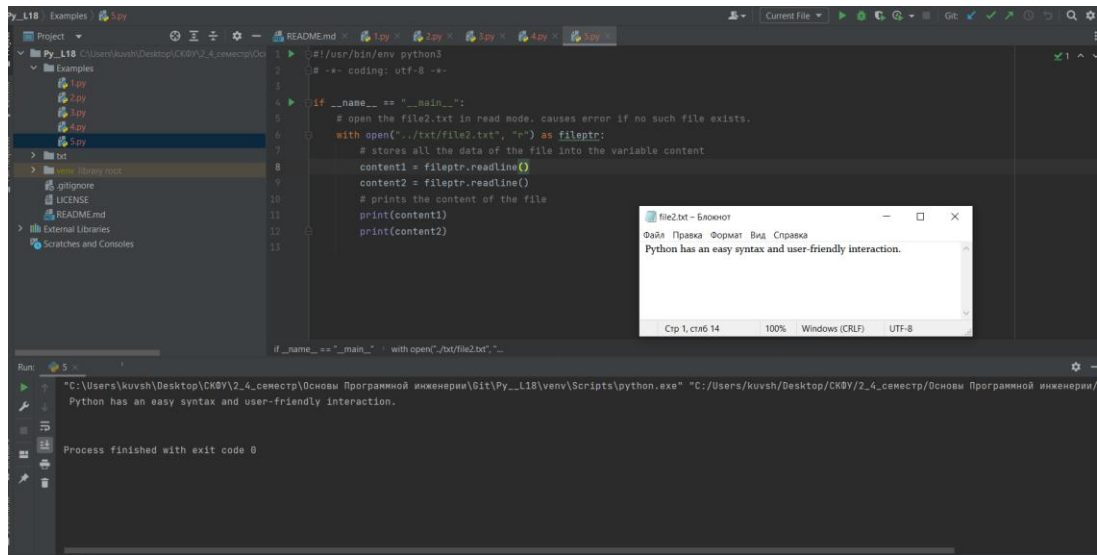


Рисунок 18.9 – Результат проработки пятого примера

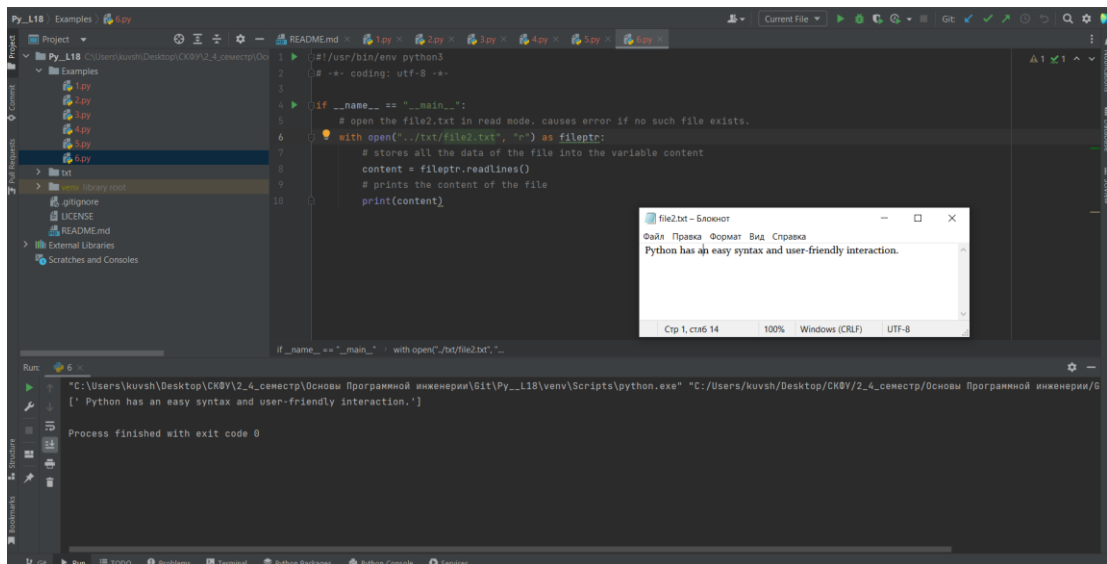


Рисунок 18.10 – Результат проработки шестого примера

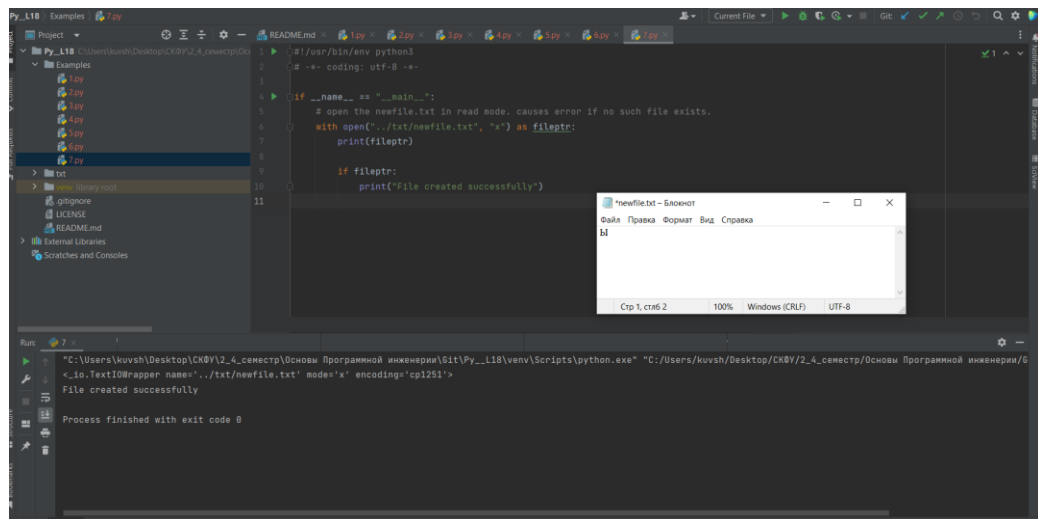


Рисунок 18.11 – Результат проработки седьмого примера

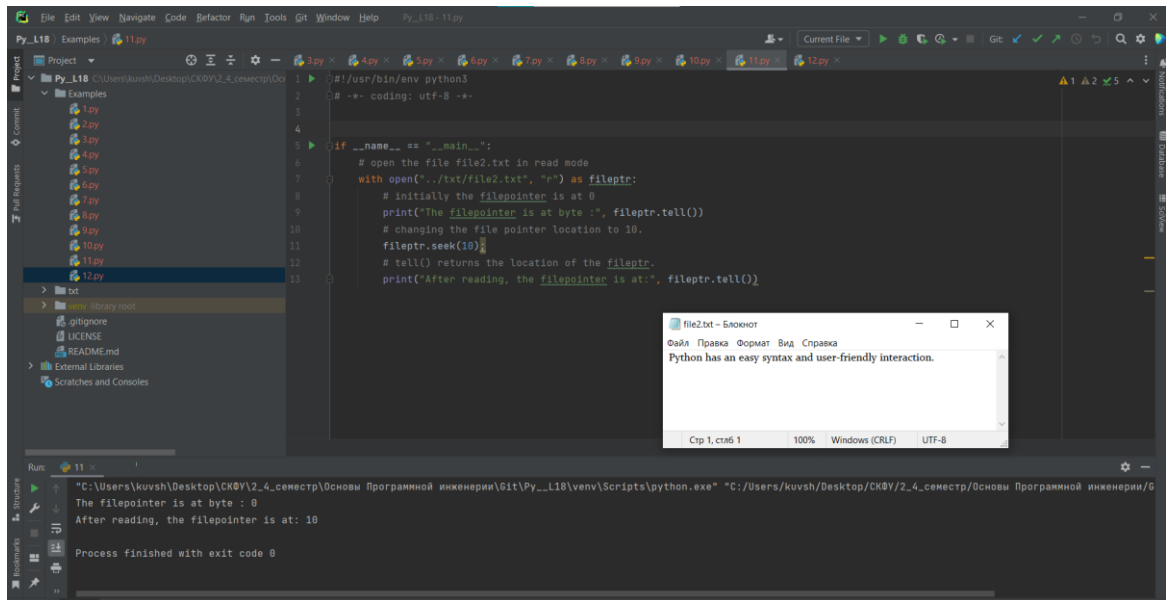


Рисунок 18.15 – Результат проработки одиннадцатого примера

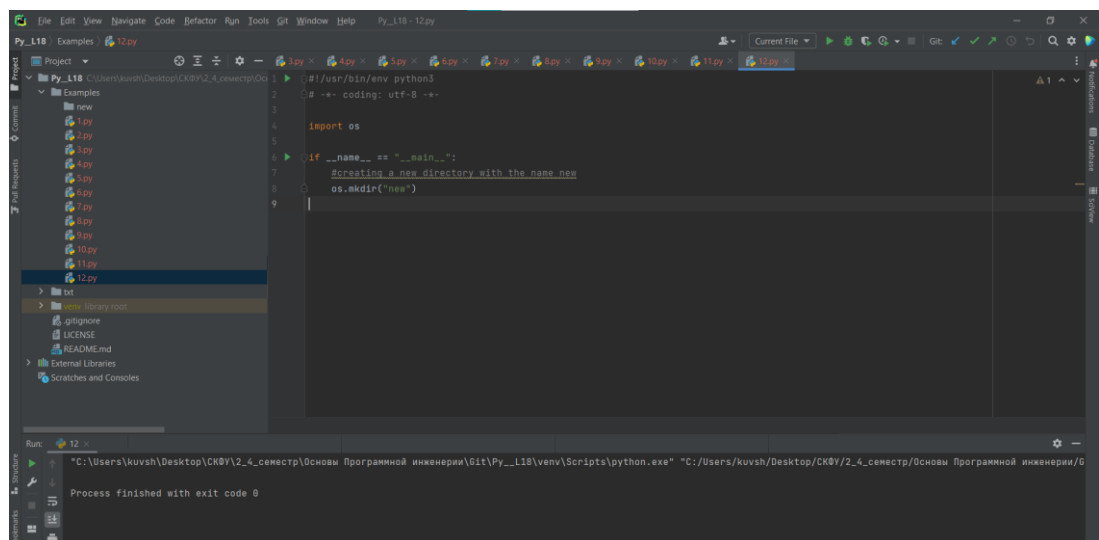


