

# **LINUX ASSIGNMENT**

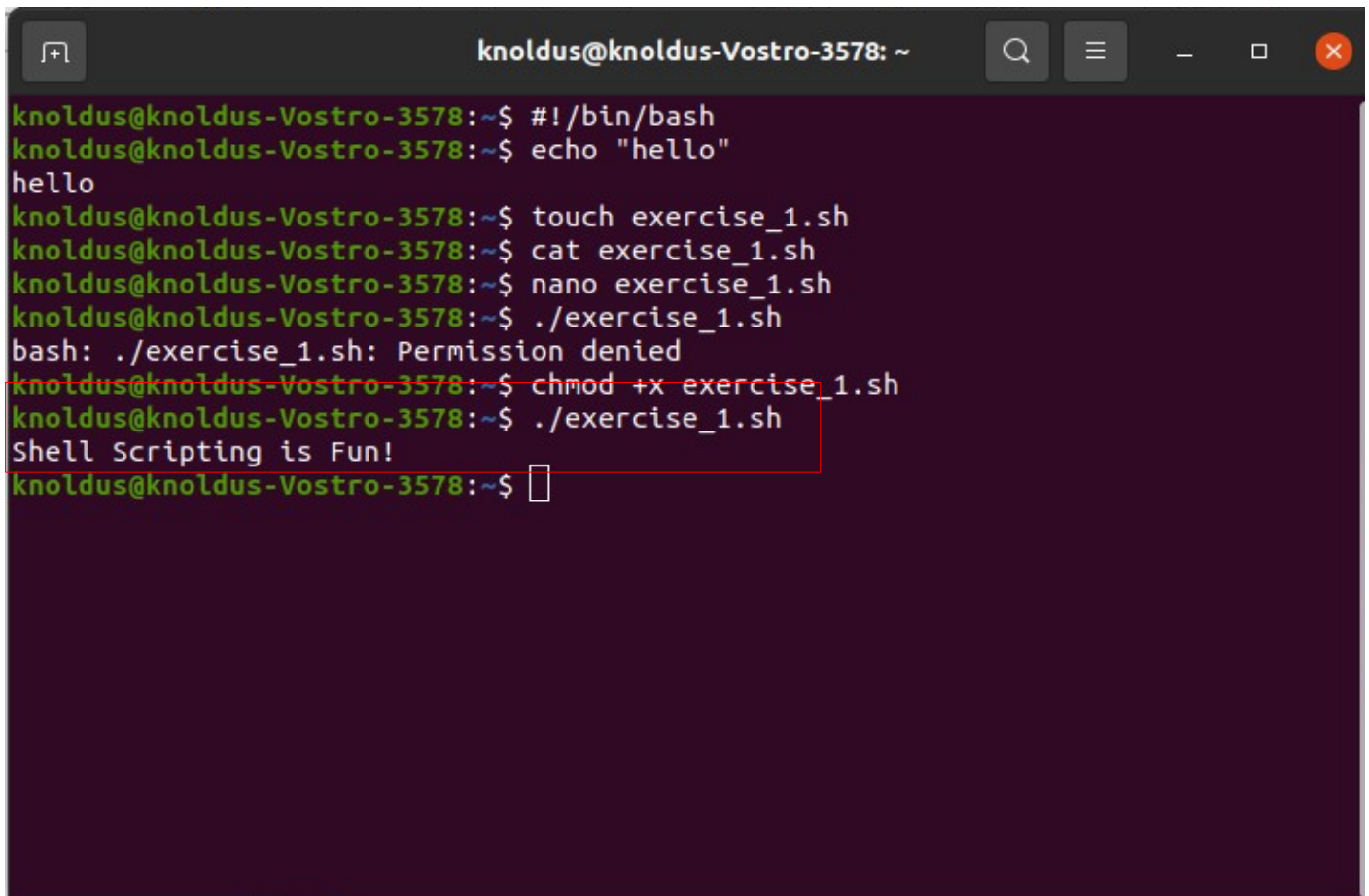
Submitted By:  
**Anuj Tomar**

Employee ID:  
**1733**

**Exercise\_1 - Write a shell script that prints "Shell Scripting is Fun!" on the screen.**

**Solution: Code:**

echo "Shell Scripting is Fun!"

