

Липецкий государственный технический университет

Кафедра автоматизированных систем управления

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

Студент

подпись, дата

Закиров Р.Р.
фамилия, инициалы

Группа АС-20-1

Руководитель

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

подпись, дата

Кургасов В.В.
фамилия, инициалы

Липецк 2022 г.

Содержание

Цель работы	3
Задание кафедры	4
Ход работы	7
Выводы	24

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.

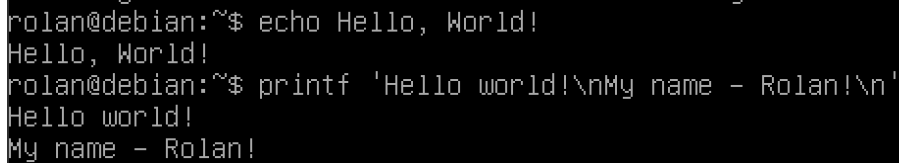
24. Командой TAR осуществляется сборка всех текстовых файлов текущего каталога в один архивный файл my.tar, после паузы просматривается содержимое файла my.tar, затем командой GZIP архивный файл my.tar

сжимается.

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

1. Ход работы

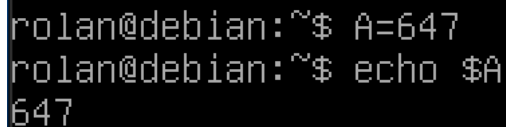
1. Используя команды ECHO, PRINTF вывести информационные сообщения на экран.



```
rolan@debian:~$ echo Hello, World!  
Hello, World!  
rolan@debian:~$ printf 'Hello world!\nMy name - Rolan!\n'  
Hello world!  
My name - Rolan!
```

Рисунок 1 – Задание 1.

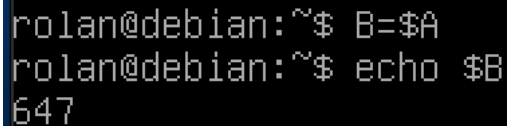
2. Присвоить переменной А целочисленное значение. Просмотреть значение переменной А.



```
rolan@debian:~$ A=647  
rolan@debian:~$ echo $A  
647
```

Рисунок 2 – Задание 2.

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



```
rolan@debian:~$ B=$A  
rolan@debian:~$ echo $B  
647
```

Рисунок 3 – Задание 3.

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

```
rolan@debian:~$ C=$PWD
rolan@debian:~$ echo $C
/home/rolan
rolan@debian:~$ cd ..
rolan@debian:/home$ cd ..
rolan@debian:/$ cd $C
rolan@debian:~$ pwd
/home/rolan
```

Рисунок 4 – Задание 4.

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

```
rolan@debian:~$ D=date
rolan@debian:~$ $D
Чт 01 дек 2022 22:07:32 MSK
```

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.

```
rolan@debian:~$ echo Hello > 1.txt
rolan@debian:~$ E=cat
rolan@debian:~$ $E 1.txt
Hello
```

Рисунок 6 – Задание 6.

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

```
rolan@debian:~$ printf 'A\nD\nC\nB\n' > 1.txt
rolan@debian:~$ F=sort
rolan@debian:~$ $F 1.txt
A
B
C
D
rolan@debian:~$ cat 1.txt
A
D
C
B
```

Рисунок 7 – Задание 7.

Написать скрипты, при запуске которых выполняются следующие действия:

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

```
rolan@debian:~$ printf 'echo Input:\nread A=\necho Output:\necho $A\n' > scr
rolan@debian:~$ chmod ugo+x scr
rolan@debian:~$ sh scr
Input:
27
Output:
27
```

Рисунок 8 – Задание 8.

