

Липецкий государственный технический университет

Факультет автоматизации и информатики

Кафедра Автоматизированных систем управления

ЛАБОРАТОРНАЯ РАБОТА №5

по курсу "Операционная система Linux"

Программирование на “SHELL”

Студент

Осипов А. А.

Группа АИ-18

Руководитель

Кургасов В.В.

Доцент

Липецк 2020г.

Цель лабораторной работы

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

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.

Ход работы

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



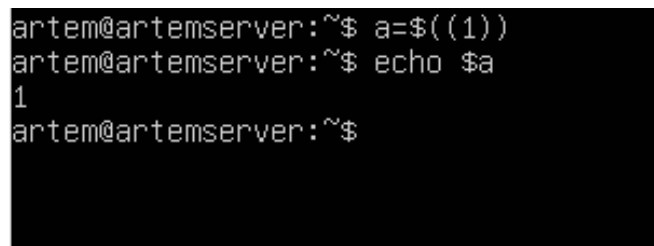
The image shows two terminal windows. The top window is a nano editor titled 'GNU nano 4.8' containing the following code: `echo Hello`, `printf World`, and `echo`. The bottom window shows the execution of a script named 'script' on a system named 'artemserver'. The output of the script is 'Hello World' followed by 'Hello' on a new line.

```
GNU nano 4.8
echo Hello
printf World
echo

artem@artemserver:~$ sh script
Hello World
Hello
artem@artemserver:~$ _
```

Рисунок 1 – Выполнение задания 1

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

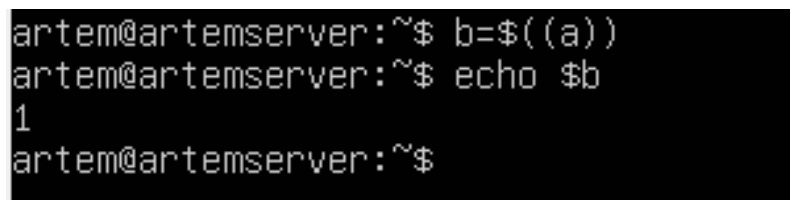


The image shows a terminal window on 'artemserver' where a variable 'a' is assigned the value 1 using `a=$((1))`. Then, the value of 'a' is printed using `echo $a`, resulting in the output '1'.

```
artem@artemserver:~$ a=$((1))
artem@artemserver:~$ echo $a
1
artem@artemserver:~$
```

Рисунок 2 – Выполнение задания 2

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



The image shows a terminal window on 'artemserver' where a variable 'b' is assigned the value of 'a' using `b=$((a))`. Then, the value of 'b' is printed using `echo $b`, resulting in the output '1'.

```
artem@artemserver:~$ b=$((a))
artem@artemserver:~$ echo $b
1
artem@artemserver:~$
```

Рисунок 3 – Выполнение задания 3

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

```
artem@artemserver:~$ c=pwd
artem@artemserver:~$ $c
/home/artem
artem@artemserver:~$
```

Рисунок 4 – Выполнение задания 4

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

```
artem@artemserver:~$ d=$(date)
artem@artemserver:~$ echo $d
Fri 27 Nov 2020 04:28:01 PM UTC
artem@artemserver:~$
```

Рисунок 5 – Выполнение задания 5

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

```
artem@artemserver:~$ f='cat names'
artem@artemserver:~$ $f
artem
dima
anna
sveta
maksim
artem@artemserver:~$
```

Рисунок 6 – Выполнение задания 6

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

```
artem@artemserver:~$ f='sort'
artem@artemserver:~$ $f names
anna
artem
dima
maksim
sveta
artem@artemserver:~$ _
```

Рисунок 7 – Выполнение задания 7

Написать скрипты, при запуске которых выполняются следующие действия:

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

```
GNU nano 4.8
#!/bin/bash
read A
echo $A
```

Рисунок 8 – Скрипт для задания 8

```
artem@artemserver:~$ sh script
Hi
Hi
artem@artemserver:~$
```

