

Липецкий государственный технический университет

Факультет автоматизации и информатики

Кафедра автоматизированных систем управления

ЛАБОРАТОРНАЯ РАБОТА №4

по курсу «Операционная система Linux»

на тему «Программирование на SHELL. Использование командных файлов.»

Студент

Киренский Д. К.

Группа ПИ-19-1

Руководитель

Доцент

Кургасов В.В.

Липецк 2021 г.

Задание

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.

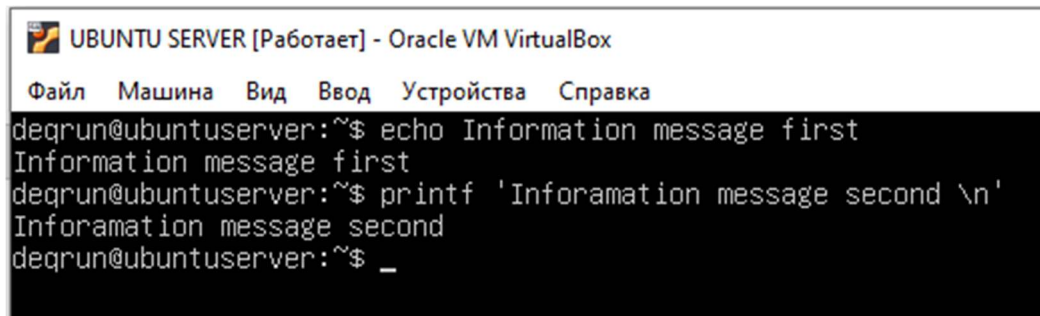
25. Написать скрипт с использованием функции, например, функции, суммирующей значения двух переменных.

Оглавление

| | |
|----------------------|----|
| 1. I часть | 4 |
| 2. II Часть. | 6 |
| Заключение. | 17 |

1. I часть

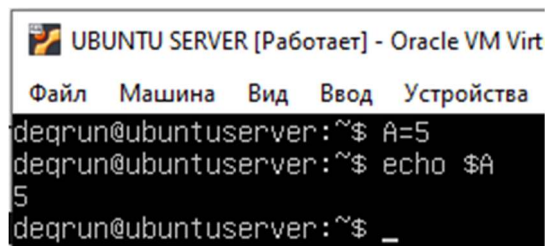
1. Используя команды ECHO, PRINTF вывести информационные сообщения на экран.



```
UBUNTU SERVER [Работает] - Oracle VM VirtualBox
Файл  Машина  Вид  Ввод  Устройства  Справка
degrun@ubuntuserver:~$ echo Information message first
Information message first
degrun@ubuntuserver:~$ printf 'Inforamation message second \n'
Inforamation message second
degrun@ubuntuserver:~$ _
```

Рисунок 1 – результат команд ECHO и PRINTF.

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



```
UBUNTU SERVER [Работает] - Oracle VM Virt
Файл  Машина  Вид  Ввод  Устройства
degrun@ubuntuserver:~$ A=5
degrun@ubuntuserver:~$ echo $A
5
degrun@ubuntuserver:~$ _
```

Рисунок 2 – результат присваивания значения переменной A.

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

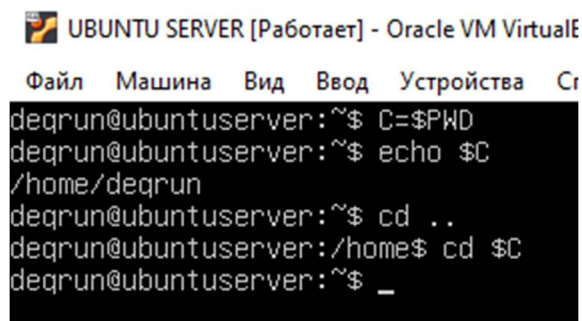


```
degrun@ubuntuserver:~$ B=$A
degrun@ubuntuserver:~$ echo $B
5
degrun@ubuntuserver:~$
```

Рисунок 3 – результат присваивания значения переменной B.

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

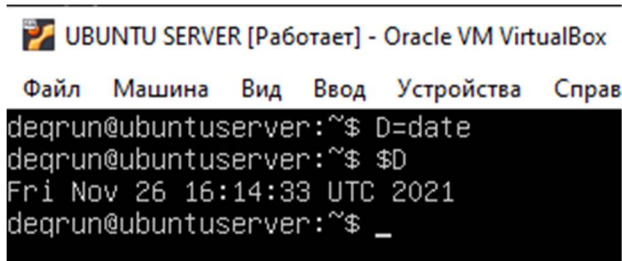
Для этого воспользуемся командой PWD – вывод пути до рабочей директории.



```
degrun@ubuntuserver:~$ C=$PWD
degrun@ubuntuserver:~$ echo $C
/home/degrun
degrun@ubuntuserver:~$ cd ..
degrun@ubuntuserver:/home$ cd $C
degrun@ubuntuserver:~$ _
```

Рисунок 4 – присваивание переменной C значение пути до этой переменной.

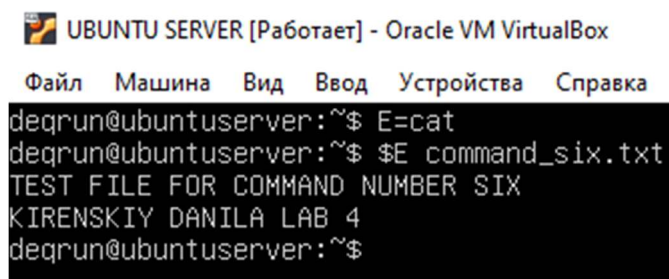
5. Присвоить переменной D значение имя команды, а именно, команды DATE. Выполнить эту команду, используя значение переменной.



```
degrun@ubuntuserver:~$ D=date
degrun@ubuntuserver:~$ $D
Fri Nov 26 16:14:33 UTC 2021
degrun@ubuntuserver:~$ _
```

Рисунок 4 – присваивание переменной D значение команды date.