Рисунок 18.16 – Результат проработки двенадцатого примера

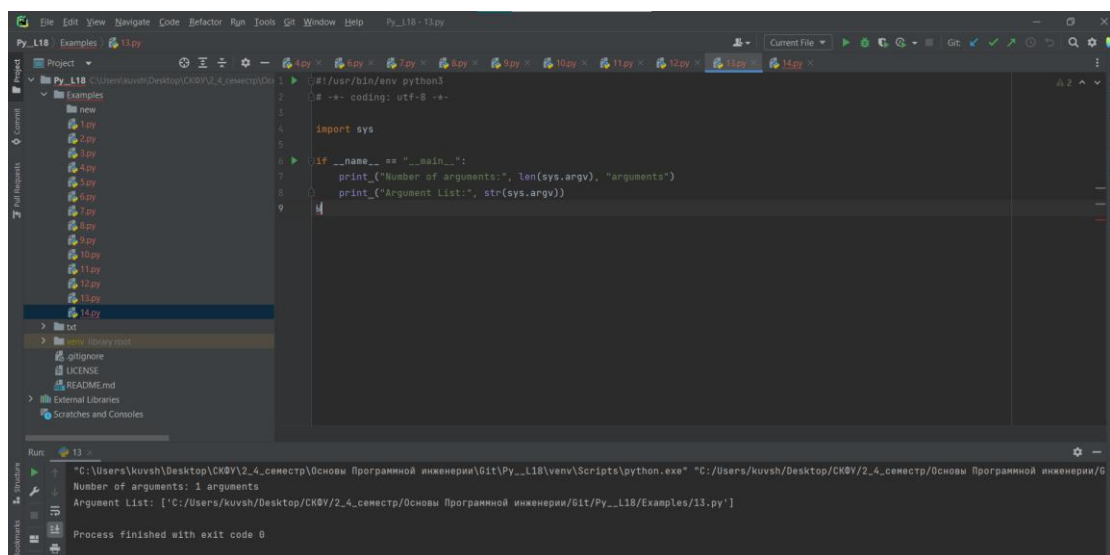


Рисунок 18.17 – Результат проработки тринадцатого примера

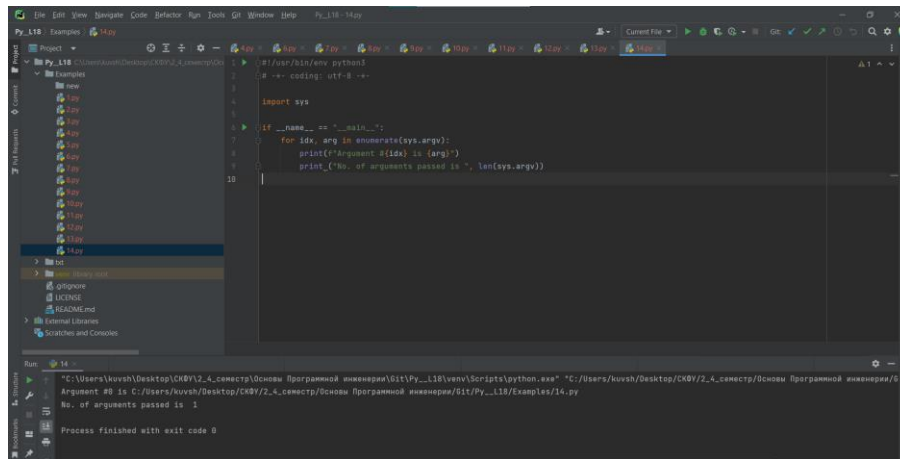


Рисунок 18.18 – Результат проработки четырнадцатого примера

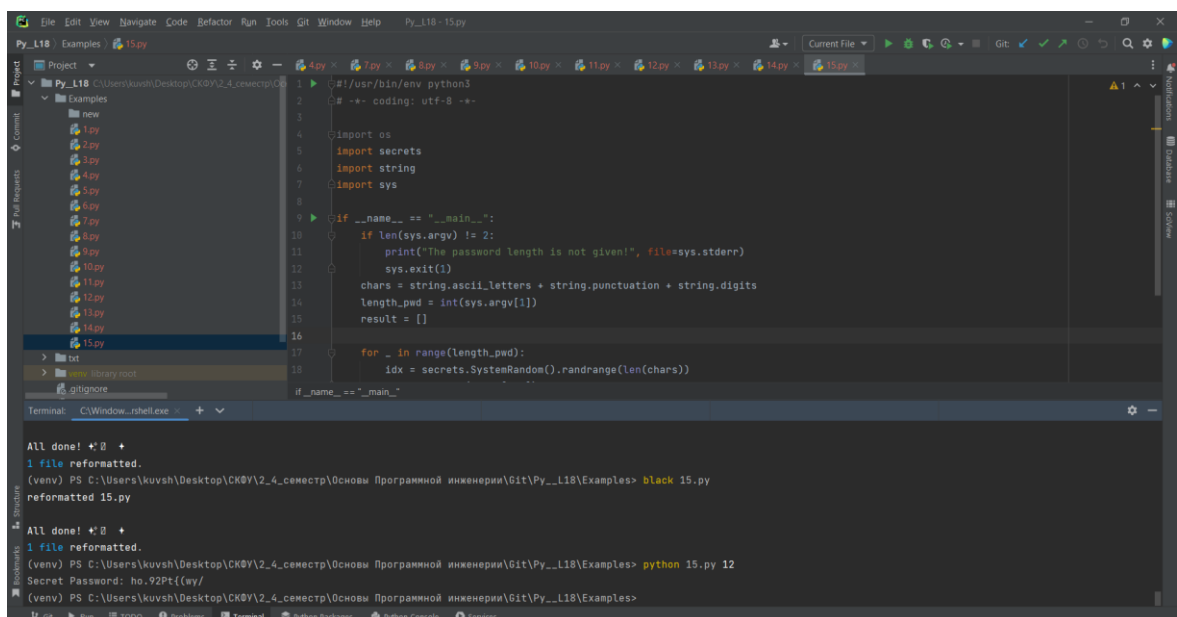


Рисунок 18.19 – Результат проработки пятнадцатого примера

8. Выполнить индивидуальные задания.
9. Зафиксировать изменения в репозитории.

Задание 1

Составить программу с использованием списков и словарей для решения задачи. Номер варианта определяется по согласованию с преподавателем. Исходный файл, из которого выполняется чтение, необходимо также добавить в репозиторий, каждое предложение в файле должно находиться на отдельной строке.

8. Написать программу, которая считывает текст из файла и выводит на экран только цитаты, то есть предложения, заключенные в кавычки.