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

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

Рисунок 9 – Задание 9 (Текст скрипта).

```
rolan@debian:~$ sh scr
Input name:
Rolan
Hello, Rolan
```

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).

```
rolan@debian:~$ sh scr1
Input A:32
Input B:43
Output with expr:
Summa: 75
Raznost: -11
Proizvedenie: 1376
Delenie: 0
Output with BC:
Summa: 75
Raznost: -11
Proizvedenie: 1376
Delenie: 0
```

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).

```
rolan@debian:~$ sh scr1  
Input S: 15  
  
Input h: 53  
  
V = 795
```

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).

```
root@debian:~# ./scr Rolan wrote a script
Program name - ./scr
Number of arguments - 4
\nArgument value - Rolan
\nArgument value - wrote
\nArgument value - a
\nArgument value - script
```

Рисунок 16 – 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 17 – Задание 13 (Текст скрипта).

```
rolan@debian:~$ ./scr 1.txt
Hello
World
```

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).

```
rolan@debian:~$ ./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 20 ]
then
    echo "Number is not equil to 20"
else
    echo "The number is 20"
fi
```

Рисунок 21 – Задание 15 (Текст скрипта).

```
rolan@debian:~$ sh scr
A=20
The number is 20
rolan@debian:~$ sh scr
A=15
Number is not equil to 20
```

Рисунок 22 – Задание 15.

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

```
#!/bin/bash
printf "YEAR - "
read year=
if [ $((year % 4)) -eq 0 ]
then
    if [ $((year % 100)) -eq 0 ]
    then
        if [ $((year % 400)) -eq 0 ]
        then
            echo "$year високосный"
        else
            echo "$year не високосный"
        fi
    else
        echo "$year не високосный"
    fi
else
    echo "$year не високосный"
fi_
```

Рисунок 23 – Задание 16 (Текст скрипта).

```
rolan@debian:~$ sh scr
YEAR - 2000
2000 високосный
rolan@debian:~$ sh scr
YEAR - 2002
2002 не високосный
rolan@debian:~$ sh scr
YEAR - 2022
2022 не високосный
```

Рисунок 24 – Задание 16.

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

```
read a
read b
read A
read B
if [ $a -gt $A ] || [ $b -gt $A ]
then
while [ $a -lt $B ] || [ $b -lt $B ]
do
a=$(expr $a + 1)
b=$(expr $b + 1)
done
fi
echo $a
echo $b
```

Рисунок 25 – Задание 17 (Текст скрипта).

```
rolan@debian:~$ sh scr
5
6
2
5
5
6
```

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).

```
итого 724
drwxr-xr-x 72 root root 4096 дек 2 16:07 .
drwxr-xr-x 18 root root 4096 ноя 28 22:05 ..
-rw-r--r-- 1 root root 2981 ноя 12 23:46 adduser.conf
-rw-r--r-- 1 root root 44 ноя 12 23:55 adjtime
drwxr-xr-x 2 root root 4096 ноя 12 23:54 alternatives
-rw-r--r-- 1 root root 401 фев 6 2021 anacrontab
-rw-r--r-- 1 root root 4185 июл 29 2019 analog.cfg
drwxr-xr-x 8 root root 4096 ноя 12 23:54 apache2
drwxr-xr-x 2 root root 4096 ноя 12 23:47 apparmor
drwxr-xr-x 7 root root 4096 ноя 12 23:53 apparmor.d
drwxr-xr-x 8 root root 4096 ноя 12 23:55 apt
drwxr-xr-x 2 root root 4096 ноя 12 23:53 avahi
-rw-r--r-- 1 root root 1994 мар 27 2022 bash.bashrc
-rw-r--r-- 1 root root 45 янв 25 2020 bash_completion
-rw-r--r-- 1 root root 367 июл 29 2019 bindresvport.blacklist
drwxr-xr-x 2 root root 4096 авг 7 16:25 binfmt.d
drwxr-xr-x 2 root root 4096 ноя 12 23:53 bluetooth
drwxr-xr-x 3 root root 4096 ноя 12 23:53 ca-certificates
-rw-r--r-- 1 root root 5662 ноя 12 23:53 ca-certificates.conf
drwxr-xr-x 2 root root 4096 ноя 12 23:47 console-setup
drwxr-xr-x 2 root root 4096 ноя 12 23:53 cron.d
drwxr-xr-x 2 root root 4096 ноя 12 23:54 cron.daily
drwxr-xr-x 2 root root 4096 ноя 12 23:46 cron.hourly
drwxr-xr-x 2 root root 4096 ноя 12 23:53 cron.monthly
-rw-r--r-- 1 root root 1042 фев 23 2021 crontab
drwxr-xr-x 2 root root 4096 ноя 12 23:53 cron.weekly
drwxr-xr-x 4 root root 4096 ноя 12 23:53 dbus-1
-rw-r--r-- 1 root root 2969 июн 10 2021 debconf.conf
-rw-r--r-- 1 root root 5 сен 3 15:10 debian_version
drwxr-xr-x 3 root root 4096 ноя 12 23:54 default
-rw-r--r-- 1 root root 604 июн 26 2016 deluser.conf
drwxr-xr-x 4 root root 4096 ноя 12 23:49 dhcp
drwxr-xr-x 2 root root 4096 ноя 12 23:54 dictionaries-common
drwxr-xr-x 2 root root 4096 ноя 12 23:49 discover.conf.d
-rw-r--r-- 1 root root 346 янв 15 2018 discover-modprobe.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).

