Лабораторная работа. Командные файлы BASH.

Задачи

1. Создание просты скриптов
2. Работа с операторами условного перехода
3. Работа с операторами циклов

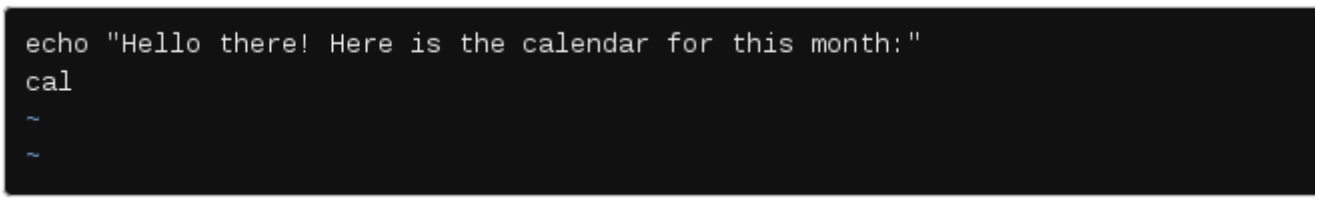
Описание

Ниже приведены команды для работы в ОС Linux в консольном режиме. Изучите приведенные команды и отработайте их в командной строке Linux. При выполнении работы проявите творчество и поэкспериментируйте с командами.

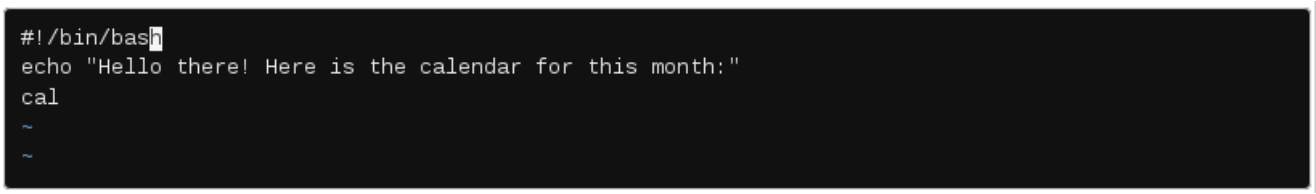
Базовый скриптинг

Скрипт позволяет вам записать сложный набор команд и выполнять их по мере необходимости. Напишите команды в файл и запусти файл как программу. Это сэкономит вам время при выполнении повторяющихся повседневных операций.
Вы можете использовать любой текстовый редактор по своему усмотрению.

Для создания простого скрипта, достаточно создать файл и записать в него команды.
Для примера создайте файл sample.sh и запишите следующие команды:
username@linux-pc:~$ nano sample.sh
echo "Hello there! Here is the calendar for this month:"
cal



Чтобы в явном виде указать, что файл является скриптом в первой строке файла необходимо указать пусть к исполняемому файлу командного процессора. Строку необходимо начать с символов #!
Например, #!/bin/bash



Для выполнения скрипта вы можете выполнить команду bash с указанием файла.
username@linux-pc:~$ bash sample.sh

Для того, чтобы не указывать bash перед скриптом, необходимо добавить файлы права на выполнение пользователем.

username@linux-pc:~$ ls -l sample.sh
username@linux-pc:~$ chmod a+x sample.sh
username@linux-pc:~$ ls -l sample.sh
username@linux-pc:~$ ./sample.sh

Обычно в скрипте могут применяться обратные кавычки для получения значения от выполнения другой команды.
Запишите в файл:
echo "Today is" `date +%A`

username@linux-pc:~$ sample.sh
Ошибка? Для запуска скрипта из текущей директории необходимо использовать ./

Чтобы иметь возможность запуска скрипта без указания пути к файлу, его необходимо поместить в одну из директорий, перечисленных в переменной $PATH

username@linux-pc:~$ echo $PATH
root@linux-pc:~# ln -s ~/sample.sh /bin
username@linux-pc:~$ sample.sh

Операторы условного перехода

В более сложных скриптах для пояснения, что происходит при выполнении файла добавляются комментарии. Комментарий — это строка, которая не выполняется командным процессором. Комментарий начинается с символа # и длится до конца строки.

