

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»

Выполнила:
Ламская Ксения Вячеславовна
2 курс, группа ПИЖ-б-о-22-1,
09.03.04 «Программная инженерия»,
направленность (профиль) «Разработка и
сопровождение программного
обеспечения», очная форма обучения

(подпись)

Доцент кафедры инфокоммуникаций
Воронкин Роман Александрович

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2024 г.

Тема: Лабораторная работа 2.15. 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.


Порядок выполнения работы

1. Создание репозитория GitHub.

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 *	Repository name *
 ksenia-lamskaya ▾	/ 18laba
✔ 18laba is available.	

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

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.](#)

This will set  main as the default branch. Change the default name in your [settings](#).

 You are creating a public repository in your personal account.

Create repository

Рисунок 1.1 – Создание репозитория

2. Запуск Anaconda Powershell Prompt.

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

# вариант - 9

if __name__ == "__main__":

    vowels = {'a', 'e', 'i', 'o', 'u'}

    try:
        with open('./1.txt', 'r', encoding='utf-8') as file:
            text = file.readlines()
            for line in text:
                words = line.split()
                for word in words:
                    word = word.strip('.', ',')
                    if (word[0].lower() in vowels) and (word[-1].lower() in vowels):
                        print(word)

    except FileNotFoundError:
        print(f"There is no file with that name")
```

Рисунок 2.1 – Код программы

```
aliqua
ullamco
ea
```

Рисунок 2.2 – Результат программы

3. Создание пароля посредством генерирования случайных символов может обернуться сложностью в запоминании полученной относительно надежной последовательности. Некоторые системы создания паролей рекомендуют сцеплять вместе два слова на английском языке, тем самым упрощая запоминание заветного ряда символов – правда, в ущерб его надежности. Напишите программу, которая будет открывать файл со списком слов, случайным образом выбирать два из них и сцеплять вместе для получения итогового пароля. При создании пароля исходите из следующего требования: он должен состоять минимум из восьми символов и максимум из

десяти, а каждое из используемых слов должно быть длиной хотя бы в три буквы. Кроме того, сделайте заглавными первые буквы обоих слов, чтобы легко можно было понять, где заканчивается одно и начинается другое. По завершении процесса полученный пароль должен быть отображен на экране..

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

# вариант - 9

import random

def generate_password():
    with open("words.txt", "r") as file:
        words = file.read().split(',')

    clean_words = [word.strip() for word in words]

    word1 = random.choice(clean_words)
    word2 = random.choice(clean_words)

    while len(word1) < 3 or len(word2) < 3:
        word1 = random.choice(clean_words)
        word2 = random.choice(clean_words)

    word1 = word1.capitalize()
    word2 = word2.capitalize()

    password = word1 + word2

    return password

if __name__ == "__main__":
    password = generate_password()
    print(f"Generated Password: {password}")
```

Рисунок 3.1 – Командная строка

Generated Password: LionNest

Рисунок 3.2 – Результат программы

Контрольные вопросы:

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

Для открытия файла только для чтения в Python используется функция `open()` с параметром `'r'`. Например: `file = open('file.txt', 'r')`

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

Для открытия файла только для записи в Python используется функция `open()` с параметром `'w'`. Например: `file = open('file.txt', 'w')`

3. Как прочитать данные из файла в языке Python?

Для чтения данных из файла в Python можно использовать методы `read()`, `readline()`, или `readlines()`. Например: `content = file.read()` # Чтение всего содержимого файла

4. Как записать данные в файл в языке Python?

Для записи данных в файл в Python можно использовать методы `write()`. Например: `file.write("Some text to write into the file")`

5. Как закрыть файл в языке Python?

Для закрытия файла в Python используется метод `close()`. Например: `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` используется для обеспечения автоматического закрытия ресурсов после выполнения блока кода. В контексте работы с файлами, она гарантирует закрытие файла после завершения работы с ним.

Это также может быть использовано для работы с другими ресурсами, требующими явного закрытия, например, сокетами или базами данных.

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

Помимо методов чтения и записи, существуют также методы для управления позицией в файле (`seek()`), проверки конца файла (`tell()`), изменения имени файла (`rename()`), удаления файла (`remove()`), создания директории (`mkdir()`), удаления директории (`rmdir()`), проверки существования файла или директории (`exists()`), и многие другие.

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

Некоторые другие функции модуля `os` для работы с файловой системой в Python включают `listdir()` для получения списка файлов в директории, `stat()` для получения информации о файле, `chdir()` для изменения текущей директории, `getcwd()` для получения текущей директории, `unlink()` для удаления файла, `chmod()` для изменения прав доступа к файлу, `utime()` для изменения временных меток файла и т. д.