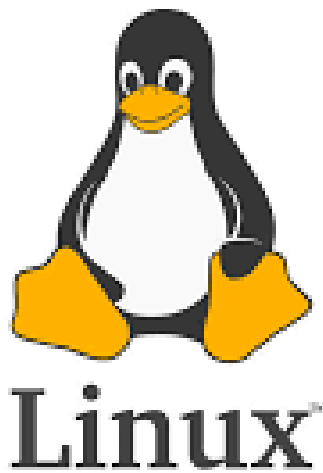


# **LINUX ASSIGNMENT**



**Submitted By :**

**Gulshan Singh**

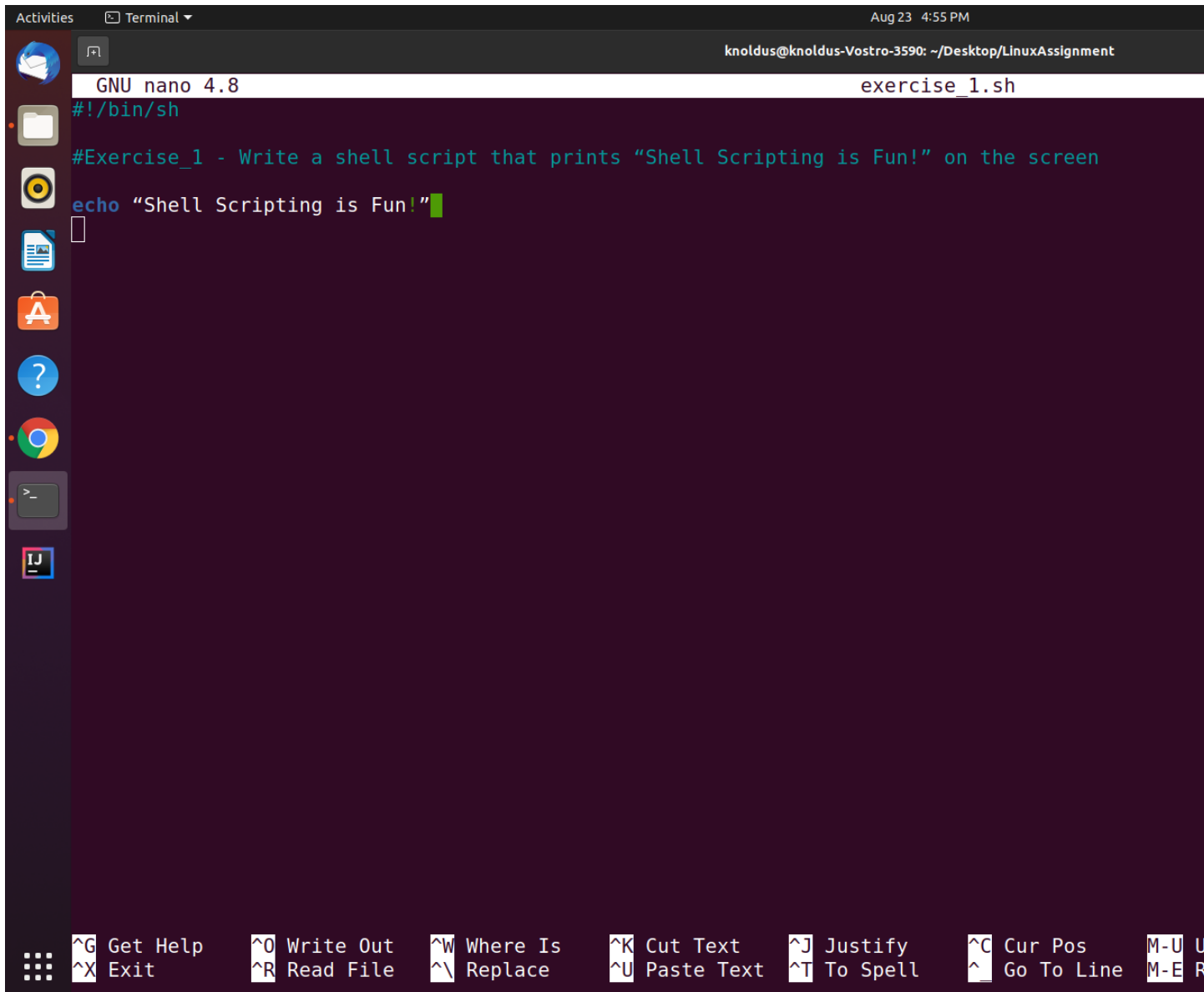
**Employee ID - 1614**

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

**Solution : Code**

```
#!/bin/bash
```

```
echo "Shell Scripting is Fun!"
```



The screenshot shows a Linux desktop environment with a terminal window open. The terminal title bar indicates the user is 'knoldus' on a machine named 'knoldus-Vostro-3590', and the current directory is '~/Desktop/LinuxAssignment'. The terminal is running the 'nano 4.8' text editor, editing a file named 'exercise\_1.sh'. The editor's status bar at the top shows 'GNU nano 4.8' on the left and 'exercise\_1.sh' on the right. The file content is as follows:

```
#!/bin/sh
#Exercise_1 - Write a shell script that prints "Shell Scripting is Fun!" on the screen
