

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 ЛАБОРАТОРНОЙ РАБОТЕ №18
дисциплины «Основы программной инженерии»

Выполнил:
Матвеев Александр Иванович
2 курс, группа ПИЖ-б-о-22-1,
09.03.04 «Программная инженерия»,
направленность (профиль) «Разработка и
сопровождение программного
обеспечения», очная форма обучения

(подпись)

Проверил Воронкин Роман Александрович

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2024 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.

Ход работы.

1. Создание нового репозитория с лицензией MIT.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk ().*

Owner * SashkaHacker / **Repository name *** laba10

✔ laba10 is available.

Great repository names are short and memorable. Need inspiration? How about [fluffy-guide](#) ?

Description (optional)

☒ **Public**
Anyone on the internet can see this repository. You choose who can commit.

☐ **Private**
You choose who can see and commit to this repository.

Initialize this repository with:

☐ **Add a README file**
This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

.gitignore template: Python

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

License: MIT License

A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

ⓘ You are creating a public repository in your personal account.

[Create repository](#)

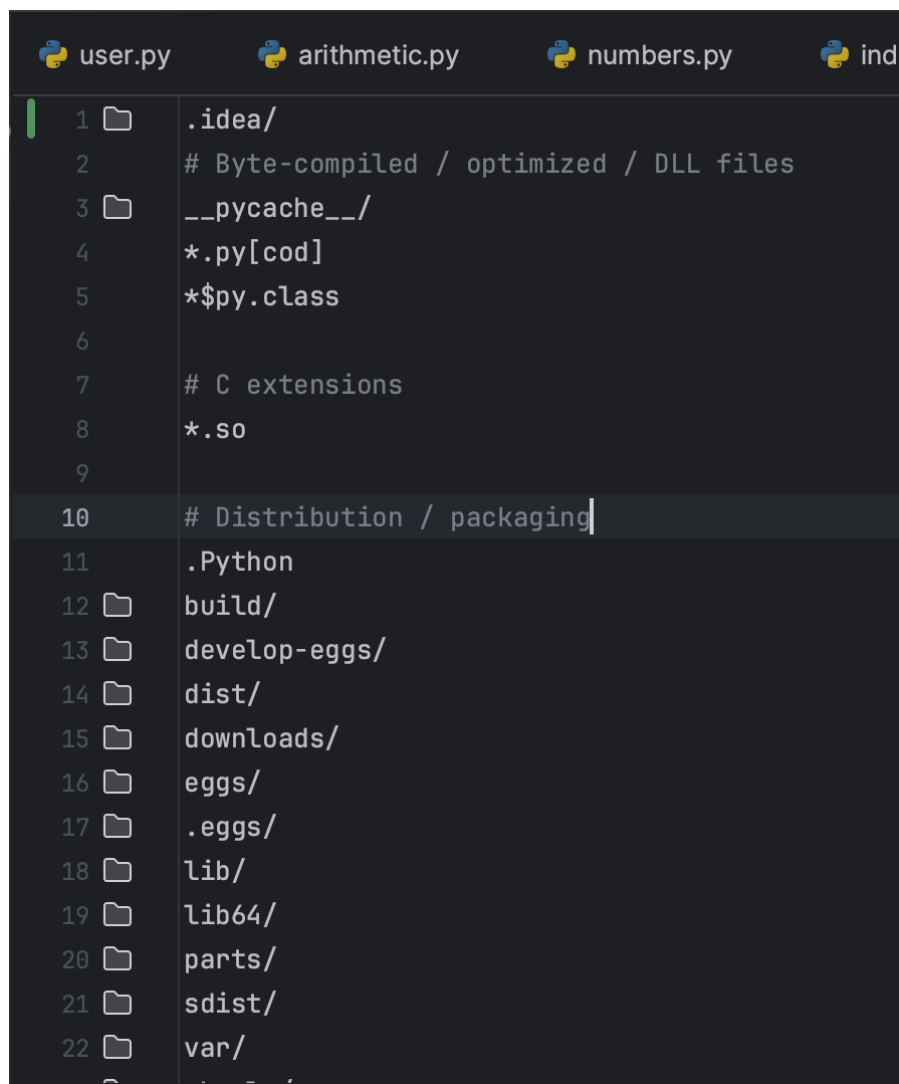
Рисунок 1 – Создание репозитория

2. Клонировал репозиторий на рабочий ПК.

```
[→ GitHub git clone https://github.com/SashkaHacker/laba16.git
Cloning into 'laba16'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (5/5), done.
```