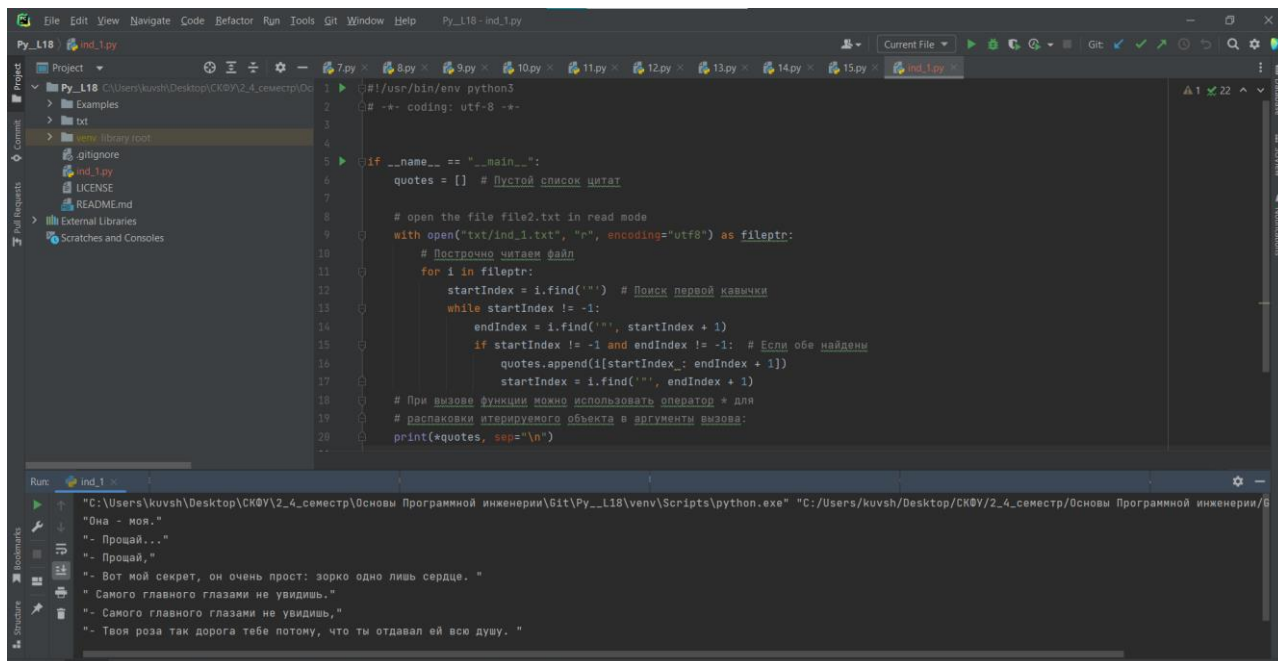


Рисунок 18.20 – Код и результат индивидуального задания

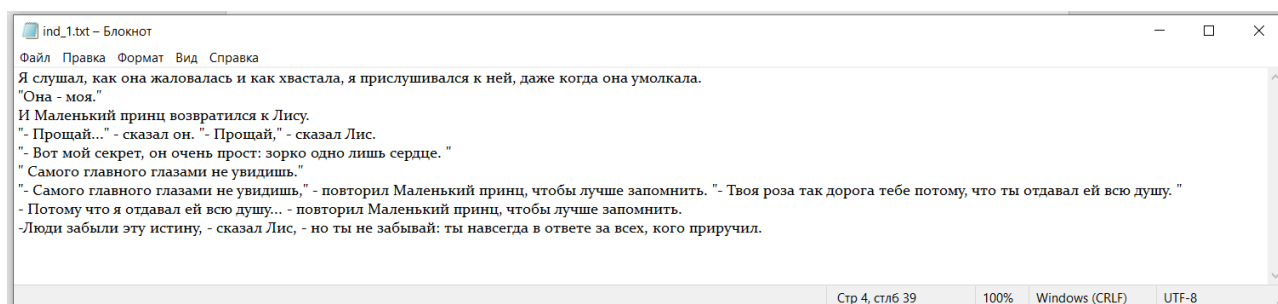


Рисунок 18.21 – Исходный файл индивидуального задания

Задание 2

Составить программу с использованием текстовых файлов. Номер варианта необходимо получить у преподавателя.

- Как вы знаете, в языке Python для создания комментариев в коде используется символ #. Комментарий начинается с этого символа и продолжается до конца строки – без возможности остановить его раньше. В данном упражнении вам предстоит написать программу, которая будет удалять все комментарии из исходного файла с кодом на языке Python. Пройдите по всем строкам в файле на предмет поиска символа #. Обнаружив его, программа должна удалить все содержимое, начиная с этого символа и до конца строки. Для простоты не будем рассматривать ситуации, когда знак решетки встречается в середине строки. Сохраните новое содержимое в созданном файле. Имена файла источника и файла назначения должны быть запрошены у пользователя. Удостоверьтесь в том, что программа корректно обрабатывает возможные ошибки при работе с обоими файлами.