A terminal window titled 'knoldus@knoldus-Vostro-3578: ~' with standard window controls. The terminal shows a series of commands and their outputs. The user creates a file 'exercise\_1.sh', views its contents, and attempts to run it. The first run fails with a 'Permission denied' error. The user then uses 'chmod +x' to make the script executable and runs it again, which successfully outputs 'Shell Scripting is Fun!'.

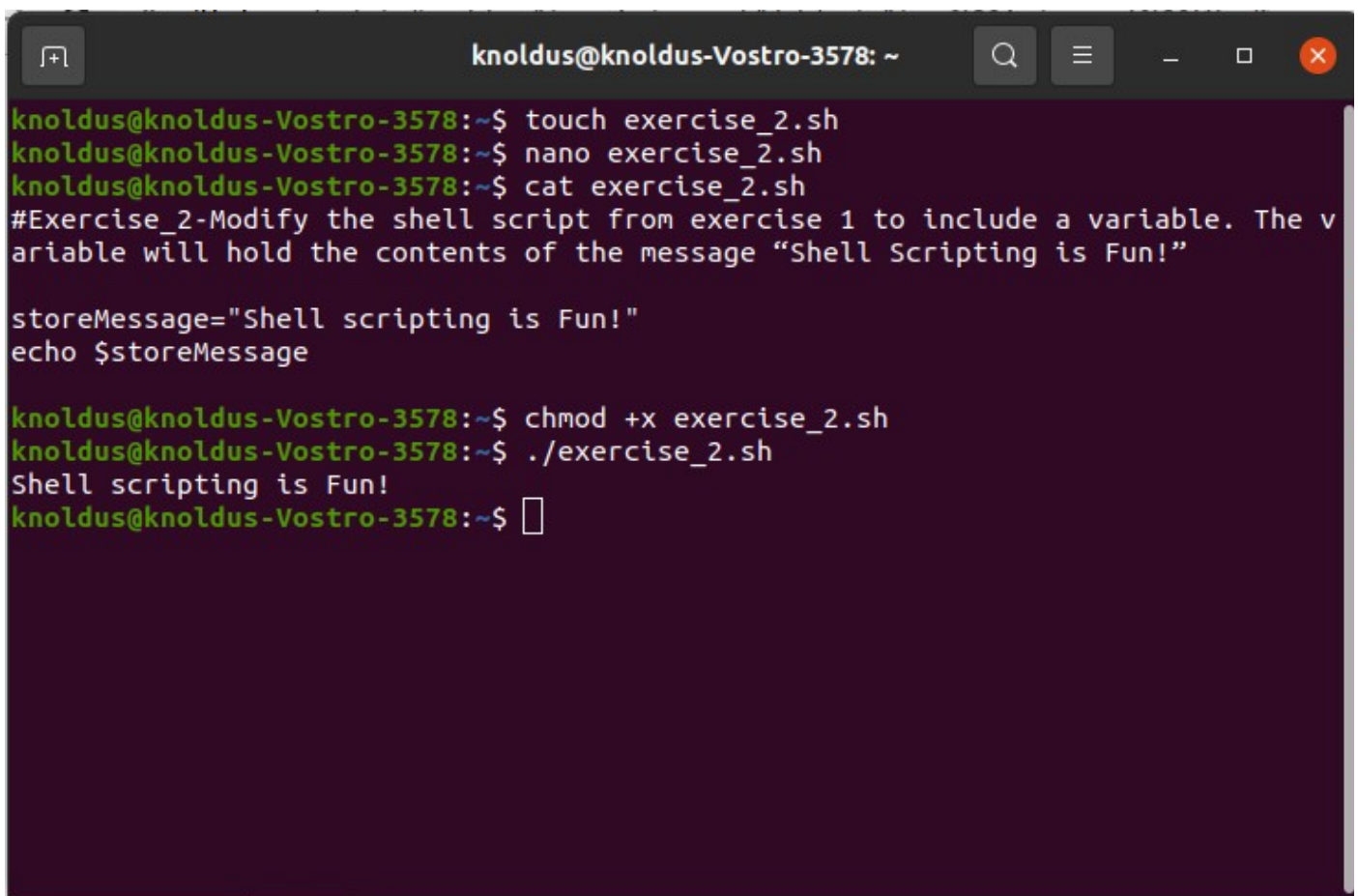
```
knoldus@knoldus-Vostro-3578:~$ #!/bin/bash
knoldus@knoldus-Vostro-3578:~$ echo "hello"
hello
knoldus@knoldus-Vostro-3578:~$ touch exercise_1.sh
knoldus@knoldus-Vostro-3578:~$ cat exercise_1.sh
knoldus@knoldus-Vostro-3578:~$ nano exercise_1.sh
knoldus@knoldus-Vostro-3578:~$ ./exercise_1.sh
bash: ./exercise_1.sh: Permission denied
knoldus@knoldus-Vostro-3578:~$ chmod +x exercise_1.sh
knoldus@knoldus-Vostro-3578:~$ ./exercise_1.sh
Shell Scripting is Fun!
knoldus@knoldus-Vostro-3578:~$
```