echo "Shell Scripting is Fun!"
```

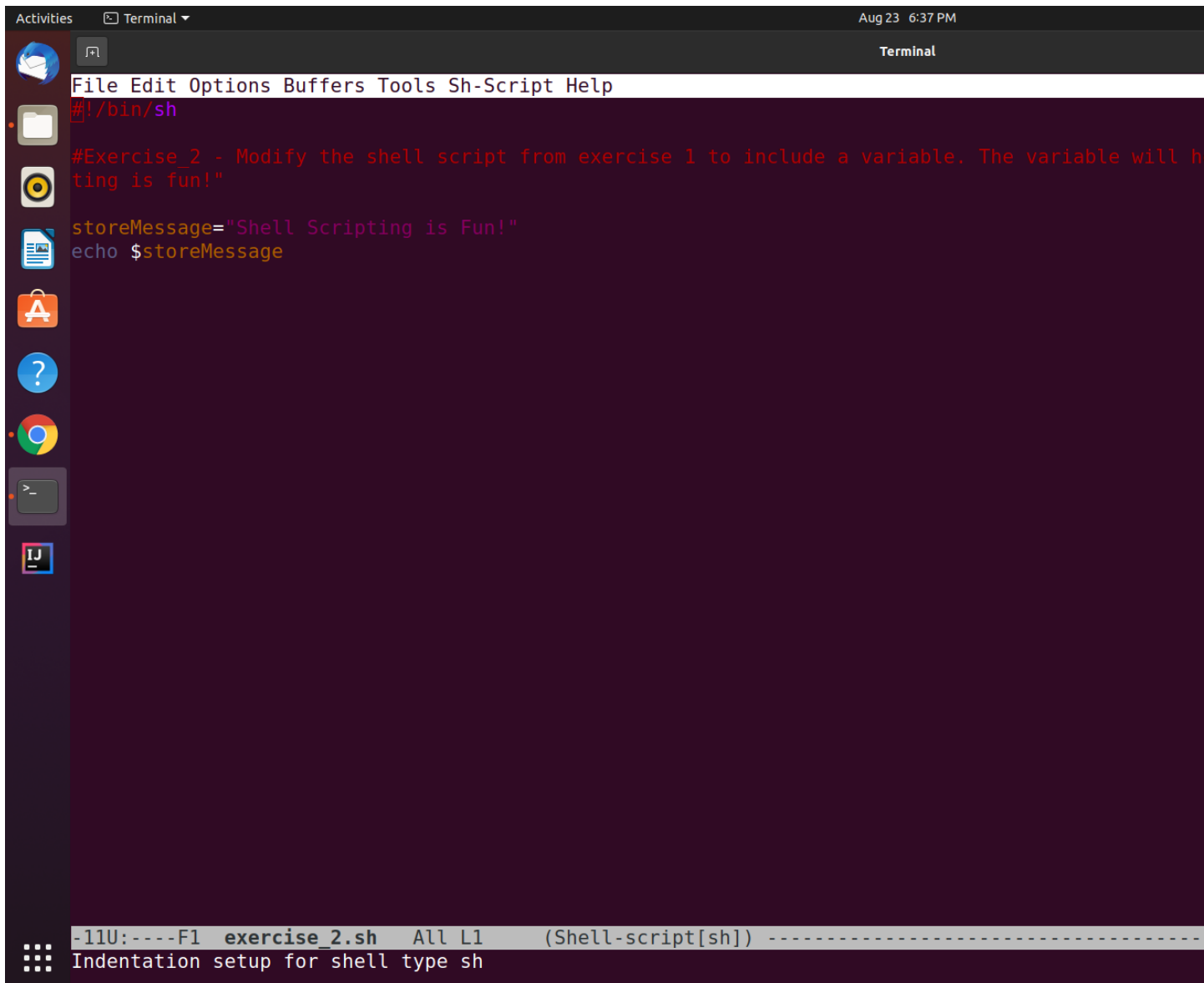
The cursor is positioned at the end of the third line. The bottom of the terminal window displays a series of keyboard shortcuts for nano, such as ^G Get Help, ^X Exit, ^O Write Out, ^R Read File, ^W Where Is, ^\_ Replace, ^K Cut Text, ^U Paste Text, ^J Justify, ^T To Spell, ^C Cur Pos, ^\_ Go To Line, ^M-U, and ^M-E.

```
knoldus@knoldus-Vostro-3590:~$ cd Desktop
knoldus@knoldus-Vostro-3590:~/Desktop$ cd LinuxAssignment
knoldus@knoldus-Vostro-3590:~/Desktop/LinuxAssignment$ nano exercise_1.sh
knoldus@knoldus-Vostro-3590:~/Desktop/LinuxAssignment$ chmod +x exercise_1.sh
knoldus@knoldus-Vostro-3590:~/Desktop/LinuxAssignment$ ./exercise_1.sh
"Shell Scripting is Fun!"
knoldus@knoldus-Vostro-3590:~/Desktop/LinuxAssignment$
```