Рисунок 2 – Клонирование репозитория

3. Дополнил файл .gitignore необходимыми инструкциями.



```
user.py arithmetic.py numbers.py ind
1  .idea/
2  # Byte-compiled / optimized / DLL files
3  __pycache__/
4  *.py[cod]
5  *$py.class
6
7  # C extensions
8  *.so
9
10 # Distribution / packaging
11 .Python
12 build/
13 develop-eggs/
14 dist/
15 downloads/
16 eggs/
17 .eggs/
18 lib/
19 lib64/
20 parts/
21 sdist/
22 var/
```

Рисунок 3 – Файл .gitignore

4. Проработка примеров из лабораторной работы.

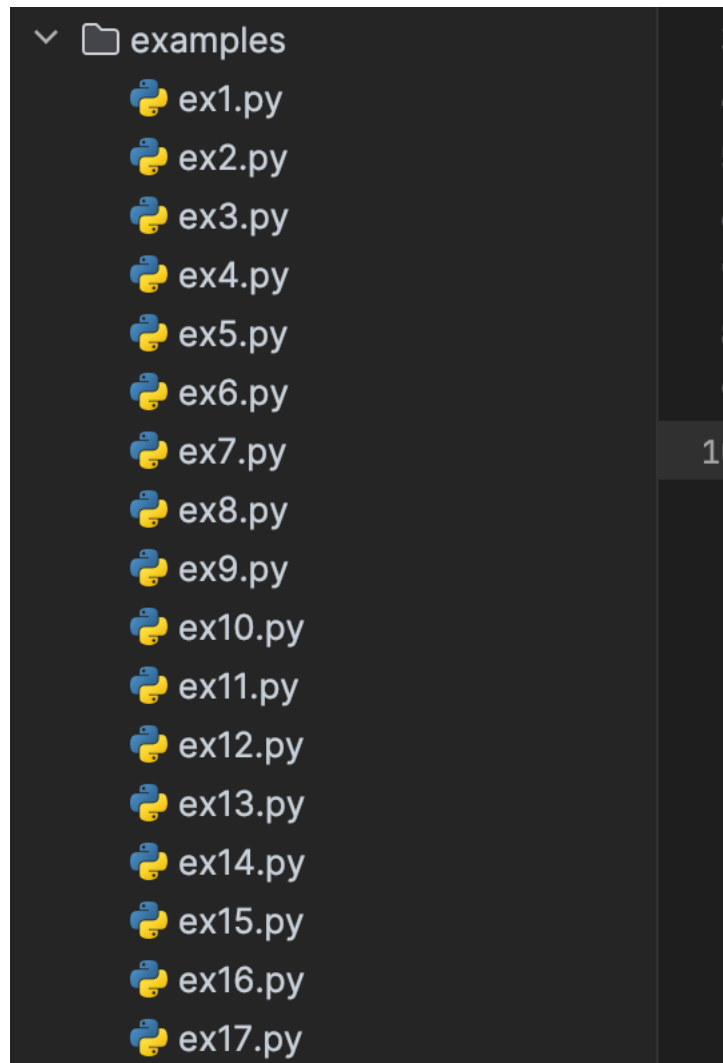


Рисунок 4 – Список проработанных примеров

5. Выполнение 1-го индивидуального задания (вариант - 10). Условие задания: написать программу, которая считывает текст из файла и выводит на экран только строки, не содержащие двузначных чисел.

```

# /usr/bin/env python3
# -*- coding: utf-8 -*-

# вариант - 10

if __name__ == "__main__":
    lst_of_int = [str(i) for i in range(10, 100)]
    try:
        with open("ind1.txt", "r") as f:
            lst_of_lines = f.readlines()
            for i in lst_of_lines:
                words = i.split()
                flag = list(filter(lambda x: x in lst_of_int, words))
                if not flag:
                    print(i[:-1])
    except Exception:
        print("There is no file with that name")

```

Рисунок 5 – Код программы

6. Выполнение индивидуального задания №2 (вариант – 10).