Рисунок 9 – Результат выполнения скрипта задания 8

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


```

GNU nano 4.8
#!/bin/bash
echo "What is your name?"
read name
echo "Hi $name nice to meet you)"

```

Рисунок 10 – Скрипт для задания 9

```

artem@artemserver:~$ sh script2
What is your name?
Artem
Hi Artem nice to meet you)
artem@artemserver:~$

```

Рисунок 11 – Результат выполнения скрипта для задания 9

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

А)

```

GNU nano 4.8
#!/bin/bash
echo "enter a:"
read a
echo "enter b:"
read b
sum=$(expr $a + $b)
echo "$a + $b = $sum"
razn=$(expr $a - $b)
echo "$a - $b = $razn"
umn=$(expr $a \* $b)
echo "$a * $b = $umn"
del=$(expr $a / $b)
echo "$a / $b = $del"

```

Рисунок 12– Скрипт для задания 10а

```

artem@artemserver:~$ sh script
enter a:
9
enter b:
3
9 + 3 = 12
9 - 3 = 6
9 * 3 = 27
9 / 3 = 3
artem@artemserver:~$

```

Рисунок 13 – Результат выполнения скрипта для задания 10а

Б)

```

GNU nano 4.8
#!/bin/bash
echo "enter a:"
read a
echo "enter b:"
read b
echo "$a + $b" | bc
echo "$a - $b" | bc
echo "$a * $b" | bc
echo "$a / $b" | bc_

```

Рисунок 14 – Скрипт для задания 10б

```

artem@artemserver:~$ sh script
enter a:
12
enter b:
4
16
8
48
3
artem@artemserver:~$ _

```

Рисунок 15 – Результат выполнения скрипта для задания 10б

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

```

GNU nano 4.8
#!/bin/bash
echo "enter S:"
read S
echo "enter h:"
read h
V=$(echo "$S * $h" | bc)
echo "V = $V"

```

Рисунок 16 – Скрипт для задания 11

```

artem@artemserver:~$ sh script
enter S:
3
enter h:
4
V = 12
artem@artemserver:~$

```

Рисунок 17 – Результат выполнения скрипта для задания 11

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

```

GNU nano 4.8 script
#!/bin/bash
echo "name programm $0, number of arguments $#"
```

```

for arg in $@
do
echo $arg
done

```

Рисунок 18 – Скрипт для задания 12

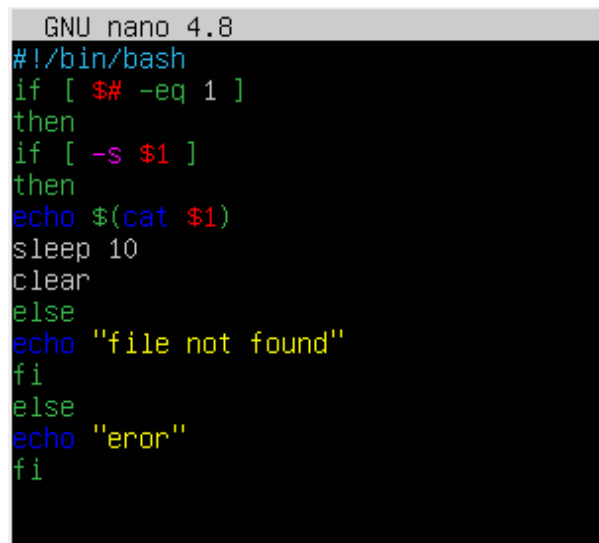
```

artem@artemserver:~$ sh script 22 33 hi
name programm script, number of arguments 3
22
33
hi
artem@artemserver:~$

```

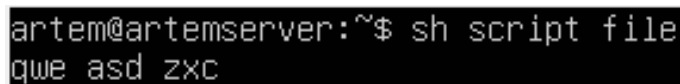
Рисунок 19 – Результат выполнения скрипта для задания 12