**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**

```
#!/bin/bash
storeMessage="Shell Scripting is Fun!"
echo $storeMessage
```

A screenshot of a Linux terminal window. The title bar shows 'Activities', 'Terminal', and the date/time 'Aug 23 6:37 PM'. The terminal has a menu bar with 'File Edit Options Buffers Tools Sh-Script Help'. The prompt is '#!/bin/sh'. The script content is displayed in red text: '#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!"', 'storeMessage="Shell Scripting is Fun!"', and 'echo \$storeMessage'. The status bar at the bottom shows '-11U:----F1 exercise\_2.sh All L1 (Shell-script[sh])' and 'Indentation setup for shell type sh'.

```
Activities Terminal Aug 23 6:37 PM
File Edit Options Buffers Tools Sh-Script Help
#!/bin/sh
#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!"
storeMessage="Shell Scripting is Fun!"
echo $storeMessage
-11U:----F1 exercise_2.sh All L1 (Shell-script[sh])
Indentation setup for shell type sh
```



```
knoldus@knoldus-Vostro-3590:~/Desktop/LinuxAssignment$ nano exercise_2.sh
```

```
knoldus@knoldus-Vostro-3590:~/Desktop/LinuxAssignment$ chmod +x exercise_2.sh
```

```
knoldus@knoldus-Vostro-3590:~/Desktop/LinuxAssignment$ ./exercise_2.sh
```

