Липецкий государственный технический университет

Кафедра прикладной математики

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

Студент _____ Богомолов Е.А.
подпись, дата фамилия, инициалы

Группа

Руководитель
Доцент, к. пед. наук _____ Кургасов В.В.
ученая степень, ученое звание подпись, дата фамилия, инициалы

Липецк 2021 г.

Содержание

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.

```
ubuntu-user@ubuntu-server:~$ echo Hello, World!
Hello, World!
ubuntu-user@ubuntu-server:~$ printf 'Hello, World!\nMy name -- Evgeny!\n'
Hello, World!
My name -- Evgeny!
ubuntu-user@ubuntu-server:~$
```

Рисунок 1 – Задание 1.

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

```
ubuntu-user@ubuntu-server:~$ A=Hello
ubuntu-user@ubuntu-server:~$ echo $A
Hello
ubuntu-user@ubuntu-server:~$
```

Рисунок 2 – Задание 2.

3. Присвоить переменной B значение переменной A. Просмотреть значение переменной B.

```
ubuntu-user@ubuntu-server:~$ B=$A
ubuntu-user@ubuntu-server:~$ echo $B
Hello
ubuntu-user@ubuntu-server:~$ _
```

Рисунок 3 – Задание 3.

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

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

Рисунок 4 – Задание 4.

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

```
ubuntu-user@ubuntu-server:~$ D=date
ubuntu-user@ubuntu-server:~$ $D
Sun 21 Nov 2021 10:58:48 AM UTC
ubuntu-user@ubuntu-server:~$ _
```

Рисунок 5 – Задание 5.

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

```
ubuntu-user@ubuntu-server:~$ echo Hello > 1.txt
ubuntu-user@ubuntu-server:~$ E=cat
ubuntu-user@ubuntu-server:~$ $E 1.txt
Hello
ubuntu-user@ubuntu-server:~$ e_
```

Рисунок 6 – Задание 6.

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

```
ubuntu-user@ubuntu-server:~$ printf 'A\nD\nC\nB\n' > 1.txt
ubuntu-user@ubuntu-server:~$ F=sort
ubuntu-user@ubuntu-server:~$ $F 1.txt
A
B
C
D
ubuntu-user@ubuntu-server:~$ cat 1.txt
A
D
C
B
ubuntu-user@ubuntu-server:~$ _
```

Рисунок 7 – Задание 7.

Написать скрипты, при запуске которых выполняются следующие действия:

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

```
ubuntu-user@ubuntu-server:~$ printf 'echo Input:\nread A=\necho Output:\necho $A\n' > scr
ubuntu-user@ubuntu-server:~$ chmod ugo+x scr
ubuntu-user@ubuntu-server:~$ sh scr
Input:
10
Output:
10
ubuntu-user@ubuntu-server:~$
```

Рисунок 8 – Задание 8.

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

```
printf 'Input name:'
read name=
printf '\nHello, '
echo $name
```

Рисунок 9 – Задание 9 (Текст скрипта).

```
ubuntu-user@ubuntu-server:~$ sh scr
Input name:Evgeny

Hello, Evgeny
ubuntu-user@ubuntu-server:~$
```

Рисунок 10 – Задание 9.

10. Программа запрашивает значения двух переменных, вычисляет сумму (разность, произведение, деление) этих переменных. Результат выводится на экран (использовать команды а) EXPR; б) BC).,

```
printf 'Input A:'
read A=
printf '\nInput B:'
read B=
printf '\nOutput with expr:\n'
sum=$(expr $A + $B)
razn=$(expr $A - $B)
proiz=$(expr $A \* $B)
delen=$(expr $A / $B)
printf "Summa: $sum\nRaznost: $razn\nProizvedenie: $proiz\nDelenie: $delen\n"
printf '\nOutput with BC:\nSumma: '
echo "$A + $B" |bc
printf '\nRaznost: '
echo "$A - $B" |bc
printf '\nProizvedenie: '
echo "$A * $B" |bc
printf '\nDelenie: '
echo "$A / $B" |bc
```

Рисунок 11 – Задание 10 (Текст скрипта).

```
ubuntu-user@ubuntu-server:~$ sh scr
Input A:12

Input B:44

Output with expr:
Summa: 56
Raznost: -32
Proizvedenie: 528
Delenie: 0

Output with BC:
Summa: 56

Raznost: -32

Proizvedenie: 528

Delenie: 0
ubuntu-user@ubuntu-server:~$
```

Рисунок 12 – Задание 10.