Условие задания: ученикам, желающим запомнить правила написания слов в английском языке, часто напоминают следующее рифмованное одностопное: «I before E except after C» (I перед E, если не после C). Это правило позволяет запомнить, в какой последовательности писать буквы I и E, идущие в слове одна за другой, а именно: буква I должна предшествовать букве E, если непосредственно перед ними не стоит буква C. Если стоит – порядок гласных будет обратным. Примеры слов, на которые действует это правило: believe, chief, fierce, friend, ceiling и receipt. Но есть и исключения из этого правила, и одним из них является слово weird (странный). Напишите программу, которая будет построчно обрабатывать текстовый файл. В каждой строке может присутствовать много слов, а может и не быть ни одного. Слова, в которых буквы E и I не соседствуют друг с другом, обработке подвергать не следует. Если же такое соседство присутствует, необходимо проверить, соответствует ли написание анализируемого слова указанному выше правилу. Создайте и выведите на экран два списка. В первом должны располагаться слова, следующие правилу, а во втором – нарушающие его. При этом списки не должны содержать повторяющиеся слова. Также

отобразите на экране длину каждого списка, чтобы пользователю было понятно, сколько слов в файле не отвечает правилу.

```
# /usr/bin/env python3
# -*- coding: utf-8 -*-

# вариант - 10

import re

if __name__ == "__main__":
    lst_ok = []
    lst_not_ok = []
    try:
        with open("ind2.txt", "r") as f:
            lst_of_string = f.readlines()
            for i in lst_of_string:
                words = re.findall(pattern: r"\b\w*ie\w*\b|\b\w*ei\w*\b", i)
                for word in words:
                    if "cie" in word:
                        lst_not_ok.append(word)
                    elif "ei" in word and "cei" not in word:
                        lst_not_ok.append(word)
                    else:
                        lst_ok.append(word)
    except Exception as e:
        print(e)

    lst_ok, lst_not_ok = list(set(lst_ok)), list(set(lst_not_ok))
    print(f"Слова, соответствующие правилу: {lst_ok}\nДлина" f" списка: {len(lst_ok)}")
    print(
        f"Слова, не соответствующие правилу: {lst_not_ok}\nДлина"
        f" списка: {len(lst_not_ok)}"
    )
```

Рисунок 6 – Код программы

7. Выполните задание: Напишите скрипт на Python, который будет сканировать заданную директорию, считывать все файлы и папки в ней, создавать текстовый файл `directory_contents.txt` с этим списком.

```
# /usr/bin/env python3
# -*- coding: utf-8 -*-

import os

# Напишите скрипт на Python, который будет сканировать заданную
# директорию, считывать все файлы и папки в ней, создавать
# текстовый файл directory_contents.txt с этим списком.

def main(directory_path):
    directory_contents = os.listdir(directory_path)

    with open('directory_contents.txt', 'w') as file:
        for item in directory_contents:
            file.write(f"{item}\n")

    print(
        f"Список содержимого директории '{directory_path}'"
        f" записан в файл 'directory_contents.txt'."
    )

if __name__ == "__main__":
    main(input())
```

Рисунок 7 – Код программы

Контрольные вопросы:

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

Необходимо использовать `open(filename, 'r')`.

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

Необходимо использовать `open(filename, 'w')`

3. Как прочитать данные из файла в языке Python?

Необходимо сначала `file = open(filename, 'r')`, затем `file.readlines()` либо `file.read()`.

4. Как записать данные в файл в языке Python?

Необходимо сначала `file = open(filename, 'w')`, затем `file.write()`.

5. Как закрыть файл в языке Python?

Необходимо `file.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` в Python называется менеджером контекста и предназначена для обертывания блока выполнения операции, которая требует закрытия или освобождения ресурсов после завершения (например, закрытие файла). Эта конструкция гарантирует, что вне зависимости от того, возникло ли исключение во время выполнения блока кода, все необходимые 'очистительные' операции выполнятся автоматически. `with` также может быть использована для управления транзакциями баз данных, освобождения блокировок, возврата ресурсов в пул и др.

7. Изучите самостоятельно документацию Python по работе с файлами.

Какие помимо рассмотренных существуют методы записи/чтения информации из файла?

Метод `.readline()` — читает одну строку из файла; `.readlines()` — читает все строки в файле и возвращает их как список строк; `.writelines()` — записывает последовательность строк в файл;

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

`os.rename()` — переименование файла или директории; `os.remove()` — удаление файла; `os.rmdir()` — удаление пустой директории; `os.mkdir()` — создание новой директории; `os.listdir()` — получение списка файлов и директорий в указанной директории; `os.path.join()` — безопасное объединение одного или нескольких компонентов пути; `os.path.exists()` — проверка существования пути; `os.path.isfile()` — проверка, является ли путь файлом; `os.path.isdir()` — проверка, является ли путь директорией;