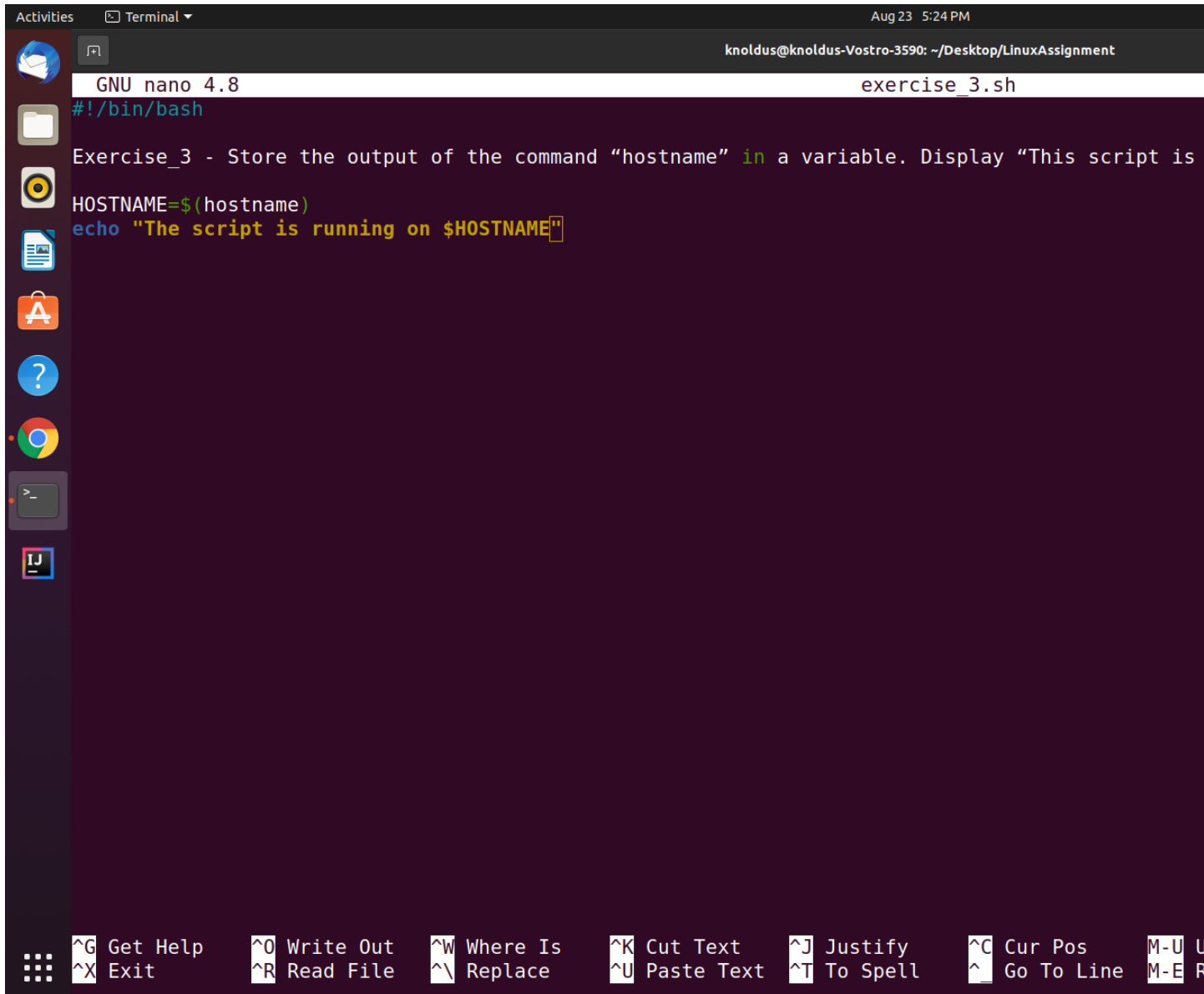
```
Shell Scripting is Fun!
```

```
knoldus@knoldus-Vostro-3590:~/Desktop/LinuxAssignment$
```

**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

```
#!/bin/bash
HOSTNAME=$(hostname)
echo "This script is running on $HOSTNAME"
```



Activities Terminal Aug 23 5:24 PM knoldus@knoldus-Vostro-3590: ~/Desktop/LinuxAssignment

GNU nano 4.8 exercise\_3.sh

```
#!/bin/bash

Exercise_3 - Store the output of the command "hostname" in a variable. Display "This script is

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

Get Help Write Out Where Is Cut Text Justify Cur Pos M-U U  
Exit Read File Replace Paste Text To Spell Go To Line M-E R



```
knoldus@knoldus-Vostro-3590:~/Desktop/LinuxAssignment$ nano exercise_3.sh
```

```
knoldus@knoldus-Vostro-3590:~/Desktop/LinuxAssignment$
```

```
knoldus@knoldus-Vostro-3590:~/Desktop/LinuxAssignment$ chmod +x exercise_3.sh
```