**Exercise\_2 - Modify the shell script from exercise 1 to include a variable. The variable will hold the contents of the message “Shell Scripting is Fun!”**

**Solution: Code:**

```
storeMessage = "shell scripting is Fun!"
```

```
echo $storeMessage
```

A terminal window titled 'knoldus@knoldus-Vostro-3578: ~' with standard window controls. The terminal shows the following commands and output:

```
knoldus@knoldus-Vostro-3578:~$ touch exercise_2.sh
knoldus@knoldus-Vostro-3578:~$ nano exercise_2.sh
knoldus@knoldus-Vostro-3578:~$ cat exercise_2.sh
#Exercise_2-Modify the shell script from exercise 1 to include a variable. The v
variable will hold the contents of the message "Shell Scripting is Fun!"

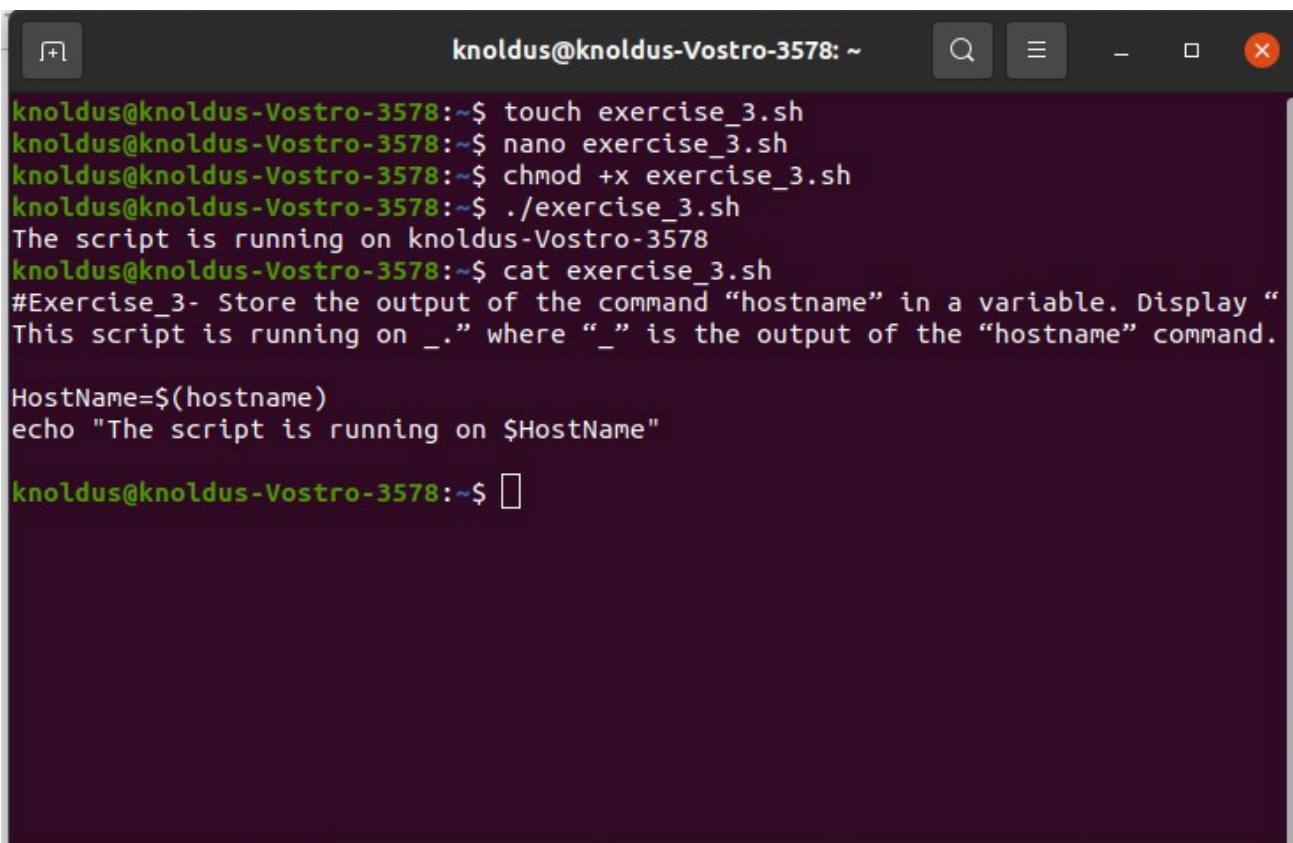
storeMessage="Shell scripting is Fun!"
echo $storeMessage

knoldus@knoldus-Vostro-3578:~$ chmod +x exercise_2.sh
knoldus@knoldus-Vostro-3578:~$ ./exercise_2.sh
Shell scripting is Fun!
knoldus@knoldus-Vostro-3578:~$
```

