

ЛАБОРАТОРНАЯ РАБОТА № 1.1

Командный язык и скрипты LINUX ч.1.

Цель. Закрепить на практике основные команды для работы с файлами и каталогами.

Краткие теоретические сведения.

Сперва научимся двигаться по папкам, создавать и переименовывать папки. Linux всегда создает домашний каталог каждому пользователю. Кроме того, в Linux имеется корневой каталог Root, где хранятся системные файлы и каталоги. С Root мы ничего делать не будем. Для выдачи команд используем консольное окно (терминал). Следующая команда

\$ pwd

отображает текущий путь.

Поскольку мы находимся в домашнем каталоге пользователь, создадим рабочую папку с именем work:

\$ mkdir имя

Имя задает имя каталога. В нашем случае это будет work. Теперь надо перейти в этот каталог. Для этого воспользуемся командой:

\$ cd work (например)

Для просмотра содержимого текущей папки используем команду ls.

Содержимое текущего каталога можно просмотреть с помощью команды **ls**

\$ ls

Показать все файлы, даже скрытые, можно с помощью флагов команды

\$ ls -a

Показать файлы с подробным их описанием (дата создания, владелец, права доступа) можно по команде

\$ ls -l (это буква эль-малая)

Можно создать текстовый файл в текущей папке. Для создания файла (пустого) используем команду touch:

\$ touch имя_файла

Чтобы занести в файл текстовые строки (в режиме добавления) воспользуемся командой:

\$ echo "hie, once again" >> file1

\$ echo "hie, once again" > file1 – перезаписывает содержимое **file1**.

Содержимое файла можно выдать с помощью команды:

\$ cat file1

Наконец перенести файл в другое место можно по команде:

\$ mv имя1 имя2

Копирование файла осуществляется командой:

\$ cp имя1 имя2

Если нужно скопировать файл в другой директорию, то **имя2** в команде копирования задает имя директория.

Можно записать несколько строк в текстовый файл таким образом:

\$ cat> file1

Hello students,

What do you want?

В конце ввода следует нажать комбинацию **ctrl-d**.

Очистить содержимое файла проще всего

\$ > file1

Для удаления файла используем

\$ rm имя

Теперь обратимся к скриптам. Скрипт представляет последовательность команд Linux. Скрипты будем писать в текстовом редакторе. Для вызова текстового редактора набираем команду

\$ gedit

Напишем простейший скрипт:

```
#!/bin/bash
read x
read y
let z="x+y"
echo $z
```

Скрипт всегда должен начинаться со строки **#!/bin/bash** или **#!/bin/sh**. Здесь **bash** и **sh** - это оболочки Linux, оболочка воспринимает команды с консоли и выполняет их. Заметим, что **bash** – более усовершенствованная версия. Команды **read x** (**read y**) читают значения переменных **x**, **y**. Команда **let z = "x+y"** выполняет сложение, а **echo \$z** выводит результат на экран.

Сохраните этот файл без всяких расширений, например, как **script_a**. Выполните его. Здесь есть нюанс. Во-первых, скрипт нужно сделать выполняемым файлом. Этого мы добиваемся с помощью команды

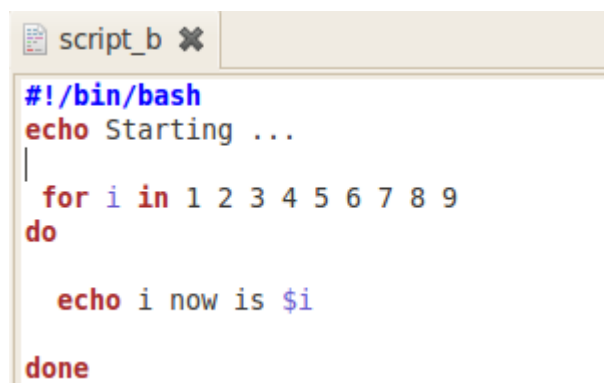
```
$ chmod +x script_a
```

Теперь выполняем скрипт

```
$ ./script_a
```

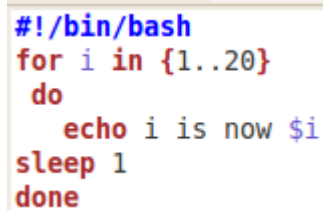
Комментарий в скриптах задается через символ **#**.

Теперь введем цикл **for**. Синтаксис цикла **for** приведен на скриншотах ниже.



```
#!/bin/bash
echo Starting ...
|
  for i in 1 2 3 4 5 6 7 8 9
do
    echo i now is $i
done
```

Или



```
#!/bin/bash
for i in {1..20}
do
    echo i is now $i
    sleep 1
done
```

Здесь \$i – это значение переменной. Команда **sleep 1** осуществляет паузу в 1 секунду.

Цикл **while** представлен на примере ниже:

```
#!/bin/bash
i=100;
while [ $i -ge 0 ];
do
    echo counting down, from 100 to 0, now at $i;
    let i--;
done
```

Конструкция **if then else** позволяет выполнить действие, если выполняется условие, иначе – выполняется секция else.

```
#!/bin/bash
if [ -f isit.txt ]
then echo "file isit.txt exists!"
else
    echo "file isit.txt not found!"
fi
```

И выполнение самого скрипта:

```
ovgerman@ovgerman:~$ ./script18
file isit.txt not found!
```

Конструкция **if then elif**, если условий выбора несколько:

```
#!/bin/bash
count=42
if [ $count -eq 42 ]
then
    echo "42 is correct."
elif [ $count -gt 42 ]
then
    echo "Too much."
else
    echo "Not enough."
fi
```

Команда **find** позволяет найти имена файлов, заканчивающихся на “e1”

\$find –name “*e1”

Найти все файлы, заканчивающиеся на .conf:

```
$find -type f -name "*.conf" > file2
```

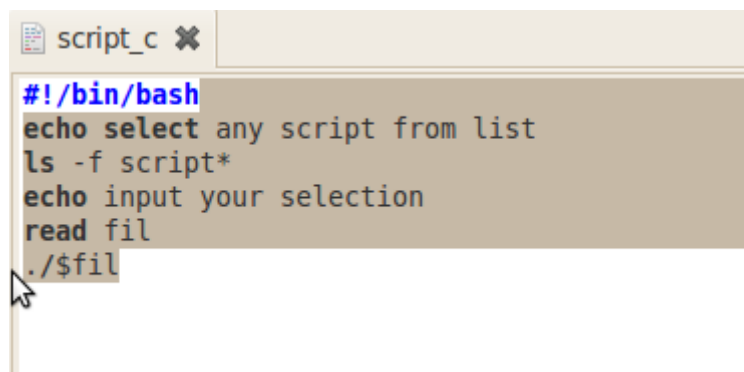
Здесь результат выполнения операции записывается в файл с именем file2.

Найти все подкаталоги текущего каталога:

```
$find -type d
```

ЗАДАНИЕ.

1. Написать скрипт, который записывает содержимое одного файла в другой и при этом оба файла существуют до операции. Дать два разных варианта выполнения.
2. Написать скрипт, в котором предлагается выбрать скрипт из списка и выполнить его.



```
#!/bin/bash
echo select any script from list
ls -f script*
echo input your selection
read fil
./$fil
```

3. Найти все файлы, начинающиеся на слово script и записать их имена в файл list.txt.
4. Написать скрипт, который проверяет, содержится ли скриптовый файл в папке, если да, то выполнить его.
5. Написать скрипт, подсчитывающий сумму от 1 до 10.