```
knoldus@knoldus-Vostro-3590:~/Desktop/LinuxAssignment$ ./exercise_3.sh
```

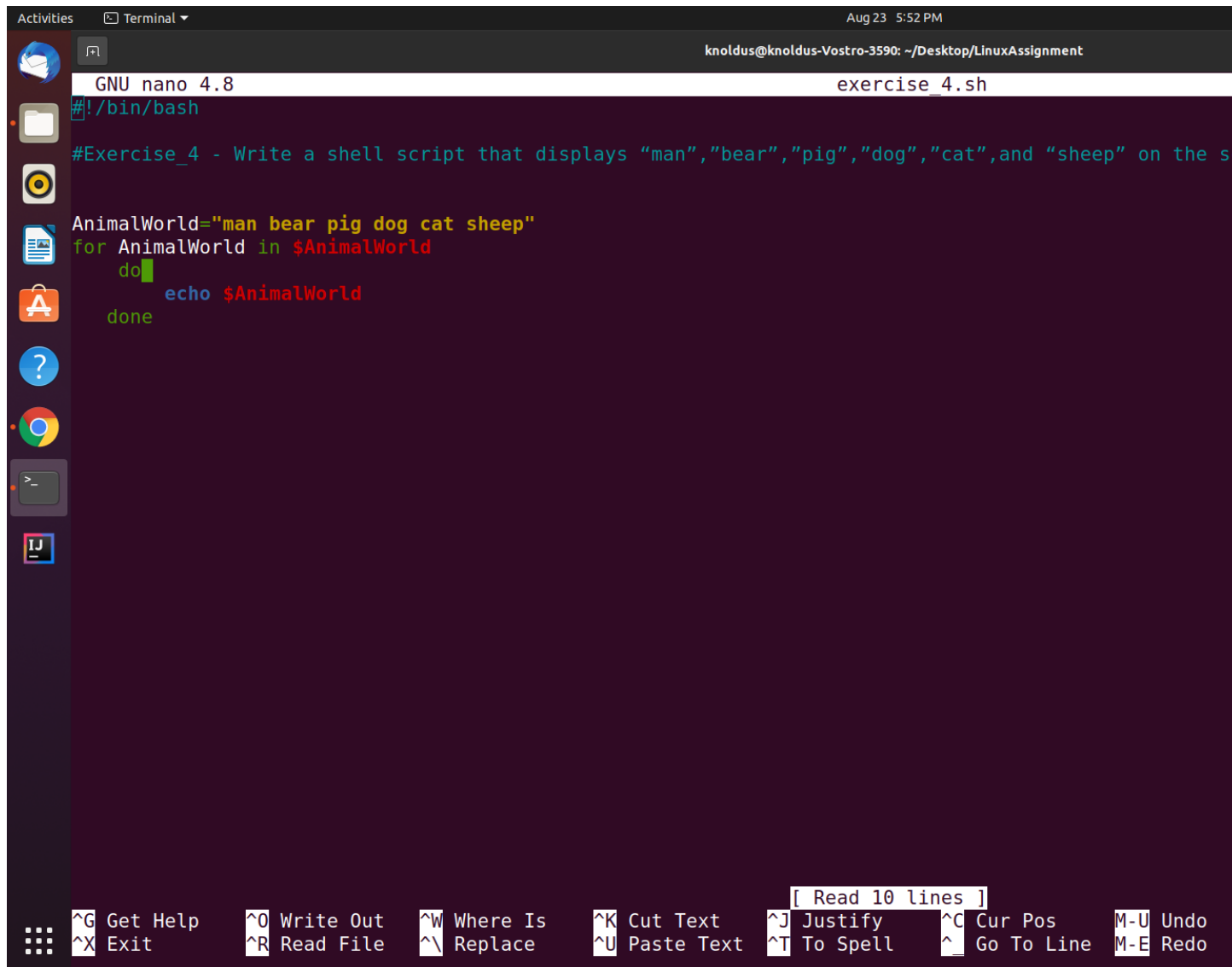
The script is running on knoldus-Vostro-3590

```
knoldus@knoldus-Vostro-3590:~/Desktop/LinuxAssignment$
```

**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**

```
#!/bin/bash
AnimalWorld="man bear pig dog cat sheep"
for AnimalWorld in $AnimalWorld
do
    echo $AnimalWorld
done
```



The screenshot shows a terminal window with the nano 4.8 text editor open. The editor is editing a file named `exercise_4.sh`. The script content is as follows:

