

Липецкий государственный технический университет

Кафедра прикладной математики

Отчет по лабораторной работе № 4
«Программирование на SHELL. Использование командных
файлов»
по курсу «Операционная система Linux»

Студент

подпись, дата

Сергеев Е.С.
фамилия, инициалы

Группа ПМ-19-2

Руководитель

Доцент, к. пед. наук
ученая степень, ученое звание

подпись, дата

Кургасов В.В.
фамилия, инициалы

Липецк 2021 г.

Содержание

Цель работы	3
Задание кафедры	4
1. Ход работы	7
Вывод	26

Цель работы

Изучение основных возможностей языка программирования Shell с целью автоматизации процесса администрирования системы за счет написания и использования командных файлов.

Задание кафедры

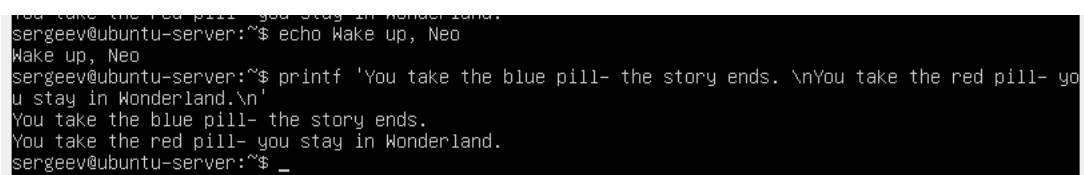
1. Используя команды ECHO, PRINTF вывести информационные сообщения на экран.
2. Присвоить переменной A целочисленное значение. Просмотреть значение переменной A.
3. Присвоить переменной B значение переменной A. Просмотреть значение переменной B.
4. Присвоить переменной C значение “путь до своего каталога”. Перейти в этот каталог с использованием переменной.
5. Присвоить переменной D значение “имя команды”, а именно, команды DATE. Выполнить эту команду, используя значение переменной.
6. Присвоить переменной E значение “имя команды”, а именно, команды просмотра содержимого файла, просмотреть содержимое переменной. Выполнить эту команду, используя значение переменной.
7. Присвоить переменной F значение “имя команды”, а именно сортировки содержимого текстового файла. Выполнить эту команду, используя значение переменной. Написать скрипты, при запуске которых выполняются следующие действия:
8. Программа запрашивает значение переменной, а затем выводит значение этой переменной.
9. Программа запрашивает имя пользователя, затем здоровается с ним, используя значение введенной переменной.
10. Программа запрашивает значения двух переменных, вычисляет сумму (разность, произведение, деление) этих переменных. Результат выводится на экран (использовать команды а) EXPR; б) BC).,
11. Вычислить объем цилиндра. Исходные данные запрашиваются программой. Результат выводится на экран.

12. Используя позиционные параметры, отобразить имя программы, количество аргументов командной строки, значение каждого аргумента командной строки.
13. Используя позиционный параметр, отобразить содержимое текстового файла, указанного в качестве аргумента командной строки. После паузы экран очищается.
14. Используя оператор FOR, отобразить содержимое текстовых файлов текущего каталога поэкранно.
15. Программой запрашивается ввод числа, значение которого затем сравнивается с допустимым значением. В результате этого сравнения на экран выдаются соответствующие сообщения.
16. Программой запрашивается год, определяется, високосный ли он. Результат выдается на экран.
17. Вводятся целочисленные значения двух переменных. Вводится диапазон данных. Пока значения переменных находятся в указанном диапазоне, их значения инкрементируются.
18. В качестве аргумента командной строки указывается пароль. Если пароль введен верно, постранично отображается в длинном формате с указанием скрытых файлов содержимое каталога /etc.
19. Проверить, существует ли файл. Если да, выводится на экран его содержимое, если нет - выдается соответствующее сообщение.
20. Если файл есть каталог и этот каталог можно читать, просматривается содержимое этого каталога. Если каталог отсутствует, он создается. Если файл не есть каталог, просматривается содержимое файла.
21. Анализируются атрибуты файла. Если первый файл существует и используется для чтения, а второй файл существует и используется для записи, то содержимое первого файла перенаправляется во второй файл. В случае несовпадений указанных атрибутов или отсутствия файлов на экран выдаются соответствующие сообщения (использовать а) имена файлов; б) позиционные параметры).

22. Если файл запуска программы найден, программа запускается (по выбору).
23. В качестве позиционного параметра задается файл, анализируется его размер. Если размер файла больше нуля, содержимое файла сортируется по первому столбцу по возрастанию, отсортированная информация помещается в другой файл, содержимое которого затем отображается на экране.
24. Командой TAR осуществляется сборка всех текстовых файлов текущего каталога в один архивный файл `my.tar`, после паузы просматривается содержимое файла `my.tar`, затем командой GZIP архивный файл `my.tar` сжимается.
25. Написать скрипт с использованием функции, например, функции, суммирующей значения двух переменных.