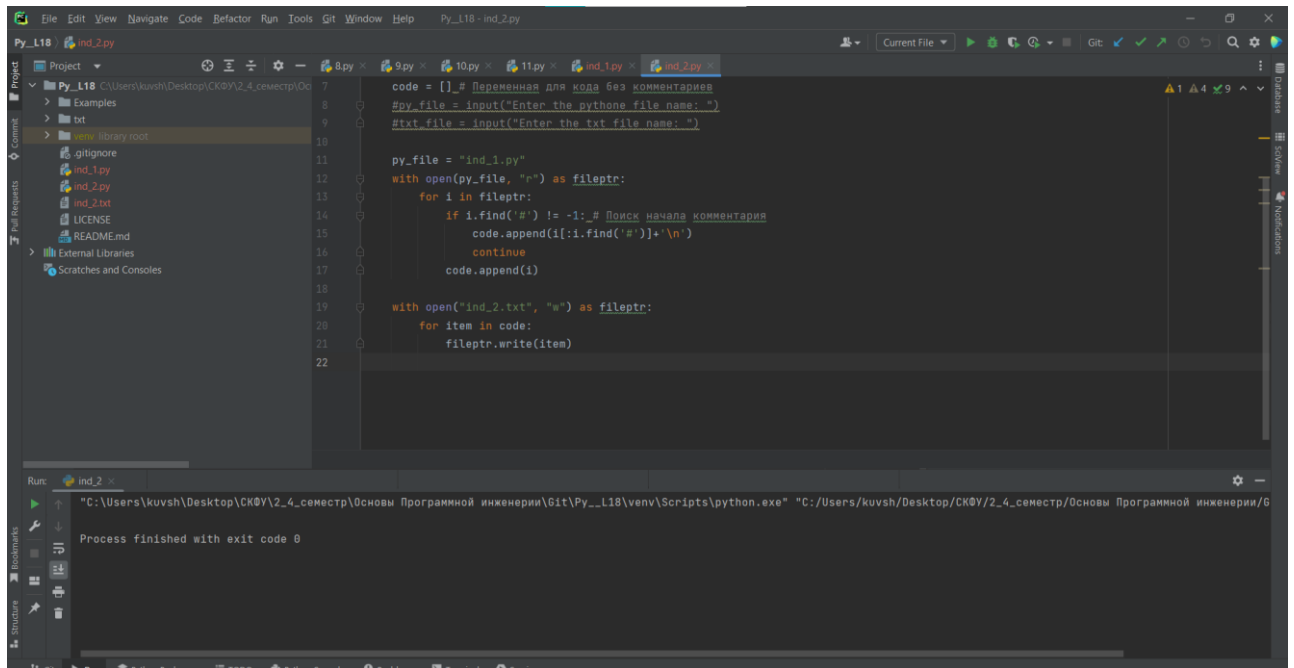


Рисунок 18.22 – Код индивидуального задания

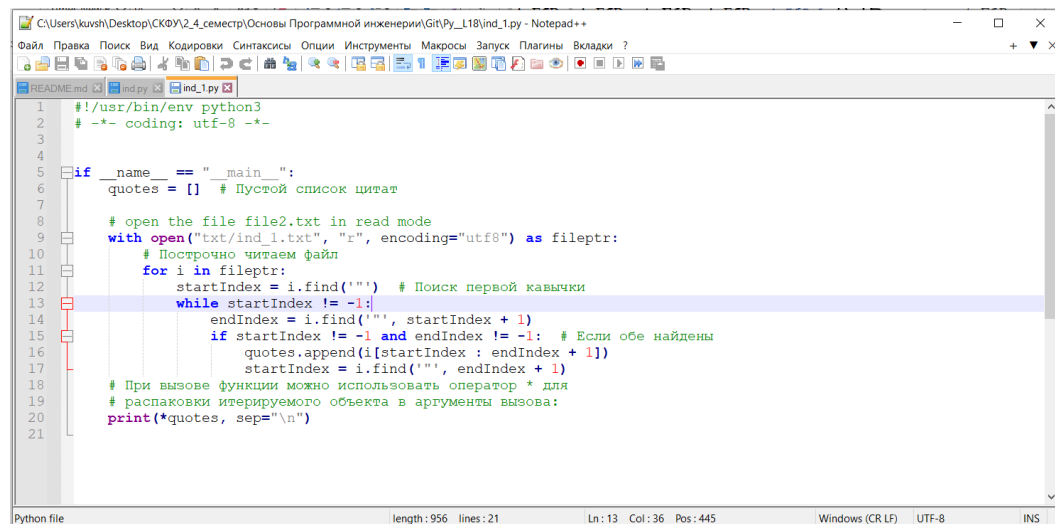