```
#!/bin/bash
#Exercise_4 - Write a shell script that displays “man”, “bear”, “pig”, “dog”, “cat”, and “sheep” on the s
AnimalWorld="man bear pig dog cat sheep"
for AnimalWorld in $AnimalWorld
do
    echo $AnimalWorld
done
```

The terminal window includes a sidebar on the left with icons for various applications. At the bottom, there is a status bar with keyboard shortcuts for navigation and editing, such as `^G Get Help`, `^O Write Out`, `^W Where Is`, `^K Cut Text`, `^J Justify`, `^C Cur Pos`, `M-U Undo`, `^X Exit`, `^R Read File`, `^N Replace`, `^U Paste Text`, `^T To Spell`, `^_ Go To Line`, and `M-E Redo`. A small box in the status bar indicates `[ Read 10 lines ]`.

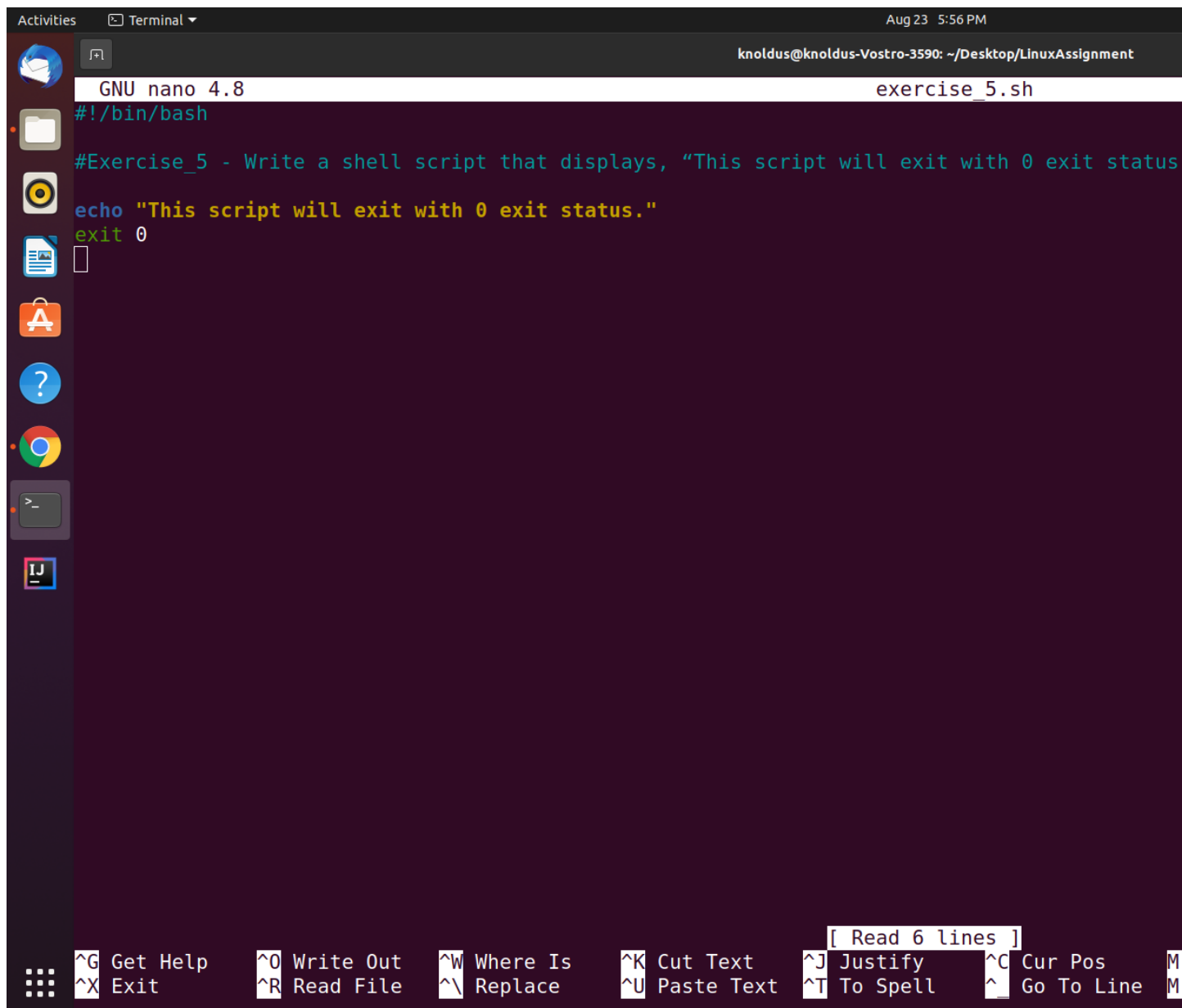