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



```
GNU nano 4.8
#!/bin/bash
if [ $# -eq 1 ]
then
if [ -s $1 ]
then
echo $(cat $1)
sleep 10
clear
else
echo "file not found"
fi
else
echo "error"
fi
```

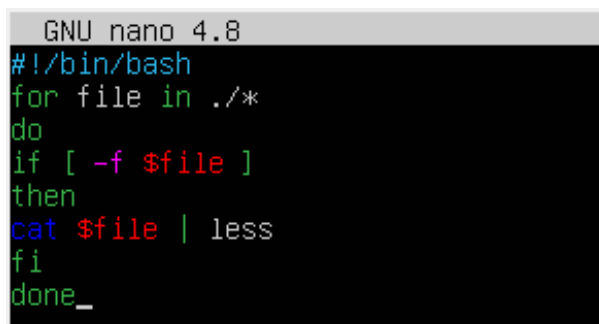
Рисунок 20 – Скрипт для задания 13



```
artem@artemserver:~$ sh script file
qwe asd zxc
```

Рисунок 21 – Результат выполнения скрипта для задания 13

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



```
GNU nano 4.8
#!/bin/bash
for file in ./*
do
if [ -f $file ]
then
cat $file | less
fi
done_
```

Рисунок 22 – Скрипт для задания 14

```
artem@artemserver:~$ sh script
qwe
asd
zxc
(END)
```

Рисунок 23 – Результат выполнения скрипта для задания 14

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

```
GNU nano 4.8
#!/bin/bash
echo "enter a:"
read a
if [ $a -ne 5 ]
then
echo "$a != 5"
else
echo "$a == 5"
fi
```

Рисунок 24 – Скрипт для задания 15

```
artem@artemserver:~$ sh script
enter a:
6
6 != 5
artem@artemserver:~$ sh script
enter a:
5
5 == 5
artem@artemserver:~$
```

Рисунок 25 – Результат выполнения скрипта для задания 15

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

```

GNU nano 4.8
#!/bin/bash
echo "enter year:"
read Year
if [ `expr $Year % 4` -eq 0 ]
then
if [ `expr $Year % 100` -ne 0 ]
then
echo "yes"
else
echo "no"
fi
elif [ `expr $Year % 400` -eq 0 ]
then
echo "yes"
else
echo "no"
fi

```

Рисунок 26 – Скрипт для задания 16

```

artem@artemserver:~$ sh script
enter year:
2004
yes
artem@artemserver:~$ sh script
enter year:
2007
no
artem@artemserver:~$

```

Рисунок 27 – Результат выполнения скрипта для задания 16

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

```

GNU nano 4.8 script
#!/bin/bash
echo "enter a & b:"
read a
read b
echo "enter dada range:"
read left
read right
while [ $left -gt $a ] && [ $left -gt $b ] && [ $right -gt $a ] && [ $right -gt $b ]
do
a=$(expr $a + 1)
b=$(expr $b + 1)
done
echo "a = $a; b = $b"

```

Рисунок 28 – Скрипт для задания 17

```
artem@artemserver:~$ sh script
enter a & b:
2
4
enter dada range:
5
7
a = 3; b = 5
artem@artemserver:~$ _
```

Рисунок 29 – Результат выполнения скрипта для задания 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.

```
GNU nano 4.8
#!/bin/bash
echo "enter password:"
read pas
d="qwer123"
if [ $d = $pas ]
then
ls -al /etc |less
else
echo "wrong password"
fi
```

Рисунок 30 – Скрипт для задания 18

```
artem@artemserver:~$ sh script
enter password:
qwer123
```