1. Ход работы

1. Используя команды ECHO, PRINTF вывести информационные сообщения на экран.



```
sergeev@ubuntu-server:~$ echo Wake up, Neo
Wake up, Neo
sergeev@ubuntu-server:~$ printf 'You take the blue pill- the story ends. \nYou take the red pill- you stay in Wonderland.\n'
You take the blue pill- the story ends.
You take the red pill- you stay in Wonderland.
sergeev@ubuntu-server:~$ _
```

Рисунок 1 – Задание 1

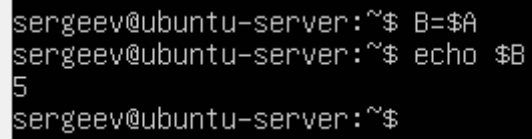
2. Присвоить переменной *A* целочисленное значение. Просмотреть значение переменной *A*.

A terminal window with a black background and white text. The prompt is 'sergeev@ubuntu-server:~\$'. The first command is 'A=5'. The second command is 'echo \$A', which outputs '5'. The prompt is shown again at the end.

```
sergeev@ubuntu-server:~$ A=5
sergeev@ubuntu-server:~$ echo $A
5
sergeev@ubuntu-server:~$
```

Рисунок 2 – Задание 2

3. Присвоить переменной В значение переменной А. Просмотреть значение переменной В.

A terminal window with a black background and white text. It shows three lines of commands and their output. The first line is 'sergeev@ubuntu-server:~\$ B=\$A'. The second line is 'sergeev@ubuntu-server:~\$ echo \$B' followed by the output '5' on the next line. The third line is 'sergeev@ubuntu-server:~\$' followed by a blank line.

```
sergeev@ubuntu-server:~$ B=$A
sergeev@ubuntu-server:~$ echo $B
5
sergeev@ubuntu-server:~$
```

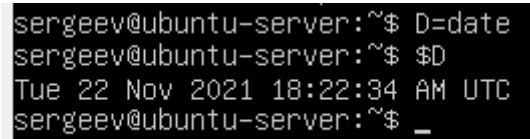
Рисунок 3 – Задание 3

4. Присвоить переменной C значение “путь до своего каталога”. Перейти в этот каталог с использованием переменной.

```
sergeev@ubuntu-server:~$ C=$PWD
sergeev@ubuntu-server:~$ echo $C
/home/sergeev
sergeev@ubuntu-server:~$ cd ..
sergeev@ubuntu-server:/home$ cd ..
sergeev@ubuntu-server:/$ cd $C
sergeev@ubuntu-server:~$ pwd
/home/sergeev
sergeev@ubuntu-server:~$ _
```

Рисунок 4 – Задание 4

5. Присвоить переменной D значение “имя команды”, а именно, команды DATE. Выполнить эту команду, используя значение переменной.



```
sergeev@ubuntu-server:~$ D=date  
sergeev@ubuntu-server:~$ $D  
Tue 22 Nov 2021 18:22:34 AM UTC  
sergeev@ubuntu-server:~$ _
```

Рисунок 5 – Задание 5

6. присвоить переменной E значение “имя команды”, а именно, команды просмотра содержимого файла, посмотреть содержимое переменной. Выполнить эту команду, используя значение переменной.

```
sergeev@ubuntu-server:~$ echo hello > test.txt
sergeev@ubuntu-server:~$ E=cat
sergeev@ubuntu-server:~$ $E test.txt
hello
sergeev@ubuntu-server:~$ _
```

Рисунок 6 – Задание 6

7. присвоить переменной F значение “имя команды”, а именно сортировки содержимого текстового файла. Выполнить эту команду, используя значение переменной.

```
sergeev@ubuntu-server:~$ printf '3\n1\n2\n' > test.txt
sergeev@ubuntu-server:~$ F=sort
sergeev@ubuntu-server:~$ $F test.txt
1
2
3
sergeev@ubuntu-server:~$ cat test.txt
3
1
2
sergeev@ubuntu-server:~$ _
```

Рисунок 7 – Задание 7

8. программа запрашивает значение переменной, а затем выводит значение этой переменной.

```
sergeev@ubuntu-server:~$ printf 'echo input:\nread A=\necho output:\necho $A\n' > scr
sergeev@ubuntu-server:~$ chmod ugo+x scr
sergeev@ubuntu-server:~$ sh scr
input:
5
output:
5
sergeev@ubuntu-server:~$ _
```