**Exercise\_3** - Store the output of the command “hostname” in a variable. Display “This script is running on \_.” where “\_” is the output of the “hostname” command.

**Solution: Code:**

```
HostName=$(hostname)
echo "The script is running on $HostName"
```

A terminal window titled 'knoldus@knoldus-Vostro-3578: ~' with standard window controls. The terminal shows a series of commands and their outputs. The user creates a file 'exercise\_3.sh', opens it with 'nano', sets permissions with 'chmod +x', and runs it with './exercise\_3.sh'. The script's output is 'The script is running on knoldus-Vostro-3578'. Then, the user runs 'cat exercise\_3.sh' to view the script's content, which includes a comment, the task description, and the shell code: 'HostName=\$(hostname)' and 'echo "The script is running on \$HostName"'. The prompt returns to the user's shell.

```
knoldus@knoldus-Vostro-3578:~$ touch exercise_3.sh
knoldus@knoldus-Vostro-3578:~$ nano exercise_3.sh
knoldus@knoldus-Vostro-3578:~$ chmod +x exercise_3.sh
knoldus@knoldus-Vostro-3578:~$ ./exercise_3.sh
The script is running on knoldus-Vostro-3578
knoldus@knoldus-Vostro-3578:~$ cat exercise_3.sh
#Exercise_3- Store the output of the command “hostname” in a variable. Display “
This script is running on _.” where “_” is the output of the “hostname” command.

HostName=$(hostname)
echo "The script is running on $HostName"

knoldus@knoldus-Vostro-3578:~$
```

**Exercise\_4** - Write a shell script that displays "man", "bear", "pig", "dog", "cat", and "sheep" on the screen with each appearing on a separate line. Try to do this in as few lines as possible.