```
knoldus@knoldus-Vostro-3590:~/Desktop/LinuxAssignment$ nano exercise_4.sh
knoldus@knoldus-Vostro-3590:~/Desktop/LinuxAssignment$ chmod +x exercise_4.sh
knoldus@knoldus-Vostro-3590:~/Desktop/LinuxAssignment$ ./exercise_4.sh
man
bear
pig
dog
cat
sheep
knoldus@knoldus-Vostro-3590:~/Desktop/LinuxAssignment$
```

**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

```
#!/bin/bash
echo "This script will exit with 0 exit status."
exit 0
```



The screenshot shows a terminal window titled "Terminal" with the date and time "Aug 23 5:56 PM". The user is logged in as "knoldus@knoldus-Vostro-3590" in the directory "~/Desktop/LinuxAssignment". The terminal is running the GNU nano 4.8 text editor, editing a file named "exercise\_5.sh". The script content is as follows:

```
#!/bin/bash
#Exercise_5 - Write a shell script that displays, "This script will exit with 0 exit status
echo "This script will exit with 0 exit status."
exit 0
```

The bottom of the terminal window displays a row of keyboard shortcuts for nano:

<b>^G</b> Get Help	<b>^O</b> Write Out	<b>^W</b> Where Is	<b>^K</b> Cut Text	<b>^J</b> Justify	<b>^C</b> Cur Pos
<b>^X</b> Exit	<b>^R</b> Read File	<b>^_\b</b> Replace	<b>^U</b> Paste Text	<b>^T</b> To Spell	<b>^_</b> Go To Line

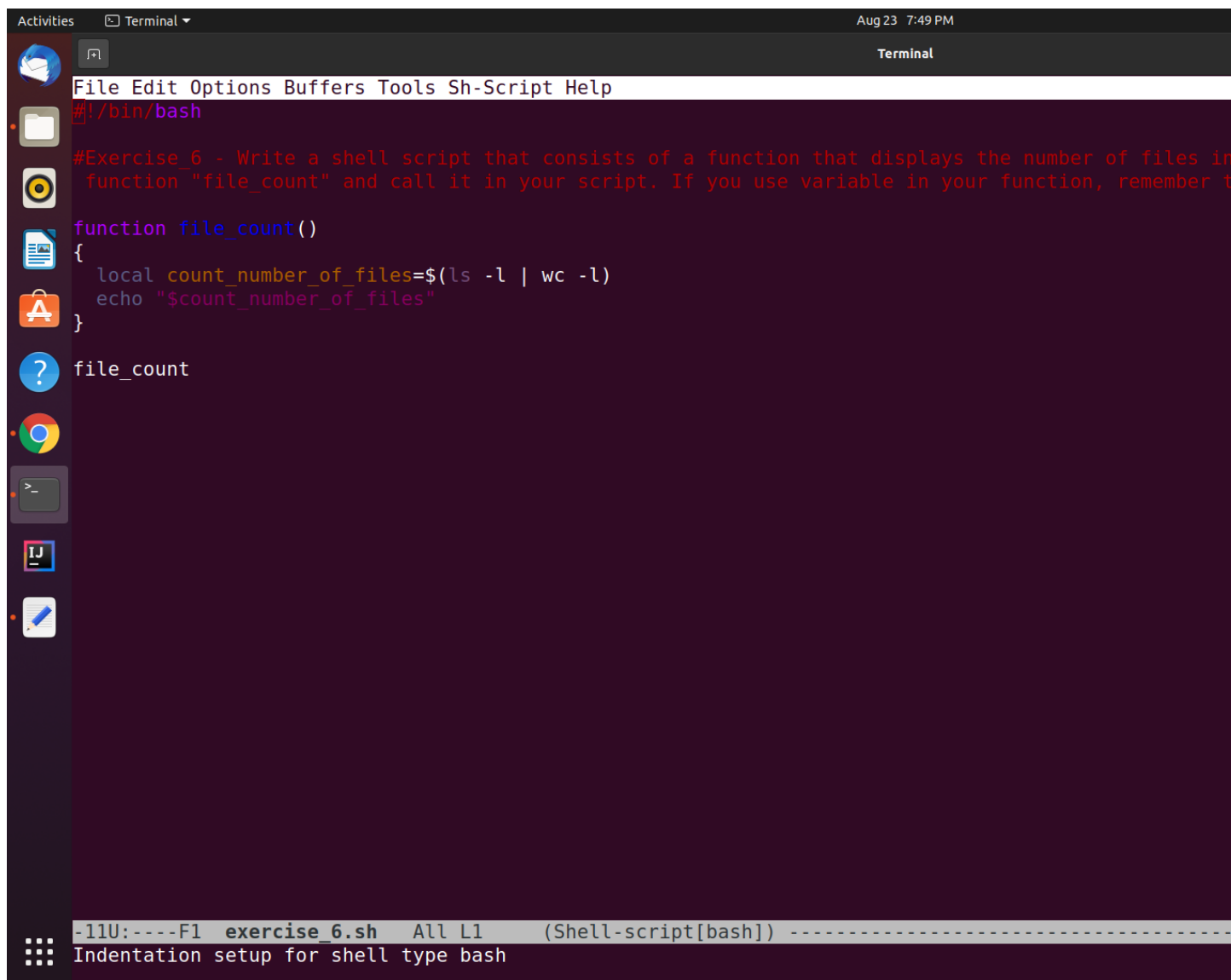
A status bar at the bottom right indicates "[ Read 6 lines ]".