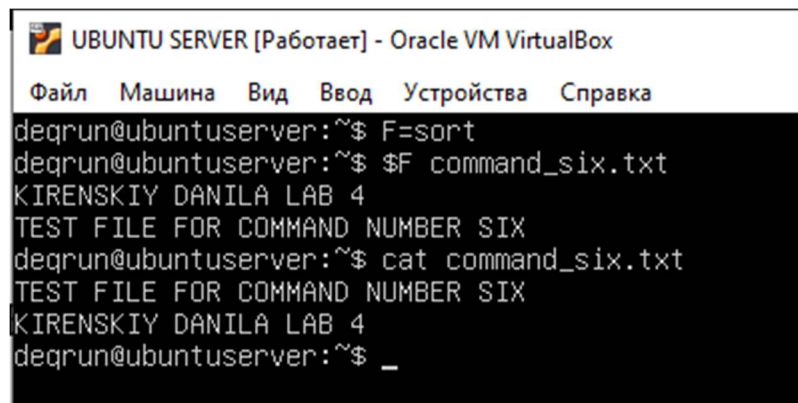
6. Присвоить переменной E значение “имя команды”, а именно, команды просмотра содержимого файла, просмотреть содержимое переменной. Выполнить эту команду, используя значение переменной.



```
degrun@ubuntuserver:~$ E=cat
degrun@ubuntuserver:~$ $E command_six.txt
TEST FILE FOR COMMAND NUMBER SIX
KIRENSKIY DANILA LAB 4
degrun@ubuntuserver:~$
```

Рисунок 6 – присваивание переменной E значение команды cat.

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



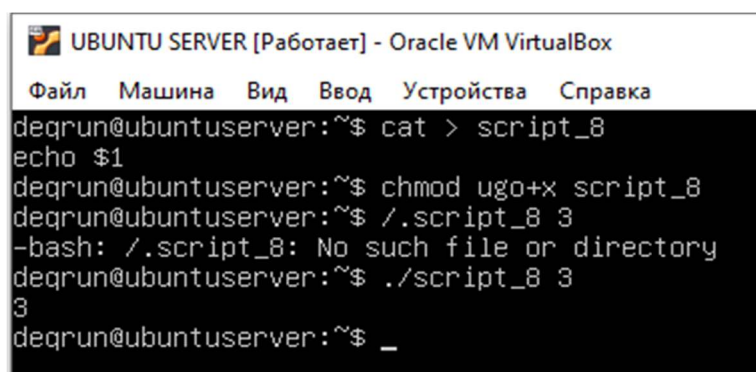
```
deqrun@ubuntuserver:~$ F=sort
deqrun@ubuntuserver:~$ $F command_six.txt
KIRENSKIY DANILA LAB 4
TEST FILE FOR COMMAND NUMBER SIX
deqrun@ubuntuserver:~$ cat command_six.txt
TEST FILE FOR COMMAND NUMBER SIX
KIRENSKIY DANILA LAB 4
deqrun@ubuntuserver:~$ _
```

Рисунок 7 – присваивание переменной F значение команды sort.

2. II Часть

Написать скрипты, при запуске которых выполняются следующие действия:

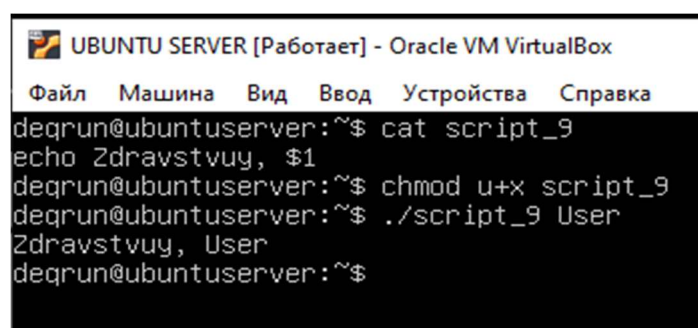
8. Программа запрашивает значение переменной, а затем выводит значение этой переменной. Чтобы сделать файл исполняемым в linux выполните `chmod u+x файл_скрипта`.



```
deqrun@ubuntuserver:~$ cat > script_8
deqrun@ubuntuserver:~$ echo $1
deqrun@ubuntuserver:~$ chmod ugo+x script_8
deqrun@ubuntuserver:~$ ./script_8 3
-bash: ./script_8: No such file or directory
deqrun@ubuntuserver:~$ ./script_8 3
3
deqrun@ubuntuserver:~$ _
```

Рисунок 8 – запись скрипта в файл.