**Solution: Code:**

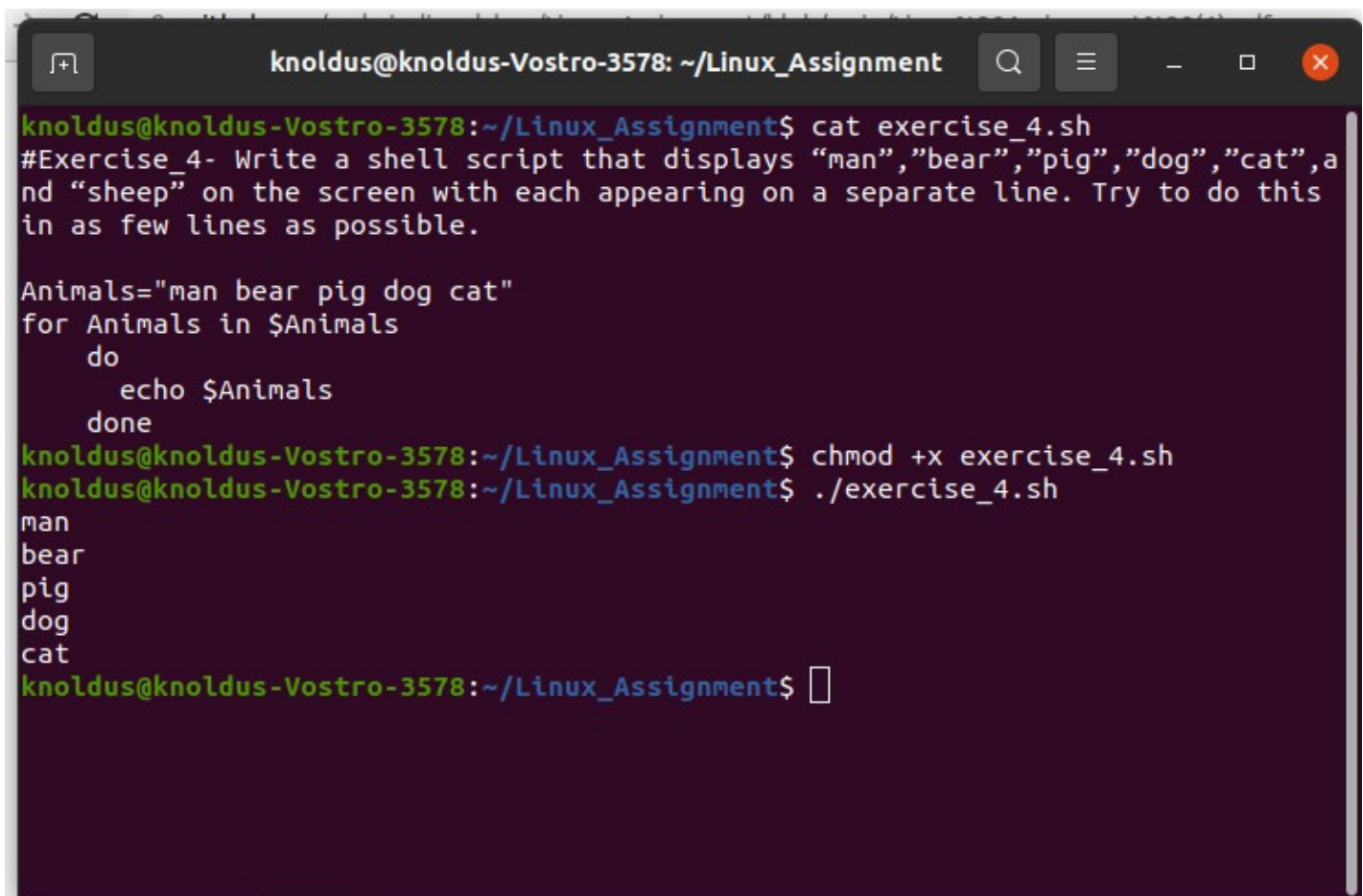
```
Animals="man bear pig dog cat"
```

```
for Animals in $Animals
```

```
do
```

```
    echo $Animals
```

```
done
```

