

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ  
ФЕДЕРАЦИИ**  
**Федеральное государственное автономное  
образовательное учреждение высшего образования  
«СЕВЕРОКАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

**Кафедра инфокоммуникаций**

**Институт цифрового развития**

**ОТЧЁТ**

**по лабораторной работе № 10**

Дисциплина: «Программирование на Python»

Тема: «Работа с файлами в языке Python»

Выполнил: студент 2 курса

группы ИВТ-б-о-21-1

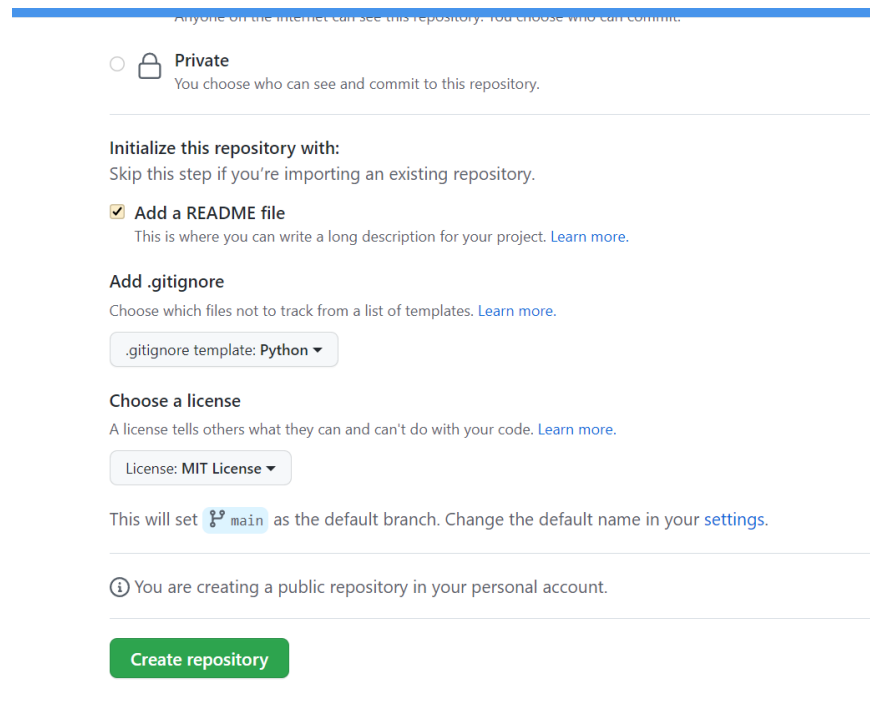
Уланбекова Айканыш Уланбековна

Ставрополь 2022


**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.

### Порядок выполнения работы:

1. Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и язык программирования Python.



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:  
Skip this step if you're importing an existing repository.

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

**Add .gitignore**  
Choose which files not to track from a list of templates. [Learn more.](#)  
.gitignore template: **Python** ▾

**Choose a license**  
A license tells others what they can and can't do with your code. [Learn more.](#)  
License: **MIT License** ▾

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. Создание репозитория

2. Выполните клонирование созданного репозитория.

```
C:\Users\User>cd C:\Users\User\Desktop\2 кырс Python\lab 18\2.15
C:\Users\User\Desktop\2 кырс Python\lab 18\2.15>git clone https://github.com/aikanyshkaukanbekova/2.15.git
Cloning into '2.15'...
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.
C:\Users\User\Desktop\2 кырс Python\lab 18\2.15>
```

Рисунок 2. Клонирование репозитория

4. Организуйте свой репозиторий в соответствии с моделью ветвления git-flow.

```

C:\Users\User\Desktop\2 кypc Python\lab 10\lab-10>git flow init
which branch should be used for bringing forth production releases?
- main
Branch name for production releases: [main]
Branch name for "next release" development: [develop]

How to name your supporting branch prefixes?
Feature branches? [feature/]
Bugfix branches? [bugfix/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? []
Hooks and filters directory? [C:/Users/User/Desktop/2 кypc Python/lab 10/lab-10/.git/hooks]
C:\Users\User\Desktop\2 кypc Python\lab 10\lab-10>

```

Рисунок 4. Организован модель ветвления git flow

5.Проработайте примеры лабораторной работы. Создайте для них отдельный модуль языка Python. Зафиксируйте изменения в репозитории.

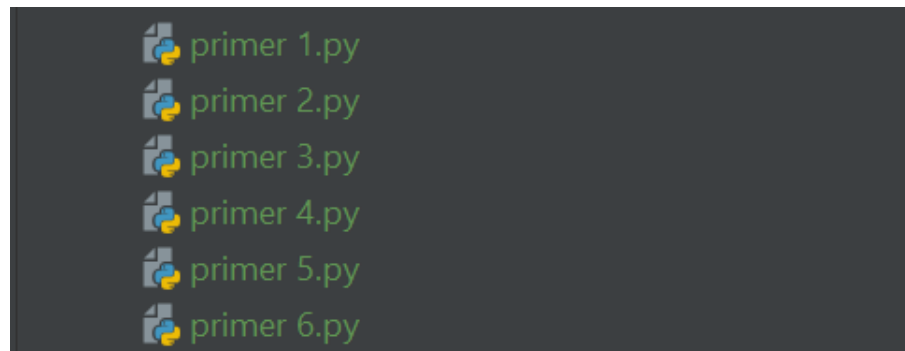


Рисунок 5. Примеры лаб работы

## 6. Индивидуальное задание

### Вариант 9.

**Задание 1.** Написать программу, которая считывает английский текст из файла и выводит на экран слова текста, начинающиеся и оканчивающиеся на гласные буквы.

```

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

def split(text):
    vowels = 'ueioay'
    for word in text:
        words = []
        for prword in vowels:
            if word[0] == prword:
                words = str(vowels)
        for prword in words:
            if word.endswith(prword):
                print(word)