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

```
printf 'Input S: '
read S=
printf '\nInput h: '
read h=
printf '\nV = '
echo "$S * $h"_|bc
printf '\n'
```

Рисунок 13 – Задание 11 (Текст скрипта).

```
ubuntu-user@ubuntu-server:~$ sh scr
Input S: 12

Input h: 44

V = 528

ubuntu-user@ubuntu-server:~$ _
```

Рисунок 14 – 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 arguments -- $#"
for argument in $@
do
echo "\nArgument value -- $argument"
done
```

Рисунок 15 – Задание 12 (Текст скрипта).

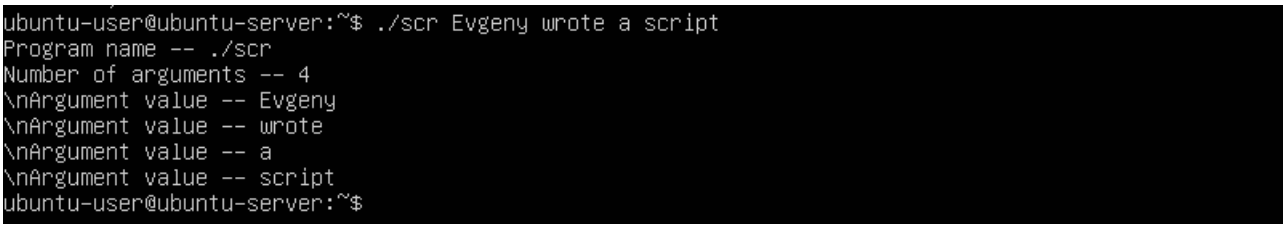

Рисунок 16 – Задание 12.

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

Рисунок 17 – Задание 13 (Текст скрипта).

Рисунок 18 – 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 19 – Задание 14 (Текст скрипта).

```
ubuntu-user@ubuntu-server:~$ ./scr
Hello
World
(END)
```

Рисунок 20 – 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 12 ]
then
       echo "Number is not equil to 12"
else
        echo "The number is 12"
fi_
```

Рисунок 21 – Задание 15 (Текст скрипта).

```
ubuntu-user@ubuntu-server:~$ sh scr
A=12
The number is 12
ubuntu-user@ubuntu-server:~$ sh scr
A=44
Number is not equil to 12
ubuntu-user@ubuntu-server:~$
```

Рисунок 22 – Задание 15.

16. Программой запрашивается год, определяется, високосный ли он. Результат выдается на экран.

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

Рисунок 23 – Задание 16 (Текст скрипта).

```
ubuntu-user@ubuntu-server:~$ sh scr
YEAR -- 2000
Leap year
ubuntu-user@ubuntu-server:~$ sh scr
YEAR -- 2012
Leap year!
ubuntu-user@ubuntu-server:~$ sh scr
YEAR -- 2001
Common year
```

Рисунок 24 – Задание 16.

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

```
#!/bin/bash
printf "First count -- "
read count1=
printf "\nSecond count -- "
read count2=
printf "\nLeft border of the range -- "
read bord1=
printf "\nRight border of the range -- "
read bord2=
while [ $count1 -le $bord2 ] && [ $count1 -gt $bord1 ]
do
        count1=$(expr $count1 + 1)
done
while [ $count2 -le $bord2 ] && [ $count2 -gt $bord1 ]
do
        count2=$(expr $count2 + 1)
done
printf "First count -- "
echo $count1
printf "Second count -- "
echo $count2
```

Рисунок 25 – Задание 17 (Текст скрипта).

```
ubuntu-user@ubuntu-server:~$ sh scr
First count -- 1

Second count -- 11

Left border of the range -- 10

Right border of the range -- 20
First count -- 1
Second count -- 21
ubuntu-user@ubuntu-server:~$ sh scr
First count -- 12

Second count -- 100

Left border of the range -- 5

Right border of the range -- 50
First count -- 51
Second count -- 100
ubuntu-user@ubuntu-server:~$ _
```

Рисунок 26 – 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 password=
true_password="password"
if [ $password = $true_password ]
then
        ls -a -l /etc | less
else
        echo "Password not confirm!"
fi_
```

Рисунок 27 – Задание 18 (Текст скрипта).

