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 №2.15

Дисциплина: «Программирование на Python»

Тема: «Работа с файлами в языке Python»

Выполнил: студент 2 курса

группы ИВТ-б-о-21-1

Кочкаров Умар Ахматович

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

Ход работы:

1. Создать общедоступный репозиторий с лицензией МІТ и языком Python.

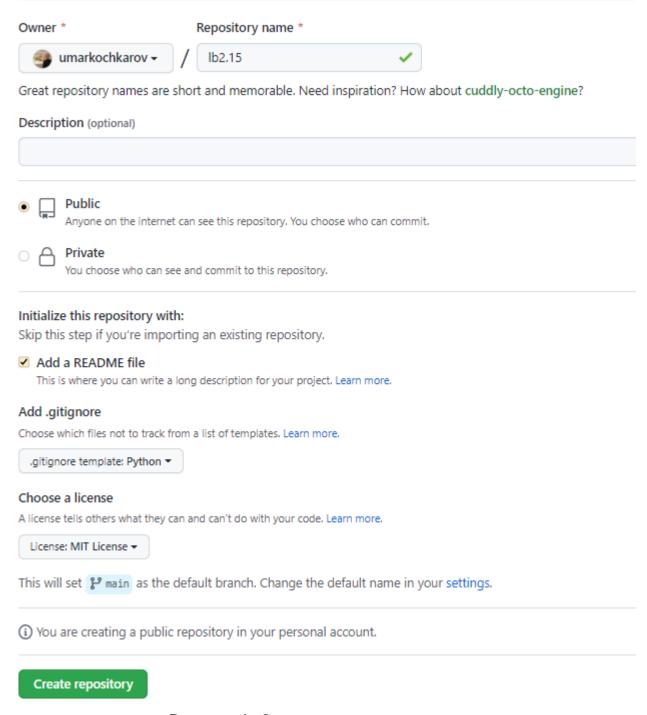


Рисунок 1. Создание репозитория

2. Клонировать репозиторий на ПК:

```
erken@LAPTOP-ESTC60GF MINGW64 ~/Desktop/python/Лаб2.13

$ git clone https://github.com/umarkochkarov/lb2.13.git
Cloning into 'lb2.13'...
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. Организовать репозиторий в соответствии с моделью ветвления git-flow.

```
S 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/erken/Desktop/python/Ла62.15/]b2.15/.git/

hooks]
```

Рисунок 3. Организация репозитория в соответствии с моделью git-flow

4. Проработка примеров из лабораторной работы:

Индивидуальные задания:

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

```
if __name__ == "__main__":
           with open("ind1.txt", "r", encoding="utf-8") as f:
               sentences = f.readlines()
           joinon = False
           str = []
10
           for sentence in sentences:
               for i in sentence:
                       if k % 2 == 1:
                           joinon = True
                       else:
                           del str[0]
                           print("".join(str))
                           joinon = False
                            str = []
                   if joinon == True:
                       str.append(i)
PC
Run:
    🍦 ind1
        C:\Users\erken\AppData\Local\Programs\Python\Python310\python.exe C:/l
        текст в кавычках
```

Рисунок 4. Индивидуальное 1

2. 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.

```
C:\Users\erken\Desktop\python\Лa62.15\lb2.15\ind>python ind2.py ind2.txt
Aa : 68
Бб : 13
Вв : 50
Гг : 14
Дд : 22
Ee : 79
Ëë : 7
Жж : 4
33 : 21
Ии : 102
Йй : 26
Кк : 36
Лл : 54
Им : 31
Нн : 67
0o : 115
Пп : 32
Pp : 39
c : 46
  : 69
  : 36
  : 5
  : 54
Шш : 3
Цщ : 3
ы : 14
Ьь : 16
  : 5
Ээ
  : 2
Яя : 14
```

Рисунок 5. Индивидуальное 2

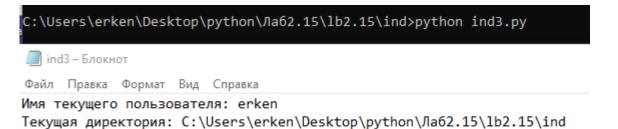


Рисунок 6. Индивидуальное 3

Контрольные вопросы:

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

Чтобы открыть файл для чтения, мы используем режим r. Для чтения мы воспользуемся функцией read(size), если параметр size не указан, функция вернет нам всю строку. file = open("text.txt", 'r', encoding = 'utf-8').

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

В Python открытие файлов выполняется с помощью функции open(), которой передается два аргумента - имя файла и режим. Файл может быть открыт в режиме чтения, записи, добавления.

3. Как прочитать данные из файла в языке Python?

Чтение данных из файла осуществляется с помощью методов read(размер) и readline(). Метод read(размер) считывает из файла определенное количество символов, переданное в качестве аргумента.

4. Как записать данные в файл в языке Python?

Запись данных в файл. Записать данные в файл можно с помощью метода write().

5. Как закрыть файл в языке Python?

После того, как мы открыли файл, и выполнили все нужные операции, нам необходимо его закрыть. Для закрытия файла используется функция close().

6. Изучите самостоятельно работу конструкции with ... as. Каково ее назначение в языке?

Конструкция with ... as используется для оборачивания выполненияблока инструкций менеджером контекста. Если в конструкции

with - as было несколько выражений, то это эквивалентно нескольким вложенным конструкциям

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

Один из самых распространенных способов вывести данные в Python – это напечатать их в консоли. Если вы находитесь на этапе изучения языка, такой способ является основным для того, чтобы быстро просмотреть результат свой работы

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

os.chdir(path) - смена текущей директории.

os.chmod (path, mode, *, dir_fd=None, follow_symlinks=True) - смена прав доступа к объекту (mode - восьмеричное число).

os.chown (path, uid, gid, *, dir_fd=None, follow_symlinks=True) - меняет id владельца и группы (Unix).

os.getcwd() - текущая рабочая директория.

os.link (src, dst, *, src_dir_fd=None, dst_dir_fd=None,

follow_symlinks=True) - создает жесткую ссылку.

os.listdir (path=".") - список файлов и директорий в папке.

os.mkdir (path, mode=0o777, *, dir_fd=None) - создаѐт директорию.

OSError, если директория существует.

os.makedirs (path, mode=0o777, exist_ok=False) - создает директорию, создавая при этом промежуточные директории.

os.remove (path, *, dir_fd=None) - удаляет путь к файлу.

os.rename (src, dst, *, src_dir_fd=None, dst_dir_fd=None) - переименовывает файл или директорию из src в dst.

os.renames (old, new) - переименовывает old в new, создавая промежуточные директории.

os.replace (src, dst, *, src_dir_fd=None, dst_dir_fd=None) - переименовывает из src в dst с принудительной заменой.

os.rmdir (path, *, dir_fd=None) - удаляет пустую директорию.

os.removedirs (path) - удаляет директорию, затем пытается удалить родительские директории, и удаляет их рекурсивно, пока они пусты.

os.sync() - записывает все данные на диск (Unix).

os.truncate (path, length) - обрезает файл до длины length.

os.utime (path, times=None, *, ns=None, dir_fd=None,

follow_symlinks=True) - модификация времени последнего доступа и изменения файла. Либо times - кортеж (время доступа в секундах, время изменения в секундах), либо ns - кортеж (время доступа в наносекундах, время изменения в наносекундах).

os.walk (top, topdown=True, onerror = None, followlinks=False) – генерация имèн файлов в дереве каталогов, сверху вниз (если topdown равен True), либо снизу вверх (если False). Для каждого каталога функция walk возвращает кортеж (путь к каталогу, список каталогов, список файлов).