Рисунок 18.23 – Исходный файл индивидуального задания

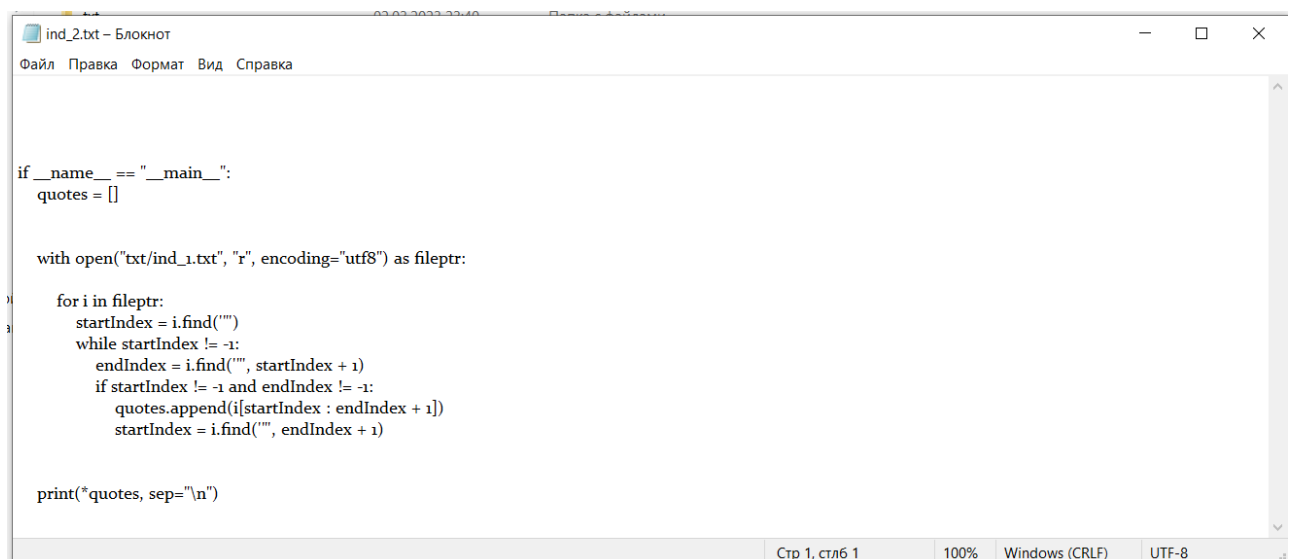
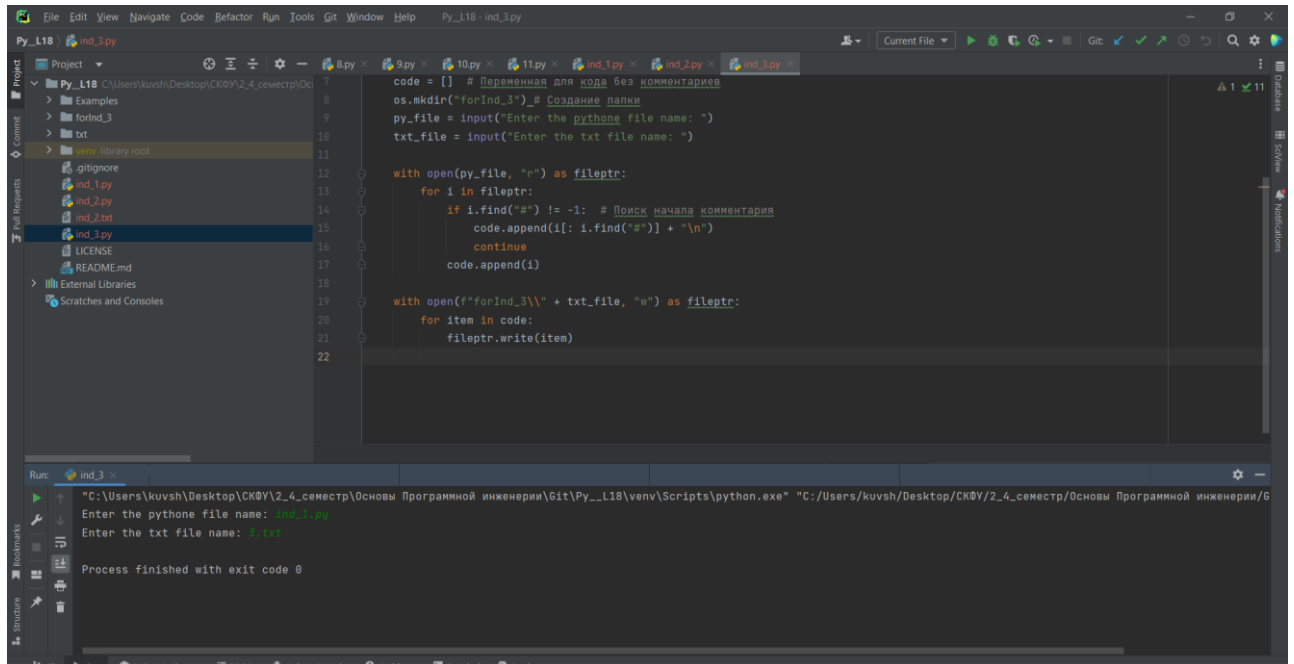