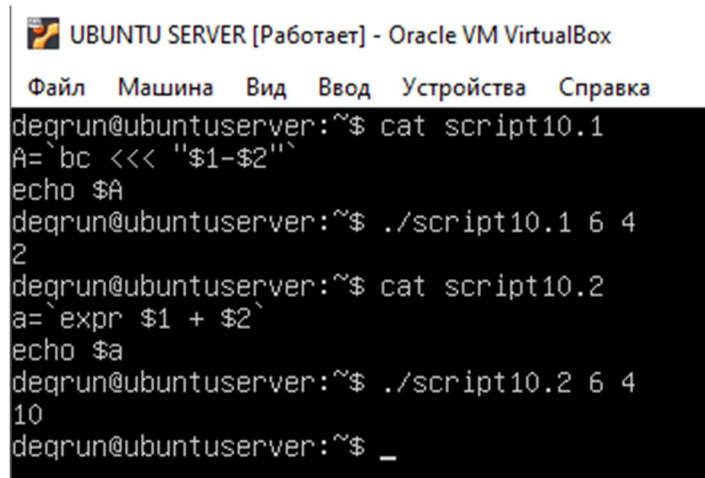
9. Программа запрашивает имя пользователя, затем здоровается с ним, используя значение введенной переменной.



```
deqrun@ubuntuserver:~$ cat script_9
deqrun@ubuntuserver:~$ echo Zdravstvuy, $1
deqrun@ubuntuserver:~$ chmod u+x script_9
deqrun@ubuntuserver:~$ ./script_9 User
Zdravstvuy, User
deqrun@ubuntuserver:~$
```

Рисунок 9 – скрипт приветствует пользователя.

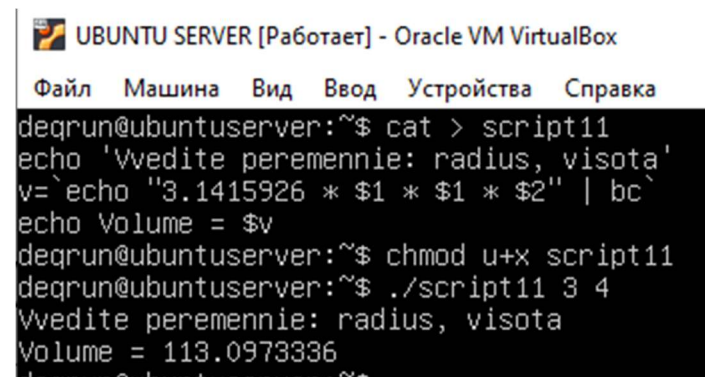
10. Программа запрашивает значения двух переменных, вычисляет сумму (разность, произведение, деление) этих переменных. Результат выводится на экран (использовать команды а) EXPR; б) BC)).



```
degrun@ubuntuserver:~$ cat script10.1
A=`bc <<< "$1-$2"`
echo $A
degrun@ubuntuserver:~$ ./script10.1 6 4
2
degrun@ubuntuserver:~$ cat script10.2
a=`expr $1 + $2`
echo $a
degrun@ubuntuserver:~$ ./script10.2 6 4
10
degrun@ubuntuserver:~$ _
```

Рисунок10 – задание 10.

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



```
degrun@ubuntuserver:~$ cat > script11
echo 'Vvedite peremennie: radius, visota'
v=`echo "3.1415926 * $1 * $1 * $2" | bc`
echo Volume = $v
degrun@ubuntuserver:~$ chmod u+x script11
degrun@ubuntuserver:~$ ./script11 3 4
Vvedite peremennie: radius, visota
Volume = 113.0973336
degrun@ubuntuserver:~$
```

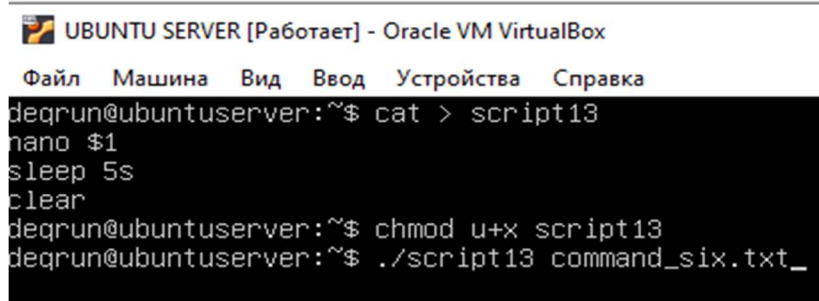
Рисунок11 – объем цилиндра.

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

```
degrun@ubuntuserver:~$ cat > script12
printf 'Programm name - %s. \nArguments quantity - %s. \n' $0 $#
printf 'Argument value - %s \n' $@
degrun@ubuntuserver:~$ ./script12 2 3 6 7 9 10 14
Programm name - ./script12.
Arguments quantity - 7.
Argument value - 2
Argument value - 3
Argument value - 6
Argument value - 7
Argument value - 9
Argument value - 10
Argument value - 14
degrun@ubuntuserver:~$ _
```

Рисунок12 – задание 12.

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



```
UBUNTU SERVER [Работает] - Oracle VM VirtualBox
Файл  Машина  Вид  Ввод  Устройства  Справка
degrun@ubuntuserver:~$ cat > script13
nano $1
sleep 5s
clear
degrun@ubuntuserver:~$ chmod u+x script13
degrun@ubuntuserver:~$ ./script13 command_six.txt_
```

Рисунок13 – исполняемый скрипт.