```
knoldus@knoldus-Vostro-3590:~/Desktop/LinuxAssignment$ nano exercise_5.sh
knoldus@knoldus-Vostro-3590:~/Desktop/LinuxAssignment$ chmod +x exercise_5.sh
knoldus@knoldus-Vostro-3590:~/Desktop/LinuxAssignment$ ./exercise_5.sh
This script will exit with 0 exit status.
knoldus@knoldus-Vostro-3590:~/Desktop/LinuxAssignment$
```

**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**

```
#!/bin/bash
function file_count()
{
    local COUNT_NUMBER_OF_FILES=$(ls -l | wc -l)
    echo "$COUNT_NUMBER_OF_FILES"
}
file_count
```

A screenshot of a Linux terminal window. The window has a title bar with 'Activities', 'Terminal', and a dropdown arrow. The date and time 'Aug 23 7:49 PM' are in the top right. A menu bar is visible with 'File', 'Edit', 'Options', 'Buffers', 'Tools', 'Sh-Script', and 'Help'. The terminal content shows the script from the previous block, with syntax highlighting: purple for shebang, red for comments, blue for function names, and yellow for variables. The prompt is '-11U:----F1'. At the bottom, a status bar shows 'exercise\_6.sh', 'All L1', '(Shell-script[bash])', and 'Indentation setup for shell type bash'.

```
File Edit Options Buffers Tools Sh-Script Help
#!/bin/bash
#Exercise_6 - Write a shell script that consists of a function that displays the number of files in
function "file_count" and call it in your script. If you use variable in your function, remember t
function file_count()
{
    local count_number_of_files=$(ls -l | wc -l)
    echo "$count_number_of_files"
}
file_count

-11U:----F1 exercise_6.sh All L1 (Shell-script[bash]) -----
Indentation setup for shell type bash
```

```
knoldus@knoldus-Vostro-3590:~/Desktop/LinuxAssignment$ nano exercise_6.sh
knoldus@knoldus-Vostro-3590:~/Desktop/LinuxAssignment$ chmod +x exercise_6.sh
knoldus@knoldus-Vostro-3590:~/Desktop/LinuxAssignment$ ./exercise_6.sh
7
knoldus@knoldus-Vostro-3590:~/Desktop/LinuxAssignment$
```