Рисунок 18.24 – Результат индивидуального задания

10. Самостоятельно подберите или придумайте задачу для работы с изученными функциями модуля `os`. Приведите решение этой задачи.

Задание:

Дополнить индивидуальное задание_2. Создать папку, где будет храниться результат работы программы.



```
code = [] # Переменная для кода без комментариев
os.mkdir("forInd_3") # Создание папки
py_file = input("Enter the python file name: ")
txt_file = input("Enter the txt file name: ")

with open(py_file, "r") as fileptr:
    for i in fileptr:
        if i.find("#") != -1: # Поиск начала комментария
            code.append(i[: i.find("#")] + "\n")
            continue
        code.append(i)

with open("forInd_3\\" + txt_file, "w") as fileptr:
    for item in code:
        fileptr.write(item)
```

Run: ind_3

"C:\Users\kuvsh\Desktop\СКФУ\2_4_семестр\Основы Программной инженерии\Git\Py_L18\venv\Scripts\python.exe" "C:\Users\kuvsh\Desktop\СКФУ\2_4_семестр\Основы Программной инженерии\Git\Py_L18\ind_3.py"

Enter the python file name: ind_1.py

Enter the txt file name: 1.txt

Process finished with exit code 0

← → ↕ ↗ > СКФУ > 2_4_семестр > Основы Программной инженерии > Git > Py_L18				
Быстрый доступ				
Рабочий стол	.git	02.03.2023 23:34	Папка с файлами	
Загрузки	.idea	03.03.2023 14:01	Папка с файлами	
Документы	Examples	02.03.2023 23:30	Папка с файлами	
.thumbnails	txt	02.03.2023 23:40	Папка с файлами	
.tmfs	venv	02.03.2023 22:12	Папка с файлами	
Изображения	.gitignore	17.02.2023 11:00	Текстовый докум...	2 КБ
100ANDRO	ind_1.py	03.03.2023 12:00	Python File	1 КБ
BackTrace	ind_2.py	03.03.2023 13:58	Python File	1 КБ
Selfie	ind_2.txt	03.03.2023 13:58	Текстовый докум...	1 КБ
Video	ind_3.py	03.03.2023 14:01	Python File	1 КБ
Py_L18	LICENSE	02.03.2023 20:27	Файл	2 КБ
txt	README.md	02.03.2023 20:37	Файл "MD"	1 КБ