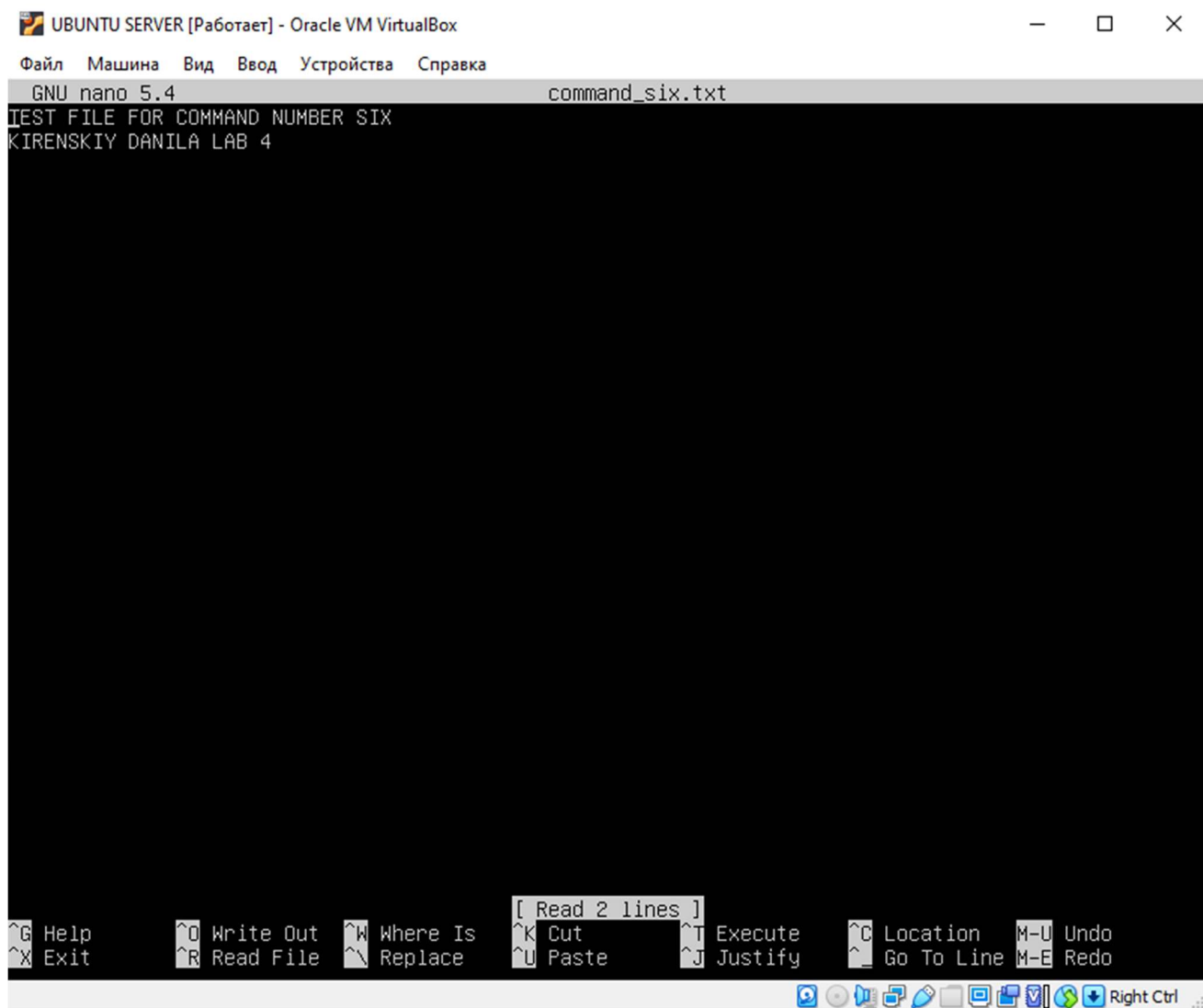


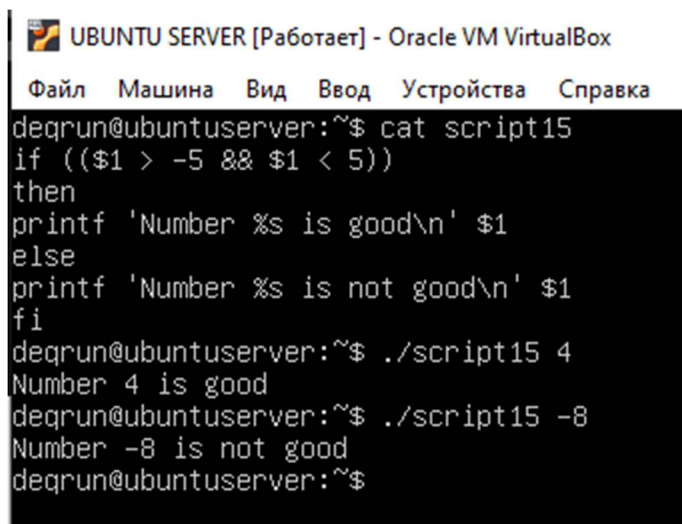
Рисунок 14 – текстовый редактор nano и выбранный файл.

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

```
UBUNTU SERVER [Работает] - Oracle VM VirtualBox
Файл  Машина  Вид  Ввод  Устройства  Справка
degrun@ubuntuserver:~$ cat > script13
for file in ./*.txt
do
cat $file
done
degrun@ubuntuserver:~$ chmod u+x script13
degrun@ubuntuserver:~$ ./script13
TEST FILE FOR COMMAND NUMBER SIX
KIRENSKIY DANILA LAB 4
degrun@ubuntuserver:~$
```

Рисунок 15 – результат скрипта.