```
total 812
drwxr-xr-x 97 root root       4096 Nov 20 12:11 .
drwxr-xr-x 20 root root       4096 Oct 13 11:14 ..
-rw-r--r--  1 root root       3028 Aug 24 08:42 adduser.conf
drwxr-xr-x  2 root root       4096 Aug 24 08:47 alternatives
drwxr-xr-x  3 root root       4096 Aug 24 08:47 apparmor
drwxr-xr-x  7 root root       4096 Aug 24 08:47 apparmor.d
drwxr-xr-x  3 root root       4096 Nov 20 12:11 apport
drwxr-xr-x  7 root root       4096 Oct 13 11:09 apt
-rw-r-----  1 root daemon      144 Nov 12  2018 at.deny
-rw-r--r--  1 root root       2319 Feb 25  2020 bash.bashrc
-rw-r--r--  1 root root         45 Jan 26  2020 bash_completion
drwxr-xr-x  2 root root       4096 Nov 20 12:11 bash_completion.d
-rw-r--r--  1 root root        367 Apr 14  2020 bindresvport.blacklist
drwxr-xr-x  2 root root       4096 Apr 22  2020 binfmt.d
drwxr-xr-x  2 root root       4096 Aug 24 08:47 byobu
drwxr-xr-x  3 root root       4096 Aug 24 08:42 ca-certificates
-rw-r--r--  1 root root       6570 Oct 13 11:17 ca-certificates.conf
-rw-r--r--  1 root root       6569 Aug 24 08:45 ca-certificates.conf.dpkg-old
drwxr-xr-x  2 root root       4096 Aug 24 08:47 calendar
drwxr-xr-x  4 root root       4096 Oct 13 11:16 cloud
drwxr-xr-x  2 root root       4096 Oct 13 11:28 console-setup
drwxr-xr-x  2 root root       4096 Aug 24 08:47 cron.d
drwxr-xr-x  2 root root       4096 Nov 20 12:11 cron.daily
drwxr-xr-x  2 root root       4096 Aug 24 08:43 cron.hourly
drwxr-xr-x  2 root root       4096 Aug 24 08:43 cron.monthly
-rw-r--r--  1 root root       1042 Feb 13  2020 crontab
drwxr-xr-x  2 root root       4096 Aug 24 08:47 cron.weekly
drwxr-xr-x  2 root root       4096 Aug 24 08:47 cryptsetup-initramfs
-rw-r--r--  1 root root         54 Aug 24 08:46 crypttab
drwxr-xr-x  4 root root       4096 Aug 24 08:42 dbus-1
drwxr-xr-x  3 root root       4096 Aug 24 08:46 dconf
-rw-r--r--  1 root root       2969 Aug  3  2019 debconf.conf
-rw-r--r--  1 root root         13 Dec  5  2019 debian_version
drwxr-xr-x  3 root root       4096 Nov 20 12:11 default
-rw-r--r--  1 root root        604 Sep 15  2018 deluser.conf
:
```

Рисунок 28 – Задание 18.

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

```
#!/bin/bash
printf "File name -- "
read file_name=
if [ -s $file_name ]
then
        cat $file_name
else
        echo "This file not found!"
fi
```

Рисунок 29 – Задание 19 (Текст скрипта).

```
ubuntu-user@ubuntu-server:~$ sh scr
File name -- 1.txt
Hello
World
ubuntu-user@ubuntu-server:~$ sh scr
File name -- 2.txt
This file not found!
ubuntu-user@ubuntu-server:~$
```

Рисунок 30 – Задание 19.

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.

```
#!/bin/bash
printf "File name -- "
read file_name=
if [ -e $file_name ]
then
        echo "File exists"
        if [ -d $file_name ]
        then
                echo "It is directory"
                if [ -r $file_name ]
                then
                        ls $file_name
                fi
        else
                echo "It is file"
                cat $file_name
        fi
else
        echo "File not exists"
        mkdir $file_name
fi
```

Рисунок 31 – Задание 20 (Текст скрипта).

