

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

Кафедра инфокоммуникаций

Лабораторная работа 2.15

Работа с файлами в языке Python

Выполнил студент группы ИВТ-б-о-20-1

Симанский М.Ю « » _____ 20__ г.

Подпись студента _____

Работа защищена « » _____ 20__ г.

Проверил Воронкин Р.А. _____

(подпись)

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.

Вариант 18(1)

Написать программу, которая считывает из текстового файла три предложения и выводит их в обратном порядке.

Исходный файл(рис. 1).

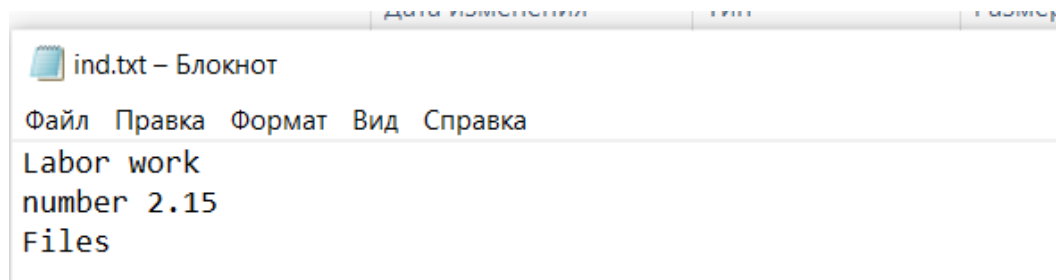


Рисунок 1 – Текст в файле

Результат выполнения работы (рис. 2).

```
C:\Users\гас-е\anaconda2\python.exe -c C:/user
Files
number 2.15

Labor work

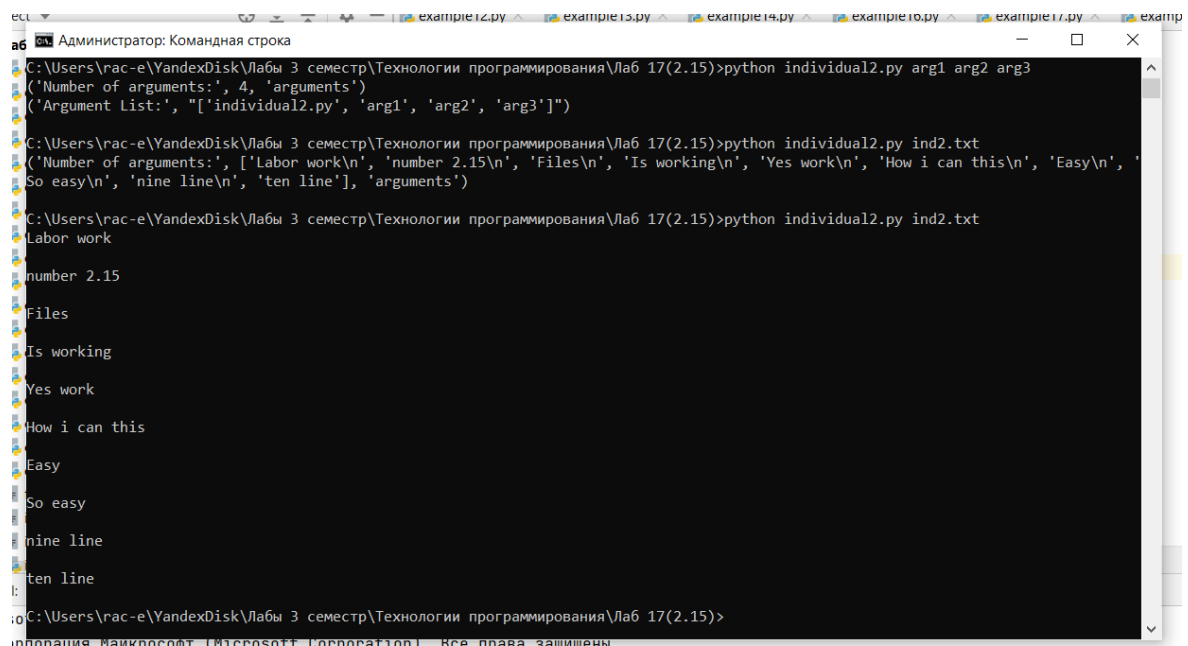
Process finished with exit code 0
```

Рисунок 2 – Результат выполнения

Задание 2

В операционных системах на базе Unix обычно присутствует утилита с названием head. Она выводит первые десять строк содержимого файла, имя которого передается в качестве аргумента командной строки. Напишите программу на Python, имитирующую поведение этой утилиты. Если файла, указанного пользователем, не существует, или не задан аргумент командной строки, необходимо вывести соответствующее сообщение об ошибке.