В этой колонке текст скрипта drive.sh	**В этой колонке комментарии к командам файла (необязательно их вводить)**
echo "Please enter your age"	# вывести текст на экран
read age	# считать значение переменной $age с клавиатуры

Более сложные скрипты могут использовать условия (команда if). Команда if может использовать результат работы команды test, которая выполняет логические операции.

Создайте файл drive.sh для исследования работы команд if и test.

В этой колонке текст скрипта drive.sh	**В этой колонке комментарии к командам файла (необязательно их вводить)**
#!/bin/bash	
echo "Please enter your age"	# вывести текст на экран
read age	# считать значение переменной $age с клавиатуры
if test $age -lt 16	# test $age -lt 16 возвращает "true" если $age численно меньше 16
then	
echo "You are not old enough to drive."	# выполняется, если команда test вернула "true"
else	
echo "You can drive!"	# выполняется, если команда test вернула "false"
fi	# завершение команды if

Сделайте файл исполняемым и запустите его:
username@linux-pc:~$ cat drive.sh
username@linux-pc:~$ chmod a+x drive.sh
username@linux-pc:~$ ./drive.sh

Примечание: $age должно быть целым числом, иначе скрипт завершится с ошибкой.

Команда test автоматически вызывается, когда вы помещаете условие через пробелы внутри квадратных скобок if [$age -lt 16]

Модифицируйте файл drive.sh и запусти его.
username@linux-pc:~$ ./drive.sh

Обратите внимание, что вокруг квадратных скобок должны быть пробелы.

Вы также можете использовать и другие команды, возвращающие истину или ложь. Создайте файл check.sh для проверки наличия имени пользователя в файле /etc/passwd

username@linux-pc:~$ nano check.sh

```
#!/bin/bash
echo "Enter a username to check: "
read name
if grep $name /etc/passwd > /dev/null
then
   echo "$name is on this system"
else
   echo "$name does not exist"
fi
```

Добавьте права на выполнение и запустите файл.

username@linux-pc:~$ chmod a+x check.sh
username@linux-pc:~$ ./check.sh

Операторы цикла

Оператор цикла while выполняет код повторно до тех пор, пока условие истинно (результат работы в переменной $? команды равен 0).

Создайте файл num.sh для исследования работы цикла while.

В этой колонке текст скрипта num.sh	**В этой колонке комментарии к командам файла (необязательно их вводить)**
#!/bin/bash	
echo "Please enter a number greater than 100"	
read num	
while [$num -le 100]	# выполняет код от do до done пока условие истинно
do	
echo "$num is NOT greater than 100."	
echo "Please enter a number greater than 100"	
read num	
done	# завершение цикла
echo "Finally, $num is greater than 100"	

Добавьте права на выполнение и запустите файл.
username@linux-pc:~$ chmod a+x num.sh
username@linux-pc:~$ ./num.sh

Скрипты — это часть командного процессора BASH. Это означает что вы можете использовать выражения непосредственно в консоли. Это может быть полезно для цикла for, который помещает в переменную одно значение из списка, что позволяет выполнить ряд действий над каждым элементом.
Например, выполните данную команду в консоли:

```
for name in /etc/passwd /etc/hosts /etc/group
do
        wc $name
done
```

```
root@RTR:~# for NAME in /etc/passwd /etc/hosts /etc/group
> do
> wc $NAME
> done
  28   42 1501 /etc/passwd
   7   22 186 /etc/hosts
  52   52 728 /etc/group
root@RTR:~# _
```

Команды wc будет выполнена для каждого файла в списке в отдельности.

Часто команда seq используется совместно с циклом for. Команда seq генерирует список целых чисел в заданном диапазоне.
Например, выполните команду для создания 12 файлов от file1 до file12.

```
for num in `seq 1 12`
do
        touch test$num
done
```

Задание

Создайте скрипт для выполнения следующей задачи:

1. Скрипт должен создавать в домашнем каталоге пользователя директорию BackUp, если она не существует, если директория существует — вывести сообщение на экран и продолжить выполнение команд.
2. Если в домашнем каталоге пользователя имеются файлы (не директории), то создать tar архив из данных файлов и поместить архив в каталог BackUp с именем backup-текущая-дата.tar, например, backup-2017-09-06.tar.
3. Если в директории BackUp такой файл существует, то добавить к имени файла текущее время.