Быстрый доступ				
Рабочий стол	.git	02.03.2023 23:34	Папка с файлами	
Загрузки	.idea	03.03.2023 14:03	Папка с файлами	
Документы	Examples	02.03.2023 23:30	Папка с файлами	
.thumbnails	forInd_3	03.03.2023 14:03	Папка с файлами	
.tmfs	txt	02.03.2023 23:40	Папка с файлами	
Изображения	venv	02.03.2023 22:12	Папка с файлами	
100ANDRO	.gitignore	17.02.2023 11:00	Текстовый докум...	2 КБ
BackTrace	ind_1.py	03.03.2023 12:00	Python File	1 КБ
Selfie	ind_2.py	03.03.2023 13:58	Python File	1 КБ
Video	ind_2.txt	03.03.2023 13:58	Текстовый докум...	1 КБ
Py_L18	ind_3.py	03.03.2023 14:01	Python File	1 КБ
txt	LICENSE	02.03.2023 20:27	Файл	2 КБ
	README.md	02.03.2023 20:37	Файл "MD"	1 КБ

Рисунок 18.25 – Код и результат индивидуального задания

11. Зафиксируйте изменения в репозитории.
12. Добавьте отчет по лабораторной работе в формате PDF в папку doc репозитория. Зафиксируйте изменения.
13. Выполните слияние ветки для разработки с веткой master/main.
14. Отправьте сделанные изменения на сервер GitHub.
15. Отправьте адрес репозитория GitHub на электронный адрес преподавателя.

Контрольные вопросы

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

```
file object = open(<file-name>, <access-mode>, <buffering>)
```

Доступ к файлам можно получить с помощью различных режимов, таких как чтение, запись или добавление. Ниже приведены подробные сведения о режимах доступа для открытия файла.

- **r** – открывает файл в режиме **только для** чтения. Указатель файла существует в начале. Файл по умолчанию открывается в этом режиме, если не передан режим доступа.
- **rb** – открывает файл в двоичном формате **только для** чтения. Указатель файла существует в начале файла.

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

- `w` – только для записи. Он перезаписывает файл, если он существовал ранее, или создает новый, если файл с таким именем не существует. Указатель имеется в начале файла.

- `wb` – открывает файл для записи только в двоичном формате. Перезаписывает файл, если он существует ранее, или создает новый, если файл не существует. Указатель файла существует в начале файла.

3. Как прочитать данные из файла в языке Python?

Чтобы прочитать файл с помощью сценария Python, Python предоставляет метод `read()`. Метод `read()` считывает строку из файла. Он может читать данные как в текстовом, так и в двоичном формате.

Синтаксис метода `read()` приведен ниже.

```
fileobj.read(<count>)
```

4. Как записать данные в файл в языке Python?

Запись файла

Чтобы записать текст в файл, нам нужно открыть файл с помощью метода `open` с одним из следующих режимов доступа.

- `'w'`: он перезапишет файл, если какой-либо файл существует. Указатель файла находится в начале файла.
- `'a'`: добавит существующий файл. Указатель файла находится в конце файла. Он создает новый файл, если файл не существует.

Пример 1.

```
# open the file2.txt in append mode. Create a new file if no such file exists.
fileptr = open("file2.txt", "w")

# appending the content to the file
fileptr.write(
    "Python is the modern day language. It makes things so simple.\n"
    "It is the fastest-growing programing language"
)

# closing the opened the file
fileptr.close()
```

5. Как закрыть файл в языке Python?

Метод `close()`

После того, как все операции будут выполнены с файлом, мы должны закрыть его с помощью нашего скрипта Python, используя метод `close()`. Любая незаписанная информация уничтожается после вызова метода `close()` для файлового объекта.

Мы можем выполнить любую операцию с файлом извне, используя файловую систему, которая в данный момент открыта в Python; поэтому рекомендуется закрыть файл после выполнения всех операций.

Синтаксис использования метода `close()` приведен ниже.

```
fileobject.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 для оборачивания выполнения блока инструкций менеджером контекста. Иногда это более удобная конструкция, чем `try...except...finally`.

Синтаксис конструкции `with ... as`:

```
"with" expression ["as" target] ("," expression ["as" target])* ":"  
suite
```

Теперь по порядку о том, что происходит при выполнении данного блока:

1. Выполняется выражение в конструкции `with ... as`.
2. Загружается специальный метод `__exit__` для дальнейшего использования.
3. Выполняется метод `__enter__`. Если конструкция `with` включает в себя слово `as`, то возвращаемое методом `__enter__` значение записывается в переменную.
4. Выполняется `suite`.
5. Вызывается метод `__exit__`, причём неважно, выполнилось ли `suite` или произошло исключение. В этот метод передаются параметры исключения, если оно произошло, или во всех аргументах значение `None`, если исключения не было.

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

7. Input and Output — Python 3.11.2 documentation

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

[os — Miscellaneous operating system interfaces — Python 3.11.2 documentation](#)