```
ubuntu-user@ubuntu-server:~$ sh scr
File name -- 1.txt
File exists
It is file
Hello
World
ubuntu-user@ubuntu-server:~$ sh scr
File name -- MYDIR
File exists
It is directory
MIFILE1  MYDIR1  MYDIR2  MYDIR3  MYFILE3
ubuntu-user@ubuntu-server:~$ sh scr
File name -- MYDIR_EMP
File not exists
ubuntu-user@ubuntu-server:~$ ls -li
total 16
269475 -rw-rw-r-- 1 ubuntu-user ubuntu-user   12 Nov 21 21:24 1.txt
269769 prw-rw-r-- 1 ubuntu-user ubuntu-user    0 Nov 12 14:15 myChannel
287027 drwxr-xr-x 5 root        root        4096 Nov 12 13:53 MYDIR
295001 drwxrwxr-x 2 ubuntu-user ubuntu-user 4096 Nov 22 05:31 MYDIR_EMP
269480 -rwxrwxr-x 1 ubuntu-user ubuntu-user  299 Nov 22 05:30 scr
ubuntu-user@ubuntu-server:~$ _
```

Рисунок 32 – Задание 20.

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).

```
#!/bin/bash
printf "First file name -- "
read file_1=
printf "\nSecond file name -- "
read file_2=
if [ -e $file_1 ]
then
        if [ -r $file_1 ]
        then
                if [ -e $file_2 ]
                then
                        if [ -w $file_2 ]
                        then
                                cat $file_1 > $file_2
                        else
                                echo "File $file_2 is not writing!"
                        fi
                else
                        echo "File $file_2 not exists!"
                fi
        else
                echo "File $file_1 is not readable!"
        fi
else
        echo "File $file_1 not exists!"
fi
```

Рисунок 33 – Задание 21, а) (Текст скрипта).

```
ubuntu-user@ubuntu-server:~$ sh scr
First file name -- 1.txt

Second file name -- 2.txt
ubuntu-user@ubuntu-server:~$ cat 2.txt
Hello
World
ubuntu-user@ubuntu-server:~$ cat 1.txt
Hello
World
ubuntu-user@ubuntu-server:~$ sh scr
First file name -- 1.txt

Second file name -- 3.txt
File 3.txt not exists!
ubuntu-user@ubuntu-server:~$ _
```

Рисунок 34 – Задание 21, а).

```
#!/bin/bash
printf "First file name -- $1\n"
file_1=$1
printf "Second file name -- $2\n"
file_2=$2
if [ -e $file_1 ]
then
        if [ -r $file_1 ]
        then
                if [ -e $file_2 ]
                then
                        if [ -w $file_2 ]
                        then
                                cat $file_1 > $file_2
                        else
                                echo "File $file_2 is not writing!"
                        fi
                else
                        echo "File $file_2 not exists!"
                fi
        else
                echo "File $file_1 is not readable!"
        fi
else
        echo "File $file_1 not exists!"
fi
```

Рисунок 35 – Задание 21, б) (Текст скрипта).

```
ubuntu-user@ubuntu-server:~$ ./scr 1.txt 2.txt
First file name -- 1.txt
Second file name -- 2.txt
ubuntu-user@ubuntu-server:~$ ./scr 1.txt 3.txt
First file name -- 1.txt
Second file name -- 3.txt
File 3.txt not exists!
ubuntu-user@ubuntu-server:~$ _
```

Рисунок 36 – Задание 21, б).

22. Если файл запуска программы найден, программа запускается (по выбору).

```
#!/bin/bash
printf "Input .exe file -- "
read exe
if [ -e $exe ]
then
        if [ -x $exe ]
        then
                sh $exe
        else
                echo "File is not executable!"
        fi
else
        echo "File is not exists!"
fi
```

Рисунок 37 – Задание 22, а) (Текст скрипта).

```
ubuntu-user@ubuntu-server:~$ ./new_script
Input .exe file -- scr
Hello, world!
ubuntu-user@ubuntu-server:~$ ./new_script
Input .exe file -- 1
File is not exists!
ubuntu-user@ubuntu-server:~$ ./new_script
Input .exe file -- 1.txt
File is not executable!
ubuntu-user@ubuntu-server:~$
```

Рисунок 38 – Задание 22, а).

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.