A terminal window titled "knoldus@knoldus-Vostro-3578: ~/Linux\_Assignment" with standard window controls. The terminal shows the following commands and output:

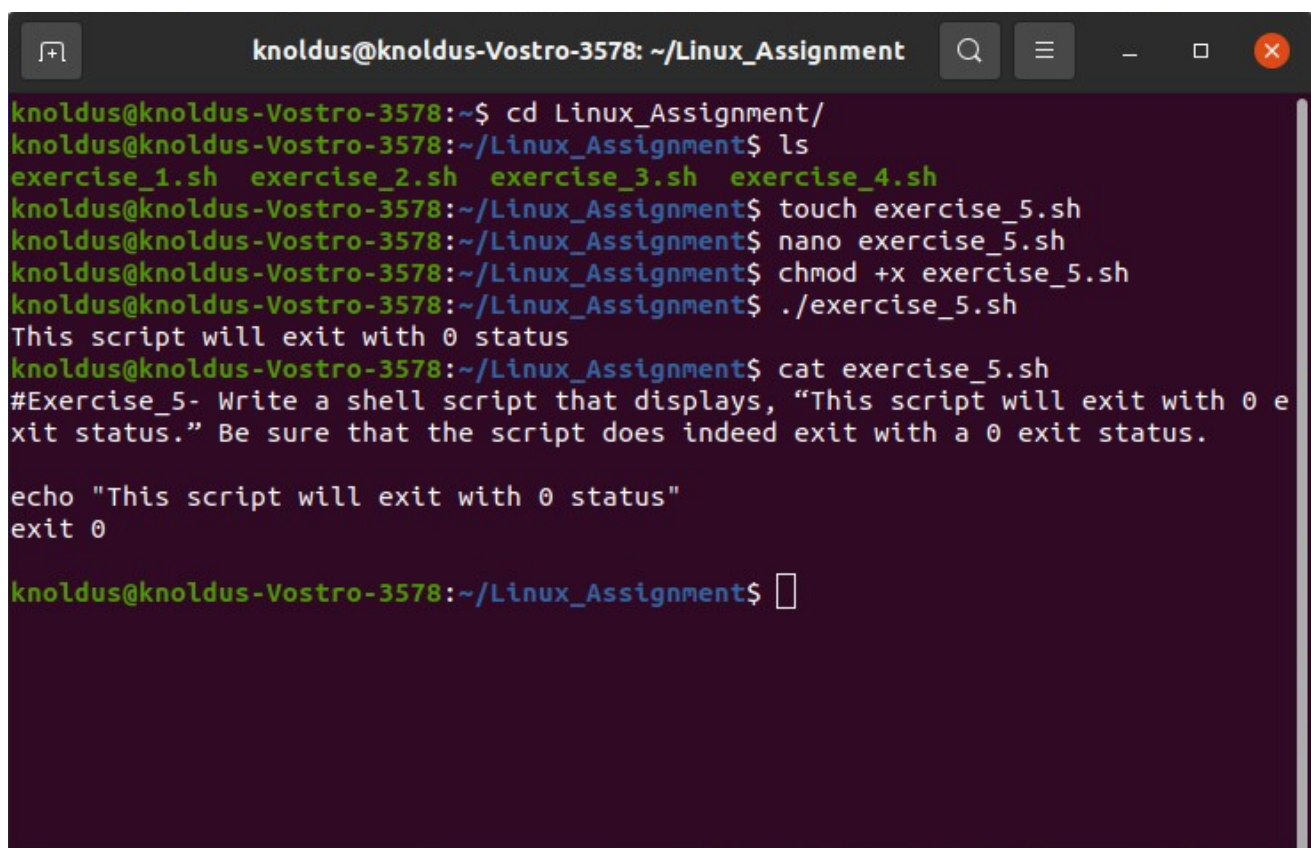
```
knoldus@knoldus-Vostro-3578:~/Linux_Assignment$ cat exercise_4.sh
#Exercise_4- Write a shell script that displays "man","bear","pig","dog","cat",a
nd "sheep" on the screen with each appearing on a separate line. Try to do this
in as few lines as possible.

Animals="man bear pig dog cat"
for Animals in $Animals
do
    echo $Animals
done
knoldus@knoldus-Vostro-3578:~/Linux_Assignment$ chmod +x exercise_4.sh
knoldus@knoldus-Vostro-3578:~/Linux_Assignment$ ./exercise_4.sh
man
bear
pig
dog
cat
knoldus@knoldus-Vostro-3578:~/Linux_Assignment$
```