Рисунок 31 – Ввод пароля

```

total 796
drwxr-xr-x 93 root root      4096 Nov 27 15:08 .
drwxr-xr-x 20 root root      4096 Oct 28 07:22 ..
-rw-r--r--  1 root root      3028 Jul 31 16:28 adduser.conf
drwxr-xr-x  2 root root      4096 Jul 31 16:29 alternatives
drwxr-xr-x  3 root root      4096 Jul 31 16:29 apparmor
drwxr-xr-x  7 root root      4096 Jul 31 16:29 apparmor.d
drwxr-xr-x  3 root root      4096 Nov 27 15:06 apport
drwxr-xr-x  7 root root      4096 Oct 28 07:21 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 27 15:06 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 binfo.d
drwxr-xr-x  2 root root      4096 Jul 31 16:29 byobu
drwxr-xr-x  3 root root      4096 Jul 31 16:28 ca-certificates
-rw-r--r--  1 root root     6505 Nov  6 09:30 ca-certificates.conf
-rw-r--r--  1 root root     5714 Jul 31 16:29 ca-certificates.conf.dpkg-old
drwxr-xr-x  2 root root      4096 Jul 31 16:29 calendar
drwxr-xr-x  4 root root      4096 Oct 28 07:23 cloud
drwxr-xr-x  2 root root      4096 Oct 28 07:24 console-setup
drwxr-xr-x  2 root root      4096 Jul 31 16:29 cron.d
drwxr-xr-x  2 root root      4096 Nov 27 15:06 cron.daily
drwxr-xr-x  2 root root      4096 Jul 31 16:28 cron.hourly
drwxr-xr-x  2 root root      4096 Jul 31 16:28 cron.monthly
-rw-r--r--  1 root root     1042 Feb 13  2020 crontab
drwxr-xr-x  2 root root      4096 Jul 31 16:30 cron.weekly
drwxr-xr-x  2 root root      4096 Nov 27 15:04 cryptsetup-initramfs
-rw-r--r--  1 root root      54 Jul 31 16:29 crypttab
drwxr-xr-x  4 root root      4096 Jul 31 16:28 dbus-1
drwxr-xr-x  3 root root      4096 Jul 31 16:29 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 27 15:08 default
-rw-r--r--  1 root root      604 Sep 15  2018 deluser.conf
:_

```

Рисунок 32 – Результат выполнения скрипта для задания 18

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

```

GNU nano 4.8
#!/bin/bash
echo "enter filename:"
read filename
if [ -e $filename ]
then
cat $filename
else
echo "error"
fi

```

Рисунок 33 – Скрипт для задания 19


```

artem@artemserver:~$ sh script
enter filename:
file
qwe
asd
zxc
artem@artemserver:~$ sh script
enter filename:
df
error
artem@artemserver:~$

```

Рисунок 34 – Результат выполнения скрипта для задания 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.

```

GNU nano 4.8
#!/bin/bash
echo "enter file:"
read filename
if [ -e $filename ]
then
if [ -d $filename ]
then
if [ -r $filename ]
then
ls $filename
else
echo "error"
fi
else
cat $filename
fi
else
mkdir $filename
fi_

```

Рисунок 35 – Скрипт для задания 20

```

artem@artemserver:~$ ls
channel  FILE    file_2  loop    loop.sh  res    script
file    file_1  file2   loop2.sh new      ress   script2
artem@artemserver:~$ sh script
enter file:
file
qwe
asd
zxc
artem@artemserver:~$ sh script
enter file:
fg
artem@artemserver:~$ ls
channel  file  file_1  file2  loop2.sh  new  ress  script2
fg      FILE  file_2  loop   loop.sh   res  script
artem@artemserver:~$

```

Рисунок 36 – Результат выполнения скрипта для задания 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).

```
GNU nano 4.8 script
#!/bin/bash
echo "enter file1 and file2:"
read f1
read f2
if [ -e $f1 ]
then
if [ -e $f2 ]
then
if [ -r $f1 ]
then
if [ -w $f2 ]
then
echo "$f1:"
cat $f1
echo "$f2:"
cat $f2
cat $f1 > $f2
echo "completed"
echo "$f2:"
cat $f2
else
echo "$f2 not available for recording"
fi
else
echo "$f1 not available for reading"
fi
else
echo "$f2 not found"
fi
else
echo "$f1 not found"
fi
```

