

## Задание №2

### Шаг 1. Команды Linux

`ls` – список файлов в текущем каталоге

`cd` – переход в другой каталог

`cd /` - переход в корневой каталог

`cd /home` – переход в каталог

`cd home`

`cd bsk23-01-student-1` – переход в личную папку пользователя «`bsk23-01-student-1`». В командной строке можно набрать «`cd bsk`» и нажать клавишу `tab`. Система автоматически заполнит оставшееся название каталога

`pwd` – отображает наше текущее положение

`/home/bsk23-01-student-1`

Стрелки вверх/вниз – перемещение по истории команд (чтобы не набирать заново).



Другие базовые команды в Linux (`uname`, `man`, `mkdir`, `touch`, `cat`, `nano`, `cp`,

`mv`, `rm`, `df`).



### Шаг 2. Установка пакетов

`apt` – менеджер программ/пакетов

`apt list` – показывает все установленные пакеты

`grep` – фильтр строк

`|` - передача вывода команды другой команде

`apt list | grep python` – показывает все пакеты с «`python`» в названии

`sudo apt install mc` – установка `midnight commander`  
подтверждаем установку «`Y`»

### Шаг 3. Midnight Commander

`mc` – файловый менеджер

**F7** – создание новой папки. Создаём папку «python»



Другие базовые команды MC и Nano.



#### Шаг 4. Первый скрипт Python

Создаем новый файл.

Можно через MC нажатием **Shift+F4** – создание нового файла.

При первом редактировании MC спросит какой редактор использовать. Выбрали редактор Nano.

Можно в командной строке командой nano с именем файла.

В созданном файле пишем:

```
print("Hello!")
```

**Ctrl+X** (сохранить и выйти) затем **Y**

Указываем или подтверждаем название файла «hello.py»

Если мы были в MC то **F10** – выход из mc

**python3** – запустили Python Shell

**exit()** – вышли из Python Shell

*так же можно использовать сочетание клавиш **Ctrl+D***

Python скрипт можно запустить сразу, если ввести:

```
python3 <название>.py
```

Скрипт выполнил команду и написал нам:

```
Hello!
```

*Теперь вы можете написать в резюме, что имеете опыт написания скрипта Python на*

*виртуальном сервере Linux Debian*



#### Шаг 5. Простой цикл

Создаем новый файл **for.py**

В файле пишем:

```
for i in range(1, 10):  
    print(i)
```



Тут более подробное описание цикла for в Python



### Шаг 6. Чтение файла

Создайте новый файл с именем **file.txt**. Напишите в нем любые 5 или больше строк.

Создайте новый файл скрипта Python с именем **fileread.py**

В скриптах Python можно писать комментарии. Всё, что написано после знака «#» считается комментарием

```
# открываем файл на чтение  
f = open('file.txt', 'r')  
# выводим содержимое файла на экран  
print(*f)  
# закрываем открытый файл  
f.close
```

В этом скрипте мы открыли файл **file.txt** (функция open) на чтение (параметр 'r' в функции open) и вывели на экран всё содержимое файла (\*f). Затем закрыли файл

### Шаг 7. Чтение одной строки файла

Вместо вывода всего файла считаем одну строку (функция readline) запишем её в переменную l и выведем её на экран (print(l))

```
# открываем файл на чтение  
f = open('file.txt', 'r')  
# читаем строку файла записываем в переменную l  
l = f.readline()  
# выводим переменную l на экран  
print(l)  
# закрываем открытый файл  
f.close
```

### Шаг 8. Чтение 3х строк из файла

Теперь считаем из файла 3 строки с помощью цикла for

```
# открываем файл на чтение  
f = open('file.txt', 'r')  
# запускаем цикл 3 раза
```

```
for i in range(1,4):
    # читаем строку файла записываем в переменную l
    l = f.readline()
    # выводим переменную l на экран
    print(l)
# закрываем открытый файл
f.close
```

### Шаг 9. Чтение и изменение 3х строк из файла

Считаем их файла 3 строки и изменим их, добавив в конце каждой строки !!!!

```
# открываем файл на чтение
f = open('file.txt', 'r')
# запускаем цикл 3 раза
for i in range(1,4):
    # читаем строку файла, добавляем «!» записываем в переменную l
    l = f.readline()[::-1] + '!!!!'
    # выводим переменную l на экран
    print(l)
# закрываем открытый файл
f.close
```

[::-1] - метод удаляет последний символ в строке (у нас это перевод строки)

### Шаг 10. Чтение 3х строк из файла, изменение и запись в другой файл

Выведем результат не на экран, а в другой файл

```
# открываем файл на чтение
f = open('file.txt', 'r')
# открываем файл на запись
fw = open('newfile.txt', 'w')
# запускаем цикл 3 раза
for i in range(1,4):
    # читаем строку файла, добавляем «!» записываем в переменную l
    l = f.readline()[::-1] + '!!!!'
    # выводим переменную l, но не на экран, а в файл fw
    print(l, file = fw)
# закрываем открытые файлы
f.close
fw.close
```

После выполнения этого скрипта вы не увидите результата на экране. Но можете увидеть новый файл «newfile.txt», в котором будут измененные строки

**Шаг 11. Чтение файла полностью**

```
# открываем файл на чтение
f = open('file.txt', 'r')
# открываем второй файл на запись
fw = open('newfile.txt', 'w')
# выполняем пока есть возможность считать строку из файла
while True:
    # считываем строку
    l = f.readline()
    # прерываем цикл, если строка пустая
    if not l:
        break
    # добавляем к строке «!»
    ll = l[:-1] + '!!!!'
    # выводим строку
    print(ll)
# закрываем открытые файлы
f.close
fw.close
```

**Fix Me!**

Тут более подробное описание цикла While в Python

**Fix Me!**

From:

<https://sibgu-itlab-wiki.data-pool.ru/> - **SIBGU-ITLAB-WIKI**

Permanent link:

[https://sibgu-itlab-wiki.data-pool.ru/zadanie\\_2?rev=1709526915](https://sibgu-itlab-wiki.data-pool.ru/zadanie_2?rev=1709526915)Last update: **2024/03/04 04:35**