Рисунок 8 – Задание 8

9. Программа запрашивает имя пользователя, затем здоровается с ним, используя значение введенной переменной.

```
sergeev@ubuntu-server:~$ printf 'echo Your name:\nread name=\necho Hello, $name\n' > scr
sergeev@ubuntu-server:~$ sh scr
Your name:
Egor
Hello, Egor
sergeev@ubuntu-server:~$
```

Рисунок 9 – Задание 9

10. Программа запрашивает значения двух переменных, вычисляет сумму (разность, произведение, деление) этих переменных. Результат выводится на экран

```
sergeev@ubuntu-server:~$ printf 'echo A:\nread A=\n echo B:\nread B=\n echo sum=$(expr $A + $B)\n echo  
diff=$(expr $A - $B)\n echo proizv=$(expr $A \* $B)\n echo del=$(expr $A / $B)\n' > scr  
sergeev@ubuntu-server:~$ sh scr  
A:  
4  
B:  
2  
sum=6  
diff=2  
proizv=8  
del=2  
sergeev@ubuntu-server:~$
```

Рисунок 10 – Задание 10

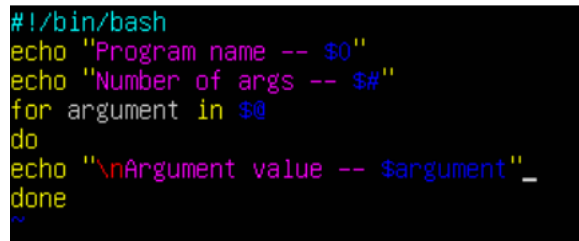
11. Вычислить объем цилиндра. Исходные данные запрашиваются программой. Результат выводится на экран.

```
sergeev@ubuntu-server:~$ printf 'echo S:\nread S\n echo h:\nread h\n echo V = $(expr $S \* $h)' > scr
sergeev@ubuntu-server:~$ sh scr
S:
3
h:
5
V = 15
sergeev@ubuntu-server:~$
```

Рисунок 11 – Задание 11

12. Используя позиционные параметры, отобразить имя программы, количество аргументов командной строки, значение каждого аргумента командной строки.

```
#!/bin/bash
echo "Program name -- $0"
echo "Number of args -- $#"
```



```
for argument in $@
do
echo "\nArgument value -- $argument"
done
```

Рисунок 12 – Задание 12 скрипт

```
sergeev@ubuntu-server:~$ ./script.sh Egor is very tired
Program name -- ./script.sh
Number of args -- 4
\nArgument value -- Egor
\nArgument value -- is
\nArgument value -- very
\nArgument value -- tired
sergeev@ubuntu-server:~$ _
```

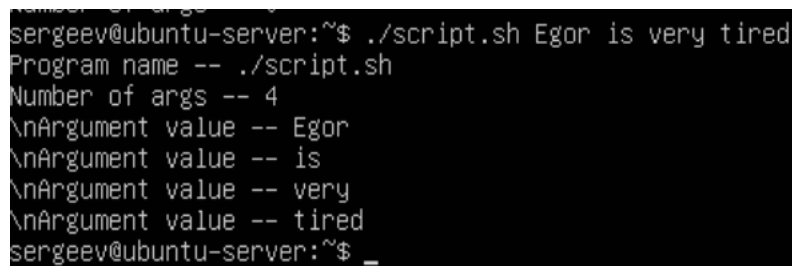
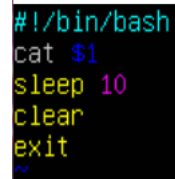


Рисунок 13 – Задание 12

13. Используя позиционный параметр, отобразить содержимое текстового файла, указанного в качестве аргумента командной строки. После паузы экран очищается.



```
#!/bin/bash
cat $1
sleep 10
clear
exit
```

Рисунок 14 – Задание 13 скрипт



```
sergeev@ubuntu-server:~$ ./script.sh 1.txt
what's up,
dude?
```

Рисунок 15 – Задание 13

14. Используя оператор FOR, отобразить содержимое текстовых файлов текущего каталога поэкранно.

```
#!/bin/bash
for eachfile in ./*
do
    if [ -f $eachfile ]
    then
        cat $eachfile | less
    fi
done_
```

Рисунок 16 – Задание 14 скрипт

```
sergeev@ubuntu-server:~$ ./script.sh
what's up,
dude?
(END)_
```

Рисунок 17 – Задание 14

15. Программой запрашивается ввод числа, значение которого затем сравнивается с допустимым значением. В результате этого сравнения на экран выдаются соответствующие сообщения.

```
#!/bin/bash
printf "a="
read a=
if [ $a -ne 10 ]
then
    echo "a!=10"
else
    echo "a=10"
fi
```