**Exercise\_5** - Write a shell script that displays, "This script will exit with 0 exit status." Be sure that the script does indeed exit with a 0 exit status.

**Solution: Code:**

```
echo "This script will exit with 0 status"
exit 0
```

A terminal window titled 'knoldus@knoldus-Vostro-3578: ~/Linux\_Assignment' with standard window controls. The terminal shows a series of commands: 'cd Linux\_Assignment/', 'ls' (listing exercise\_1.sh through exercise\_4.sh), 'touch exercise\_5.sh', 'nano exercise\_5.sh', 'chmod +x exercise\_5.sh', and './exercise\_5.sh'. The output of the script is 'This script will exit with 0 status'. Then, 'cat exercise\_5.sh' is run, showing the script's content: '#Exercise\_5- Write a shell script that displays, "This script will exit with 0 exit status." Be sure that the script does indeed exit with a 0 exit status.' followed by 'echo "This script will exit with 0 status"' and 'exit 0'. The prompt returns to 'knoldus@knoldus-Vostro-3578:~/Linux\_Assignment\$' with a cursor.

```
knoldus@knoldus-Vostro-3578:~/Linux_Assignment$ cd Linux_Assignment/
knoldus@knoldus-Vostro-3578:~/Linux_Assignment$ ls
exercise_1.sh exercise_2.sh exercise_3.sh exercise_4.sh
knoldus@knoldus-Vostro-3578:~/Linux_Assignment$ touch exercise_5.sh
knoldus@knoldus-Vostro-3578:~/Linux_Assignment$ nano exercise_5.sh
knoldus@knoldus-Vostro-3578:~/Linux_Assignment$ chmod +x exercise_5.sh
knoldus@knoldus-Vostro-3578:~/Linux_Assignment$ ./exercise_5.sh
This script will exit with 0 status
knoldus@knoldus-Vostro-3578:~/Linux_Assignment$ cat exercise_5.sh
#Exercise_5- Write a shell script that displays, "This script will exit with 0 e
xit status." Be sure that the script does indeed exit with a 0 exit status.

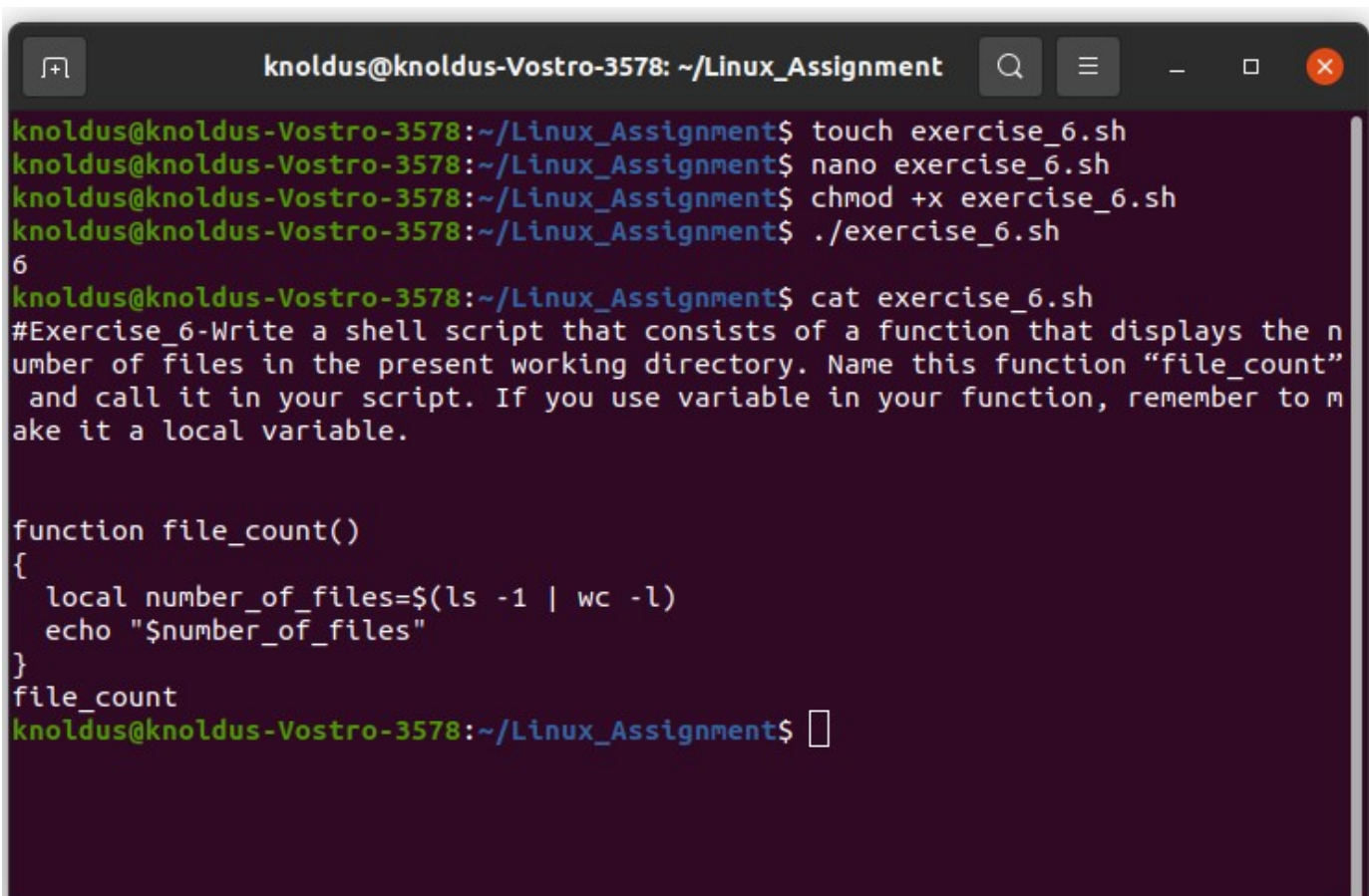
echo "This script will exit with 0 status"
exit 0
knoldus@knoldus-Vostro-3578:~/Linux_Assignment$
```