```
rolan@debian:~$ sh scr
File name - 1.txt
Hello
World
rolan@debian:~$ sh scr
File name - 2.txt
This file not found
```

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 [ -s $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).

```

rolan@debian:~$ sh scr
File name - 1.txt
File exists
It is file
Hello
World
rolan@debian:~$ sh scr
File name - MYDIR
File exists
It is directory
MYDIR1 MYDIR2 MYDIR3 MYFILE1 MYFILE3
rolan@debian:~$ sh scr
File name - MYDIR_EMP
File not exists
rolan@debian:~$ ls -li
итого 44
135791 -rw-r--r-- 1 rolan rolan      12 дек  2 16:16 1.txt
135775 -rw-r--r-- 1 rolan rolan      26 ноя 28 20:42 loop
135776 -rw-r--r-- 1 rolan rolan      40 ноя 28 20:43 loop2
135782 prw-r--r-- 1 rolan rolan       0 ноя 28 22:14 myBlog
135780 prw-r--r-- 1 rolan rolan       0 ноя 28 21:49 myCh
135779 prw-r--r-- 1 rolan rolan       0 ноя 28 21:44 myChannel
135783 drwxr-xr-x 5 root  root    4096 ноя 28 22:13 MYDIR
135793 drwxr-xr-x 2 rolan rolan    4096 дек  2 21:56 MYDIR_EMP
135781 -rw-r--r-- 1 rolan rolan       56 ноя 28 22:14 out.gz
135790 -rw-r--r-- 1 root  root   10240 ноя 28 22:14 out.tar
135792 -rwxr-xr-x 1 rolan rolan     298 дек  2 21:55 scr
135794 -rw-r--r-- 1 rolan rolan     132 дек  1 23:42 scr1

```

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).

```
rolan@debian:~$ sh scr
First file name - 1.txt

Second file name - 2.txt
File 2.txt not exists!
rolan@debian:~$ touch 2.txt
rolan@debian:~$ sh scr
First file name - 1.txt

Second file name - 2.txt
rolan@debian:~$ cat 2.txt
Hello
World
rolan@debian:~$ cat 1.txt
Hello
World
```

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).