```
#!/bin/bash
if [ $# = 1 ]
then
        if [ -s $1 ]
        then
                sort -k1 $1 > new_file.txt
                cat new_file.txt
        else
                echo "File size less then a zero!"
        fi
fi
```

Рисунок 39 – Задание 23 (Текст скрипта).

```
ubuntu-user@ubuntu-server:~$ ./scr 1.txt
File size less then a zero!
ubuntu-user@ubuntu-server:~$ echo Hello World > 1.txt
ubuntu-user@ubuntu-server:~$ ./scr 1.txt
Hello World
ubuntu-user@ubuntu-server:~$ ls -li
total 20
263970 -rw-rw-r-- 1 ubuntu-user ubuntu-user   12 Nov 22 14:55 1.txt
264674 -rw-rw-r-- 1 ubuntu-user ubuntu-user    0 Nov 22 09:30 2.txt
269769 prw-rw-r-- 1 ubuntu-user ubuntu-user    0 Nov 12 14:15 myChannel
287027 drwxr-xr-x 5 root        root        4096 Nov 12 13:53 MYDIR
295001 drwxrwxr-x 2 ubuntu-user ubuntu-user 4096 Nov 22 05:31 MYDIR_EMP
266970 -rw-rw-r-- 1 ubuntu-user ubuntu-user    0 Nov 22 09:34 my.tar.gz
264335 -rw-rw-r-- 1 ubuntu-user ubuntu-user   12 Nov 22 14:55 new_file.txt
264820 -rwxrwxr-x 1 ubuntu-user ubuntu-user  149 Nov 22 14:54 scr
ubuntu-user@ubuntu-server:~$
```

Рисунок 40 – Задание 23.

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.

```
#!/bin/bash
finds=$(find . -type f)
tar -cf my.tar $finds
tar -tf my.tar
gzip my.tar
```

Рисунок 41 – Задание 24 (Текст скрипта).

```
ubuntu-user@ubuntu-server:~$ ./scr
./1.txt
./2.txt
./.sudo_as_admin_successful.gz
./scr
./.viminfo
./.bash_logout.gz
./.bashrc.gz
./.viminfo.gz
./MYDIR/MYDIR2/MYFILE2
./MYDIR/MYFILE3
./MYDIR/MIFILE1
./new_script.gz
./.cache/motd.legal-displayed
./.profile.gz
ubuntu-user@ubuntu-server:~$ ls -li
total 24
263970 -rw-rw-r-- 1 ubuntu-user ubuntu-user    0 Nov 22 09:30 1.txt
264674 -rw-rw-r-- 1 ubuntu-user ubuntu-user    0 Nov 22 09:30 2.txt
269769 prw-rw-r-- 1 ubuntu-user ubuntu-user    0 Nov 12 14:15 myChannel
287027 drwxr-xr-x 5 root        root        4096 Nov 12 13:53 MYDIR
295001 drwxrwxr-x 2 ubuntu-user ubuntu-user 4096 Nov 22 05:31 MYDIR_EMP
266970 -rw-rw-r-- 1 ubuntu-user ubuntu-user 5220 Nov 22 09:34 my.tar.gz
264313 -rwxrwxr-x 1 ubuntu-user ubuntu-user  151 Nov 22 08:39 new_script.gz
269434 -rwxrwxr-x 1 ubuntu-user ubuntu-user   86 Nov 22 09:34 scr
ubuntu-user@ubuntu-server:~$ _
```

Рисунок 42 – Задание 24.

25. Написать скрипт с использованием функции, например, функции, суммирующей значения двух переменных.

```
#!/bin/bash
printf "Input A: "
read A
printf "Input B: "
read B
sum () {
        sum=$(expr $A + $B)
        echo $sum
}
SUM=$(sum)
printf "SUM="
echo $SUM
diff () {
        dif=$(expr $A - $B)
        echo $dif
}
printf "Difference="
DIF=$(diff)
echo $DIF_
```

Рисунок 43 – Задание 25 (Текст скрипта).

Рисунок 44 – Задание 25.

Выводы

В ходе выполнения данной лабораторной работы мной были получены знания о основных возможностях языка программирования Shell с целью автоматизации процесса администрирования системы за счет написания и использования командных файлов.