**Exercise\_6** - Write a shell script that consists of a function that displays the number of files in the present working directory. Name this function “file\_count” and call it in your script. If you use variable in your function, remember to make it a local variable.

**Solution: Code:**

```
function file_count()
{
    local number_of_files = $(ls -1 | wc -l)
    echo "$number_of_files"
}
file_count
```

A terminal window titled 'knoldus@knoldus-Vostro-3578: ~/Linux\_Assignment' shows the following commands and output:

```
knoldus@knoldus-Vostro-3578:~/Linux_Assignment$ touch exercise_6.sh
knoldus@knoldus-Vostro-3578:~/Linux_Assignment$ nano exercise_6.sh
knoldus@knoldus-Vostro-3578:~/Linux_Assignment$ chmod +x exercise_6.sh
knoldus@knoldus-Vostro-3578:~/Linux_Assignment$ ./exercise_6.sh
6
knoldus@knoldus-Vostro-3578:~/Linux_Assignment$ cat exercise_6.sh
#Exercise_6-Write a shell script that consists of a function that displays the n
umber of files in the present working directory. Name this function “file_count”
 and call it in your script. If you use variable in your function, remember to m
ake it a local variable.

function file_count()
{
    local number_of_files=$(ls -1 | wc -l)
    echo "$number_of_files"
}
file_count
knoldus@knoldus-Vostro-3578:~/Linux_Assignment$
```