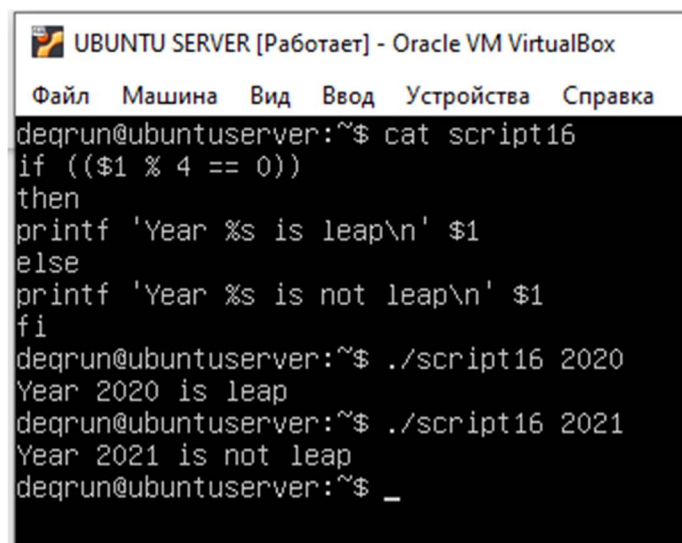
15. Программой запрашивается ввод числа, значение которого затем сравнивается с допустимым значением. В результате этого сравнения на экран выдаются соответствующие сообщения.



```
UBUNTU SERVER [Работает] - Oracle VM VirtualBox
Файл  Машина  Вид  Ввод  Устройства  Справка
degrun@ubuntuserver:~$ cat script15
if (($1 > -5 && $1 < 5))
then
printf 'Number %s is good\n' $1
else
printf 'Number %s is not good\n' $1
fi
degrun@ubuntuserver:~$ ./script15 4
Number 4 is good
degrun@ubuntuserver:~$ ./script15 -8
Number -8 is not good
degrun@ubuntuserver:~$
```

Рисунок 16 – результат скрипта.

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



```
UBUNTU SERVER [Работает] - Oracle VM VirtualBox
Файл  Машина  Вид  Ввод  Устройства  Справка
degrun@ubuntuserver:~$ cat script16
if (($1 % 4 == 0))
then
printf 'Year %s is leap\n' $1
else
printf 'Year %s is not leap\n' $1
fi
degrun@ubuntuserver:~$ ./script16 2020
Year 2020 is leap
degrun@ubuntuserver:~$ ./script16 2021
Year 2021 is not leap
degrun@ubuntuserver:~$ _
```

Рисунок 17 – результат скрипта.

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

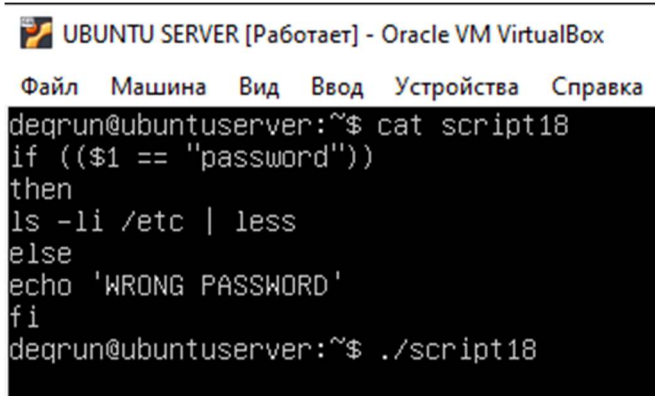
```

degrun@ubuntuserver:~$ ./script17
Range from i to j - 2 7
Values x and y - 3 4
Values in range. Increment...
x = 4, y = 5
Values in range. Increment...
x = 5, y = 6
Values in range. Increment...
x = 6, y = 7
Results: x= 6, y = 7
degrun@ubuntuserver:~$ cat script17
read -p "Range from i to j - " i j
read -p "Values x and y - " x y
while (($i < $x && $i < $y && $j > $x && $j > $y))
do
printf 'Values in range. Increment...\n'
x=$((x+1))
y=$((y+1))
printf 'x = %s, y = %s\n' $x $y
done
printf 'Results: x= %s, y = %s\n' $x $y
degrun@ubuntuserver:~$ _

```

Рисунок 18 – результат скрипта.