```

```
if __name__ == "__main__":
    with open("text1.txt", "r", encoding="utf-8") as fileptr:
        sentences = fileptr.readlines()
        text1 = str(sentences).lower().split()
        print(split(text1))
```

Рисунки 6. Выполненное индивидуальное задание

```
"C:\Users\User\Desktop\2 курс Python\lab 18\2.15\2.15\indiv\venv\Scripts\python.exe"
a
unicode
one
one-byte
None
```

Рисунок 7. Результат

**Задание 2.** 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 -*-

if __name__ == "__main__":
    with open("oc.txt", "w") as fileptr:
        slovar = {'5': 'A', '4': 'B', '3': 'C', '2': 'D', '1': 'F'}
        while True:
            i = input()

            ii = {value: key for key, value in slovar.items()}
            if i in slovar.keys():
                fileptr.write(f'{i} = {slovar[i]}\n')
                print(f'{i} = {slovar[i]}')
            elif i in ii.keys():
                fileptr.write(f'{i} = {ii[i]}\n')
                print(f'{i} = {ii[i]}')
            elif i == '':
```

```

        break
    else:
        fileptr.write(f'{i} = Данное значение является
недопустимым\n')
    print(f'{i} = Данное значение является недопустимым')
print(fileptr)

```

Рисунок 8. Выполненное индивидуальное задание

```

"C:\Users\User\Desktop\2 курс Python\lab 18\2.15\2.15\indiv\venv\Scripts\python.exe"
A
A = 5
3
3 = C
f
f = Данное значение является недопустимым
|

```

Рисунок 9. Результат выполненной работы

8. Сделала коммит, выполнил слияние с веткой main, и запустил изменения в уд. репозиторий.

```

C:\Users\User\Desktop\2 курс Python\lab 18\2.15\2.15>git add .
C:\Users\User\Desktop\2 курс Python\lab 18\2.15\2.15>git commit -m "new"
[main 2f7c055] new
16 files changed, 391 insertions(+)
create mode 100644 indiv/indiv 1.py
create mode 100644 indiv/indiv 2.py
create mode 100644 indiv/oc.txt
create mode 100644 indiv/text1.txt
create mode 100644 primery/file2.txt
create mode 100644 primery/file3.txt
create mode 100644 primery/newfile.txt
create mode 100644 primery/newfile2.txt
create mode 100644 primery/primer 1.py
create mode 100644 primery/primer 2.py
create mode 100644 primery/primer 3.py
create mode 100644 primery/primer 4.py
create mode 100644 primery/primer 5.py
create mode 100644 primery/primer 6.py
create mode 100644 primery/requirements.txt
create mode 100644 primery/text.txt
C:\Users\User\Desktop\2 курс Python\lab 18\2.15\2.15>git push
Enumerating objects: 19, done.
Counting objects: 100% (19/19), done.
Delta compression using up to 8 threads
Compressing objects: 100% (17/17), done.
Writing objects: 100% (18/18), 6.07 KiB | 1.52 MiB/s, done.
Total 18 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/aikanyshkaukanbekova/2.15.git
3a82530..2f7c055 main -> main
C:\Users\User\Desktop\2 курс Python\lab 18\2.15\2.15>_

```

Рисунок 8. Сохранения

**Ответы на контрольные вопросы:**

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

`file object = open( , ) access-mode = 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.

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

`file object = open( , ) access-mode = 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.

## **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. Python упрощает чтение файла построчно с помощью метода `readline()`. Метод `readline()` читает строки файла с самого начала, т. е. если мы используем его два раза, мы можем получить первые две строки файла.

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

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.

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

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.

## **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. Используется в сценарии, когда пара

операторов должна выполняться с блоком кода между ними. Преимущество использования оператора `with` заключается в том, что он обеспечивает гарантию закрытия файла независимо от того, как закрывается вложенный блок.

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

Функция `write()` Функция `write()` используется для записи в файлы Python, открытые в режиме записи. С помощью метода `writelines()` можно записать в файл итерируемую последовательность.

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

Функцию `os.replace()` можно использовать для перемещения файлов или каталогов. Функция `os.listdir()` возвращает список, который содержит имена файлов в папке. `os.walk()` — это генератор дерева каталогов. Он будет перебирать все переданные составляющие. Метод `os.path.join()` был использован для объединения текущего пути с именем файла/папки. Для получения информации о файле в ОС используется функция `os.stat()`, которая выполняет системный вызов `stat()` по выбранному пути.

**Вывод:** были приобретены навыки по работе с текстовыми файлами при написании программ с помощью языка программирования Python версии 3.x.