Рисунок 37 – Скрипт для зтфтадания 21а

```
artem@artemserver:~$ sh script
enter file1 and file2:
file
file2
file:
qwe
asd
zxc
file2:
Hello
completed
file2:
qwe
asd
zxc
artem@artemserver:~$ _
```

Рисунок 38 – Результат выполнения скрипта для задания 21а

```

GNU nano 4.8
#!/bin/bash
if [ $# = 2 ]
then
if [ -e $1 ]
then
if [ -e $2 ]
then
if [ -r $1 ]
then
if [ -w $2 ]
then
echo "$1:"
cat $1
echo "$2:"
cat $2
cat $1 > $2
echo "completed"
echo "$2:"
cat $2
else
echo "$2 not available for recording"
fi
else
echo "$1 not available for reading"
fi
else
echo "$2 not found"
fi
else
echo "$1 not found"
fi

```

Рисунок 39 – Скрипт для задания 216

```

artem@artemserver:~$ sh script file file2
file:
qwe
asd
zxc
file2:
qwe
asd
zxc
completed
file2:
qwe
asd
zxc
artem@artemserver:~$

```

Рисунок 40 – Результат выполнения скрипта для задания 216

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

```

GNU nano 4.8
#!/bin/bash
if [ $# = 1 ]
then
    if [ -e $1 ] && [ -x $1 ]
    then
        sh_$1
    else
        echo "Error"
    fi
else
    echo "error"
fi

```

Рисунок 41 – Скрипт для задания 22

```

artem@artemserver:~$ sh script script2
What is your name?
Artem
Hi Artem nice to meet you)
artem@artemserver:~$ _

```

Рисунок 42 – Результат выполнения скрипта для задания 22

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.

```

GNU nano 4.8
#!/bin/bash
if [ $# = 1 ]
then
    if [ -s $1 ]
    then
        sort -k1 $1 > file2
        cat file2
    else echo "$1 empty"
    fi
else
    echo "error"
fi_

```

Рисунок 43 – Скрипт для задания 23

```
artem@artemserver:~$ sh script names
anna
artem
dima
maksim
sveta
artem@artemserver:~$ cat file2
anna
artem
dima
maksim
sveta
artem@artemserver:~$
```

Рисунок 44 – Результат выполнения скрипта для задания 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.

```
GNU nano 4.8
#!/bin/bash
p=$(find . -type f -maxdepth 1)
arch="archs.tar"
tar -cf $arch $p
sleep 10
tar -tf $arch
gzip $arch_
```

Рисунок 45 – Скрипт для задания 24

```

artem@artemserver:~$ sh script
find: warning: you have specified the global option -maxdepth after the argument -type, but global options are not positional, i.e., -maxdepth affects tests specified before it as well as those specified after it. Please specify global options before other arguments.
./bashrc
./profile
./file_1
./script2
./file
./loop
./file2
./res
./viminfo
./bash_history
./ress
./loop2.sh
./loop.sh
./bash_logout
./file2.swp
./file_2
./script.swp
./script2.swp
./script
./names
./sudo_as_admin_successful
artem@artemserver:~$

```

Рисунок 46 – Результат выполнения скрипта для задания 24

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

```

GNU nano 4.8
#!/bin/bash
echo "enter a b:"
read a b
summa() {
echo $((a + b))
}
s=$(summa)
echo "$a + $b = $s"

```

Рисунок 47 – Скрипт для задания 25

```

artem@artemserver:~$ sh script
enter a b:
2 7
2 + 7 = 9
artem@artemserver:~$ _

```

Рисунок 48 – Результат выполнения скрипта для задания 25

Вывод

Я изучил основные возможности языка программирования Shell с целью автоматизации процесса администрирования системы за счет написания и использования командных файлов.