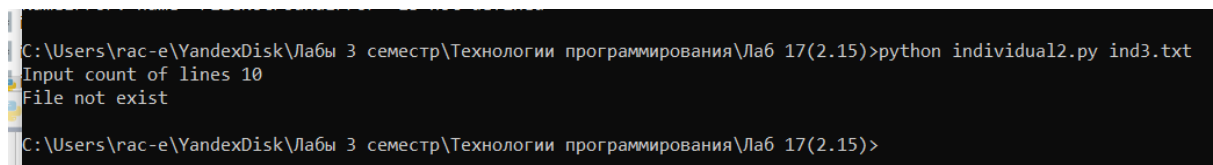
Выдача данных (рис. 3).



```
ab Администратор: Командная строка
C:\Users\rac-e\YandexDisk\Лабы 3 семестр\Технологии программирования\Лаб 17(2.15)>python individual2.py arg1 arg2 arg3
('Number of arguments:', 4, 'arguments')
('Argument List:', "['individual2.py', 'arg1', 'arg2', 'arg3']")
C:\Users\rac-e\YandexDisk\Лабы 3 семестр\Технологии программирования\Лаб 17(2.15)>python individual2.py ind2.txt
('Number of arguments:', 2, 'arguments')
('Argument List:', "['individual2.py', 'ind2.txt']")
C:\Users\rac-e\YandexDisk\Лабы 3 семестр\Технологии программирования\Лаб 17(2.15)>python individual2.py ind2.txt
Labor work
number 2.15
Files
Is working
Yes work
How i can this
Easy
So easy
nine line
ten line
C:\Users\rac-e\YandexDisk\Лабы 3 семестр\Технологии программирования\Лаб 17(2.15)>
```

Рисунок 3 – Первые 10 строк

Если файла не существует появится ошибка (рис. 4).



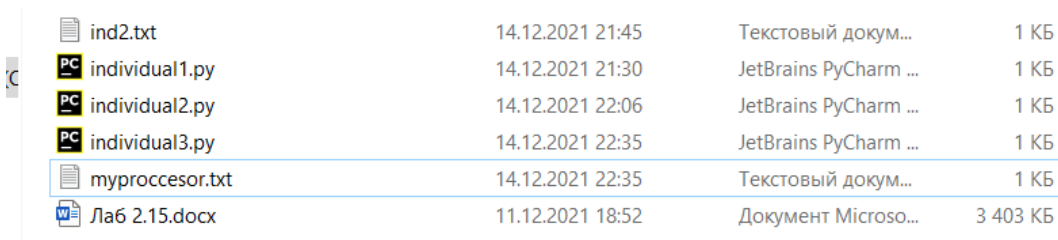
```
C:\Users\rac-e\YandexDisk\Лабы 3 семестр\Технологии программирования\Лаб 17(2.15)>python individual2.py ind3.txt
Input count of lines 10
File not exist
C:\Users\rac-e\YandexDisk\Лабы 3 семестр\Технологии программирования\Лаб 17(2.15)>
```

Рисунок 4 – Ошибка

Задание 3

Создать файл новый текстовый файл, затем требуется узнать данные архитектуры процессора и количество его ядер, а потом записать их в созданный файл, изменить имя файла на myprocessor.txt если его уже не существует, если существует выдать сообщение.

После выполнения создался файл (рис. 5).



ind2.txt	14.12.2021 21:45	Текстовый докум...	1 КБ
individual1.py	14.12.2021 21:30	JetBrains PyCharm ...	1 КБ
individual2.py	14.12.2021 22:06	JetBrains PyCharm ...	1 КБ
individual3.py	14.12.2021 22:35	JetBrains PyCharm ...	1 КБ
myprocessor.txt	14.12.2021 22:35	Текстовый докум...	1 КБ
Ла6 2.15.docx	11.12.2021 18:52	Документ Microso...	3 403 КБ

Рисунок 5 – Файл создан

Содержание файла следующее (рис. 6).

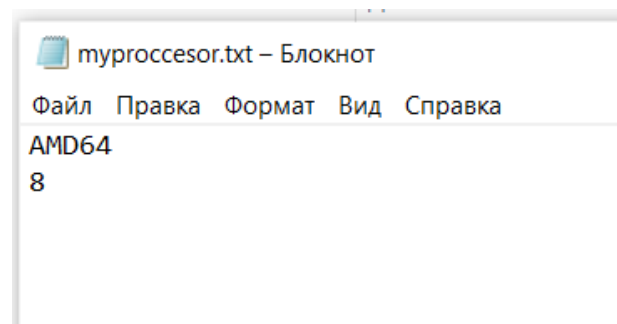


Рисунок 6 – Данные процессора

Если файл не существует получим ошибку(рис. 7).

```
C:\Users\rac-e\anaconda3\python.exe
Файл уже существует

Process finished with exit code 0
```

Рисунок 7 – Ошибка

Вывод: в результате выполнения работы были приобретены навыки по работе с текстовыми файлами при написании программ с помощью языка программирования Python версии 3.x, изучены основные методы модуля os для работы с файловой системой.

Ответы на контрольные вопросы

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

Что бы 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 выполнения блока 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 своей работы

8. Какие существуют, помимо рассмотренных, функции модуля `os` для 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)` - создаёт жёсткую ссылку.

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

`os.mkdir(path, mode=0o777, *, dir_fd=None)` - создаёт директорию.
`OSError`, если директория существует.

`os.makedirs(path, mode=0o777, exist_ok=False)` - создаёт 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.rename(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 имён 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).