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

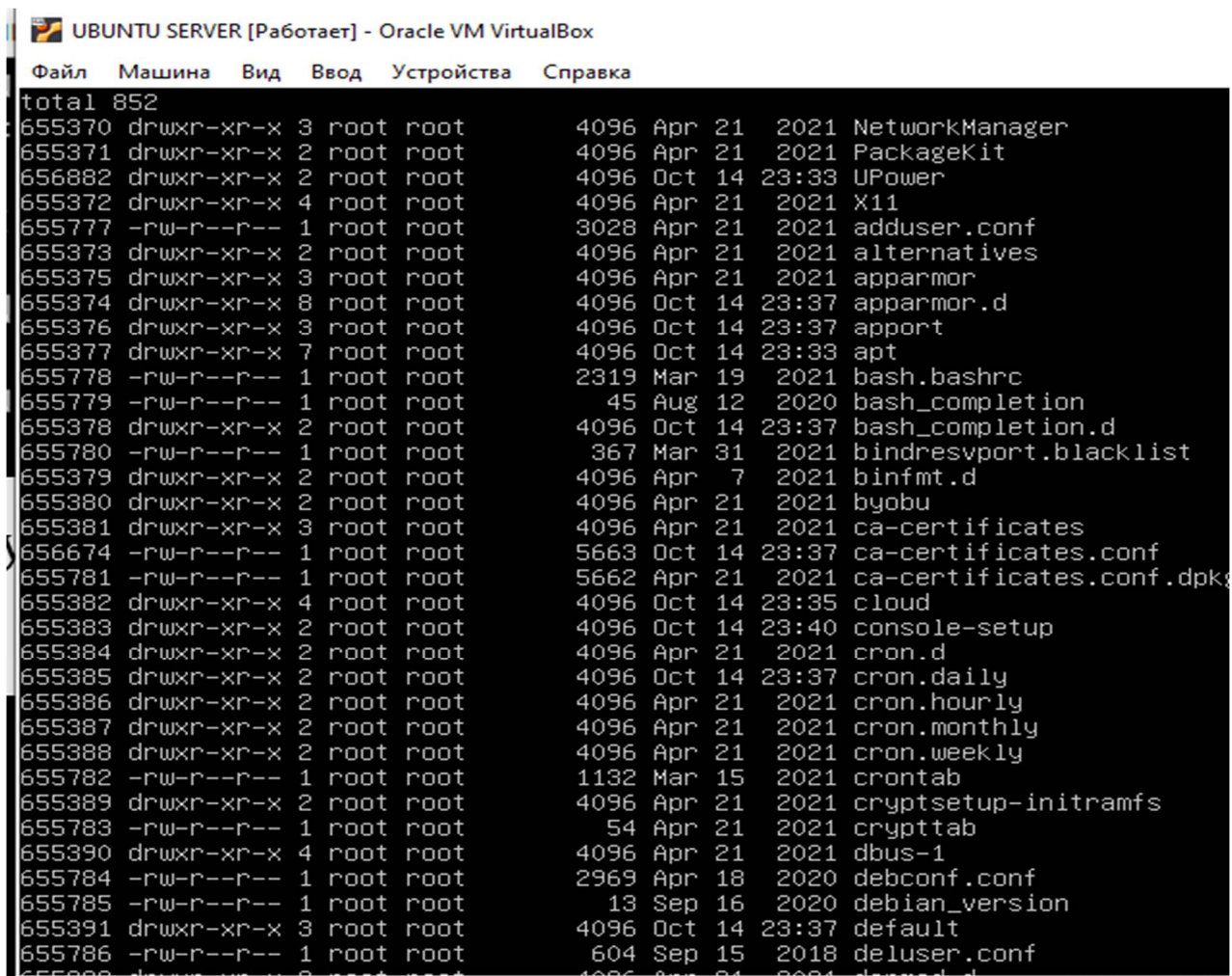


```

UBUNTU SERVER [Работает] - Oracle VM VirtualBox
Файл  Машина  Вид  Ввод  Устройства  Справка
degrun@ubuntuserver:~$ cat script18
if (( $1 == "password" ))
then
ls -li /etc | less
else
echo 'WRONG PASSWORD'
fi
degrun@ubuntuserver:~$ ./script18

```


Рисунок 19 – скрипт.



```
total 852
655370 drwxr-xr-x 3 root root      4096 Apr 21  2021 NetworkManager
655371 drwxr-xr-x 2 root root      4096 Apr 21  2021 PackageKit
656882 drwxr-xr-x 2 root root      4096 Oct 14 23:33 UPower
655372 drwxr-xr-x 4 root root      4096 Apr 21  2021 X11
655777 -rw-r--r-- 1 root root      3028 Apr 21  2021 adduser.conf
655373 drwxr-xr-x 2 root root      4096 Apr 21  2021 alternatives
655375 drwxr-xr-x 3 root root      4096 Apr 21  2021 apparmor
655374 drwxr-xr-x 8 root root      4096 Oct 14 23:37 apparmor.d
655376 drwxr-xr-x 3 root root      4096 Oct 14 23:37 apport
655377 drwxr-xr-x 7 root root      4096 Oct 14 23:33 apt
655778 -rw-r--r-- 1 root root     2319 Mar 19  2021 bash.bashrc
655779 -rw-r--r-- 1 root root        45 Aug 12  2020 bash_completion
655378 drwxr-xr-x 2 root root      4096 Oct 14 23:37 bash_completion.d
655780 -rw-r--r-- 1 root root       367 Mar 31  2021 bindresvport.blacklist
655379 drwxr-xr-x 2 root root      4096 Apr  7  2021 binfmt.d
655380 drwxr-xr-x 2 root root      4096 Apr 21  2021 byobu
655381 drwxr-xr-x 3 root root      4096 Apr 21  2021 ca-certificates
656674 -rw-r--r-- 1 root root     5663 Oct 14 23:37 ca-certificates.conf
655781 -rw-r--r-- 1 root root     5662 Apr 21  2021 ca-certificates.conf.dpkg
655382 drwxr-xr-x 4 root root      4096 Oct 14 23:35 cloud
655383 drwxr-xr-x 2 root root      4096 Oct 14 23:40 console-setup
655384 drwxr-xr-x 2 root root      4096 Apr 21  2021 cron.d
655385 drwxr-xr-x 2 root root      4096 Oct 14 23:37 cron.daily
655386 drwxr-xr-x 2 root root      4096 Apr 21  2021 cron.hourly
655387 drwxr-xr-x 2 root root      4096 Apr 21  2021 cron.monthly
655388 drwxr-xr-x 2 root root      4096 Apr 21  2021 cron.weekly
655782 -rw-r--r-- 1 root root     1132 Mar 15  2021 crontab
655389 drwxr-xr-x 2 root root      4096 Apr 21  2021 cryptsetup-initramfs
655783 -rw-r--r-- 1 root root        54 Apr 21  2021 crypttab
655390 drwxr-xr-x 4 root root      4096 Apr 21  2021 dbus-1
655784 -rw-r--r-- 1 root root     2969 Apr 18  2020 debconf.conf
655785 -rw-r--r-- 1 root root       13 Sep 16  2020 debian_version
655391 drwxr-xr-x 3 root root      4096 Oct 14 23:37 default
655786 -rw-r--r-- 1 root root       604 Sep 15  2018 deluser.conf
```

Рисунок 20 – результат скрипта.