```
rolan@debian:~$ ./scr 1.txt 2.txt
First file name - 1.txt
Second file name - 2.txt
rolan@debian:~$ ./scr 1.txt 3.txt
First file name - 1.txt
Second file name - 3.txt
File 3.txt not exists!
```

Рисунок 36 – Задание 21, б).

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

```
#!/bin/bash
printf "Input .exe - "
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).

```
rolan@debian:~$ ./script
Это работает!
rolan@debian:~$ ./scr
Input .exe - script
Это работает!
rolan@debian:~$ ./scr
Input .exe - 1
File is not exists!
```

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).

```

rolan@debian:~$ echo Hello world > 1.txt
rolan@debian:~$ ./scr 1.txt
Hello world
rolan@debian:~$ ls -li
итого 56
135791 -rw-r--r-- 1 rolan rolan    12 дек  2 22:43 1.txt
135795 -rw-r--r-- 1 rolan rolan    12 дек  2 22:14 2.txt
135775 -rw-r--r-- 1 rolan rolan   26 ноя 28 20:42 loop
135776 -rw-r--r-- 1 rolan rolan   40 ноя 28 20:43 loop2
135782 prw-r--r-- 1 rolan rolan    0 ноя 28 22:14 myBlog
135780 prw-r--r-- 1 rolan rolan    0 ноя 28 21:49 myCh
135779 prw-r--r-- 1 rolan rolan    0 ноя 28 21:44 myChannel
135783 drwxr-xr-x 5 root  root   4096 ноя 28 22:13 MYDIR
135793 drwxr-xr-x 2 rolan rolan   4096 дек  2 21:56 MYDIR_EMP
135796 -rw-r--r-- 1 rolan rolan    12 дек  2 22:43 new_file.txt
135781 -rw-r--r-- 1 rolan rolan    56 ноя 28 22:14 out.gz
135790 -rw-r--r-- 1 root  root  10240 ноя 28 22:14 out.tar
135792 -rwxr-xr-x 1 rolan rolan   150 дек  2 22:40 scr
135794 -rw-r--r-- 1 rolan rolan   132 дек  1 23:42 scr1
135797 -rwxr--r-- 1 rolan rolan    32 дек  2 22:36 script
rolan@debian:~$ cat new_file.txt
Hello world

```

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).

```

rolan@debian:~$ ./scr
./MYDIR/MYDIR2/MYFILE2
./MYDIR/MYFILE3
./MYDIR/MYFILE1
./bashrc
./loop2
./script
./out.gz
./2.txt
./scr1
./bash_history
./bash_logout
./profile
./loop
./out.tar
./1.txt
./new_file.txt
./scr
rolan@debian:~$ ls -li
итого 60
135791 -rw-r--r-- 1 rolan rolan 12 дек 2 22:43 1.txt
135795 -rw-r--r-- 1 rolan rolan 12 дек 2 22:14 2.txt
135775 -rw-r--r-- 1 rolan rolan 26 ноя 28 20:42 loop
135776 -rw-r--r-- 1 rolan rolan 40 ноя 28 20:43 loop2
135782 prw-r--r-- 1 rolan rolan 0 ноя 28 22:14 myBlog
135780 prw-r--r-- 1 rolan rolan 0 ноя 28 21:49 myCh
135779 prw-r--r-- 1 rolan rolan 0 ноя 28 21:44 myChannel
135783 drwxr-xr-x 5 root root 4096 ноя 28 22:13 MYDIR
135793 drwxr-xr-x 2 rolan rolan 4096 дек 2 21:56 MYDIR_EMP
135799 -rw-r--r-- 1 rolan rolan 3015 дек 2 22:46 my.tar.gz
135796 -rw-r--r-- 1 rolan rolan 12 дек 2 22:43 new_file.txt
135781 -rw-r--r-- 1 rolan rolan 56 ноя 28 22:14 out.gz
135790 -rw-r--r-- 1 root root 10240 ноя 28 22:14 out.tar
135792 -rwxr-xr-x 1 rolan rolan 85 дек 2 22:46 scr
135794 -rw-r--r-- 1 rolan rolan 132 дек 1 23:42 scr1
135797 -rwxr--r-- 1 rolan rolan 32 дек 2 22:36 script

```

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).


```
rolan@debian:~$ ./scr  
Input A: 43  
Input B: 12  
SUM=55  
Difference=31
```

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.