Рисунок 18 – Задание 15 скрипт

```
sergeev@ubuntu-server:~$ sh script.sh
a=5
a!=10
sergeev@ubuntu-server:~$ sh script.sh
a=10
a=10
sergeev@ubuntu-server:~$ _
```

Рисунок 19 – Задание 15

16. Программой запрашивается ввод числа, значение которого затем сравнивается с допустимым значением. В результате этого сравнения на экран выдаются соответствующие сообщения.

```
#!/bin/bash
printf "a="
read a=
if [ `expr $a % 4` -eq 0 ]
then
    if [ `expr $a % 100` -ne 0 ]
    then
        echo "leap year"
    else
        if [ `expr $a % 400` -eq 0 ]
        then
            echo "leap year"
        else
            echo "common year"
        fi
    fi
else
    echo "common year"
fi_
```

Рисунок 20 – Задание 16 скрипт

```
sergeev@ubuntu-server:~$ sh script.sh
a=2012
leap year
sergeev@ubuntu-server:~$ sh script.sh
a=2021
common year
sergeev@ubuntu-server:~$
```

Рисунок 21 – Задание 16

17. вводятся целочисленные значения двух переменных. Вводится диапазон данных. Пока значения переменных находятся в указанном диапазоне, их значения инкрементируются

```
#!/bin/bash
printf "a="
read a=
printf "\nb="
read b=
printf "\nborder:"
read border1=
read border2=
while [ $a -le $border2 ] && [ $a -gt $border1 ]
do
    a=$(expr $a + 1)
done
while [ $b -le $border2 ] && [ $b -gt $border1 ]
do
    b=$(expr $b + 1)
done
printf "a="
echo $a
printf "b="
echo $b
~
```

Рисунок 22 – Задание 17 скрипт

```
sergeev@ubuntu-server:~$ sh script.sh
a=5

b=7

border:6
10
a=5
b=11
sergeev@ubuntu-server:~$
```

Рисунок 23 – Задание 17

18. В качестве аргумента командной строки указывается пароль. Если пароль введен верно, постранично отображается в длинном формате с указанием скрытых файлов содержимое каталога /etc

```
#!/bin/bash
printf "password:"
read pass=
truepass="pass"
if [ $pass = $truepass ]
then
    ls -a -l /etc | less
else
    echo "wrong password"
fi
```

Рисунок 24 – Задание 18 скрипт

```
-rw-r--r-- 1 root root 5635 окт 13 13:43 ca-certificates
drwxr-xr-x 4 root root 4096 окт 15 20:30 cloud
drwxr-xr-x 2 root root 4096 окт 15 20:32 console-setup
drwxr-xr-x 2 root root 4096 дек 17 21:04 cron.d
drwxr-xr-x 2 root root 4096 янв 21 17:44 cron.daily
drwxr-xr-x 2 root root 4096 окт 13 13:46 cron.hourly
drwxr-xr-x 2 root root 4096 окт 13 13:46 cron.monthly
-rw-r--r-- 1 root root 1136 авг 6 10:59 crontab
drwxr-xr-x 2 root root 4096 окт 13 13:46 cron.weekly
drwxr-xr-x 2 root root 4096 окт 13 13:46 cryptsetup-initramfs
-rw-r--r-- 1 root root 54 окт 13 13:43 crypttab
drwxr-xr-x 4 root root 4096 окт 13 13:43 dbus-1
-rw-r--r-- 1 root root 2969 июн 10 2021 debconf.conf
-rw-r--r-- 1 root root 5 апр 28 2021 debian_version
drwxr-xr-x 3 root root 4096 янв 21 17:44 default
-rw-r--r-- 1 root root 604 сен 15 2018 deluser.conf
drwxr-xr-x 2 root root 4096 окт 13 13:45 depmod.d
drwxr-xr-x 4 root root 4096 окт 13 13:45 dhcp
drwxr-xr-x 4 root root 4096 окт 13 13:44 dpkg
:
```

Рисунок 25 – Задание 18

19. Проверить, существует ли файл. Если да, выводится на экран его содержимое, если нет - выдается соответствующее сообщение.

```
#!/bin/bash
printf "filename:"
read name=
if [ -s $name ]
then
    cat $name_
else
    echo "file not found"
fi
~
```

Рисунок 26 – Задание 19 скрипт

```
sergeev@ubuntu-server:~$ sh script.sh
filename:2.txt
file not found
sergeev@ubuntu-server:~$ sh script.sh
filename:1.txt
what's up,
dude?
sergeev@ubuntu-server:~$ _
```

Рисунок 27 – Задание 19

Вывод

В ходе выполнения данной лабораторной работы мной был получен опыт работы с процессами в ОС Linux.