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



```
degrun@ubuntuserver:~$ cat > script19
if [ -f $1 ]
then
echo "File exists"
cat $1
else
echo "File does not exist"
fi
degrun@ubuntuserver:~$ script19 command_six.txt
script19: command not found
degrun@ubuntuserver:~$ chmod u+x script19
degrun@ubuntuserver:~$ ./script19 command_six.txt
File exists
TEST FILE FOR COMMAND NUMBER SIX
KIRENSKIY DANILA LAB 4
degrun@ubuntuserver:~$
```

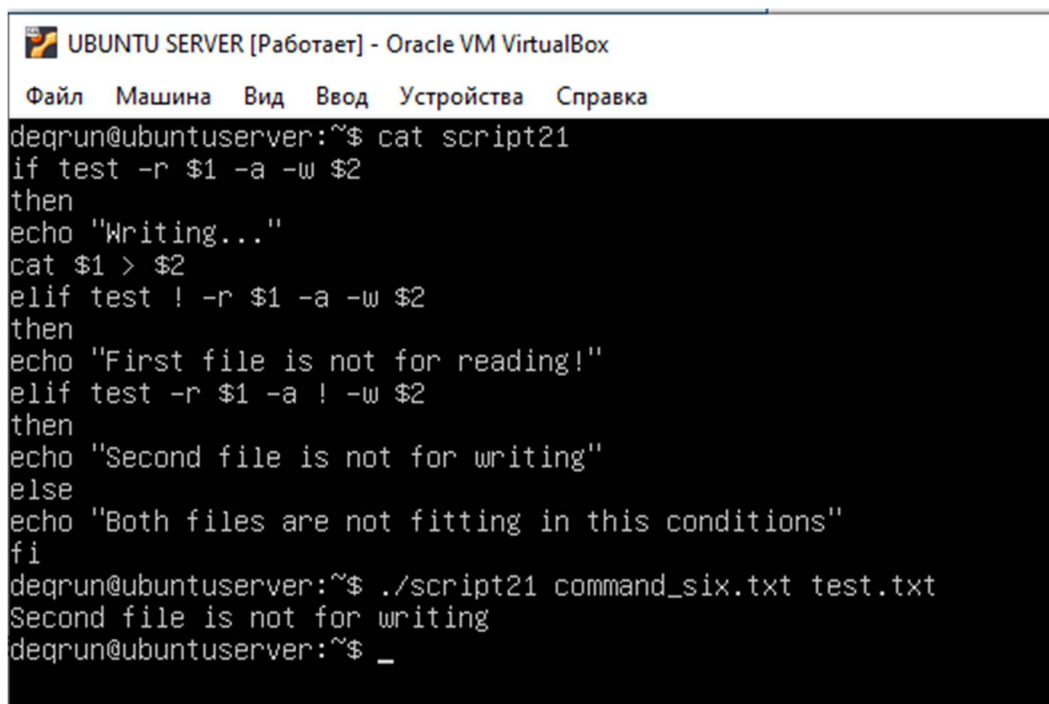
Рисунок 21 – результат скрипта.

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.

```
UBUNTU SERVER [Работает] - Oracle VM VirtualBox
Файл  Машина  Вид  Ввод  Устройства  Справка
deqrun@ubuntuserver:~$ cat script20_new
if [ -d $1 ]
then
echo "Directory exists"
ls $1
elif [ -f $1 ]
then
echo "This is file"
cat $1
else
echo "Let-s create directory."
mkdir -p $1
fi
deqrun@ubuntuserver:~$ ./script20_new directory20
Let-s create directory.
deqrun@ubuntuserver:~$ ./script20_new directory20
Directory exists
deqrun@ubuntuserver:~$
```

Рисунок 22 – результат скрипта.

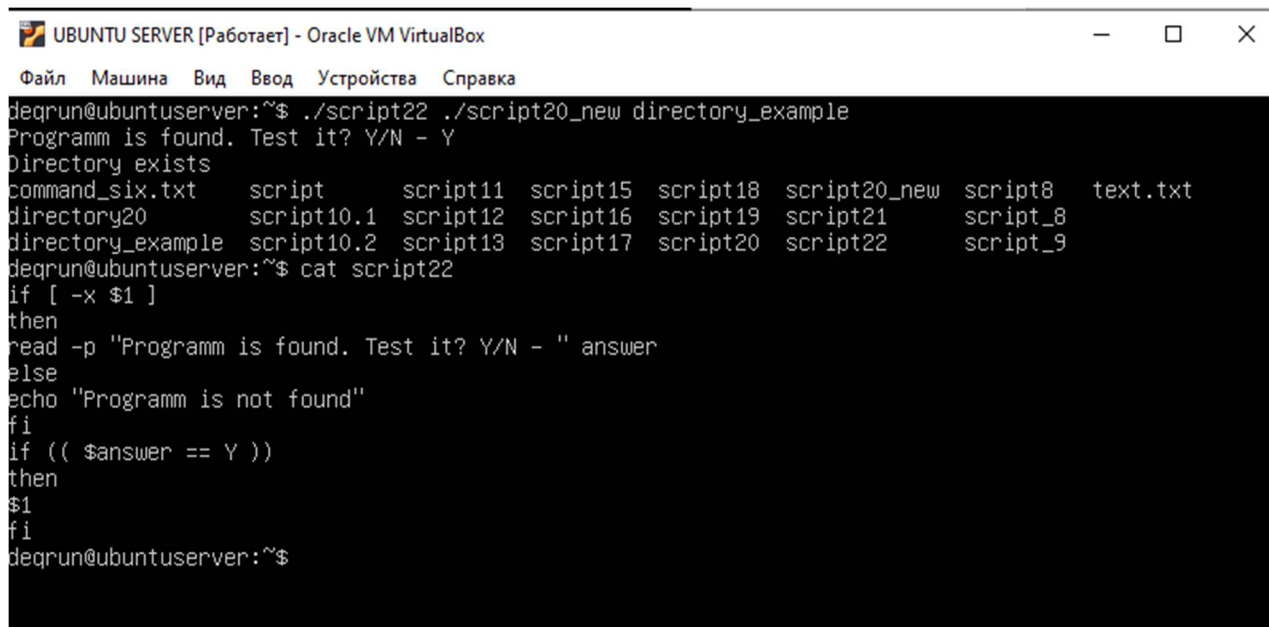
21. Анализируются атрибуты файла. Если первый файл существует и используется для чтения, а второй файл существует и используется для записи, то содержимое первого файла перенаправляется во второй файл. В случае несовпадений указанных атрибутов или отсутствия файлов на экран выдаются соответствующие сообщения (использовать а) имена файлов; б) позиционные параметры).



```
deqrun@ubuntuserver:~$ cat script21
if test -r $1 -a -w $2
then
echo "Writing..."
cat $1 > $2
elif test ! -r $1 -a -w $2
then
echo "First file is not for reading!"
elif test -r $1 -a ! -w $2
then
echo "Second file is not for writing"
else
echo "Both files are not fitting in this conditions"
fi
deqrun@ubuntuserver:~$ ./script21 command_six.txt test.txt
Second file is not for writing
deqrun@ubuntuserver:~$ _
```

Рисунок 23 – результат скрипта.

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



```
deqrun@ubuntuserver:~$ ./script22 ./script20_new directory_example
Programm is found. Test it? Y/N - Y
Directory exists
command_six.txt      script      script11    script15    script18    script20_new  script8    text.txt
directory20          script10.1  script12    script16    script19    script21      script_8
directory_example    script10.2  script13    script17    script20    script22      script_9
deqrun@ubuntuserver:~$ cat script22
if [ -x $1 ]
then
read -p "Programm is found. Test it? Y/N - " answer
else
echo "Programm is not found"
fi
if (( $answer == Y ))
then
$1
fi
deqrun@ubuntuserver:~$
```

Рисунок 24 – результат скрипта.

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.


```

test@ubuntu:~/new$ cat test.txt
One One Two
Abc der tru
WWW WWW QQQ

test@ubuntu:~/new$ ./script23 test.txt test2.txt

Abc der tru
One One Two
WWW WWW QQQ
test@ubuntu:~/new$ cat script23
size=$(wc -c $1 | awk '{print $1}')
if ((size>0))
then
sort -k1 $1
cat $1 > $2
else
echo "Oops, it's zero-file"
fi

```

Рисунок 25 – результат скрипта.

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

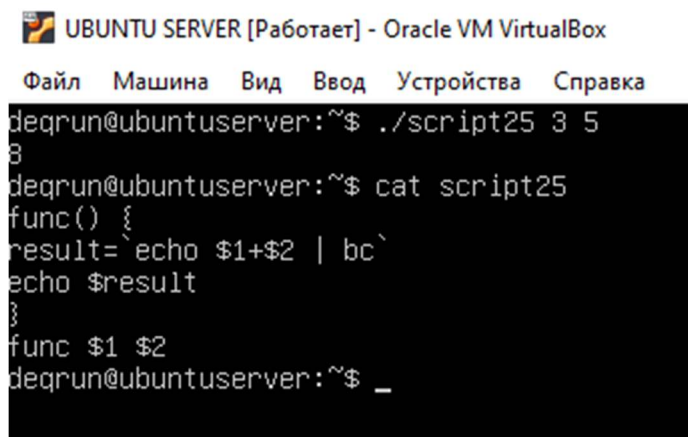
```

degrun@ubuntuserver:~$ cat > script24
echo "Zipping..."
tar -cvf my.tar *.txt
echo "Looking..."
tar -tf my.tar
echo "Pressing..."
gzip -c my.tar > my.gz
echo "Ready!"
degrun@ubuntuserver:~$ chmod u+x script24
degrun@ubuntuserver:~$ ./script24
Zipping...
command_six.txt
text.txt
Looking...
command_six.txt
text.txt
Pressing...
Ready!
degrun@ubuntuserver:~$

```

Рисунок 26 – результат скрипта.

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



```
UBUNTU SERVER [Работает] - Oracle VM VirtualBox
Файл  Машина  Вид  Ввод  Устройства  Справка
degrun@ubuntuserver:~$ ./script25 3 5
8
degrun@ubuntuserver:~$ cat script25
func() {
result=`echo $1+$2 | bc`
echo $result
}
func $1 $2
degrun@ubuntuserver:~$ _
```

Рисунок 27 – результат скрипта.

Заключение

В ходе данной лабораторной работы были изучены или повторно рассмотрены некоторые команды ОС Linux, было проведено ознакомление и анализ рекомендованной литературы, а также информации о средствах управления процессами ОС Ubuntu. Также было произведено изучение основных возможностей языка программирования Shellc целью автоматизации процесса администрирования системы за счет написания и использования командных файлов.