

# BASH SCRIPTING 1

## ARHIMETIC OPERATIONS

```
karthik@f542ba3b5a78573: ~  
karthik@f542ba3b5a78573:~$ ./greet  
Hello! What's your name?  
karthikeya  
Hello, karthikeya! Welcome to the world of shell scripting!  
karthik@f542ba3b5a78573:~$ nano greet  
karthik@f542ba3b5a78573:~$ nano bashscript  
karthik@f542ba3b5a78573:~$ chmod a+x bashscript  
karthik@f542ba3b5a78573:~$ ./bashscript  
x=8, y=2  
Addition of x & y:  
10  
Subtraction of x & y:  
6  
Multiplication of x & y:  
16  
Division of x by y:  
4  
Exponentiation of x^y:  
64  
Modular Division of x % y:  
0  
Incrementing x by 5, then x =  
13  
Decrementing x by 5, then x =  
8  
Multiplying x by 5, then x =  
40  
Dividing x by 5, then x =  
8  
Remainder of Dividing x by 5, then x =  
3  
karthik@f542ba3b5a78573:~$
```

```
GNU nano 7.2 bashscript  
#!/bin/bash  
  
x=8  
y=2  
  
echo "x=8, y=2"  
  
echo "Addition of x & y:"  
echo $(( x + y ))  
  
echo "Subtraction of x & y:"  
echo $(( x - y ))  
  
echo "Multiplication of x & y:"  
echo $(( x * y ))  
  
echo "Division of x by y:"  
echo $(( x / y ))  
  
echo "Exponentiation of x^y:"  
echo $(( x ** y ))  
  
echo "Modular Division of x % y:"  
echo $(( x % y ))  
  
echo "Incrementing x by 5, then x = "  
(( x += 5 ))  
echo $x
```

```
karthik@f542ba3b5a78573: ~  
g/npm:/snap/bin  
DBUS_SESSION_BUS_ADDRESS=unix:path=/run/user/1000/bus  
HOSTTYPE=x86_64  
PULSE_SERVER=unix:/mnt/wslg/PulseServer  
_=/usr/bin/env  
karthik@f542ba3b5a78573:~$ echo $PATH  
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/usr/lib/wsl/lib:/mnt/c/Program Files/WindowsApps/CanonicalGroupLimited.Ubuntu_2404.1.68.0_x64__79rhkpfndgsc:/mnt/c/Program Files/Python313/Scripts:/mnt/c/Program Files/Python313:/mnt/c/Program Files/Microsoft MPI/Bin:/mnt/c/Program Files/Common Files/Oracle/Java/javapath:/mnt/c/Program Files (x86)/Common Files/Oracle/Java/javapath:/mnt/c/Program Files (x86)/Common Files/Oracle/Java/javapath:/mnt/c/Windows/system32:/mnt/c/Windows:/mnt/c/Windows/System32/Wbem:/mnt/c/Windows/System32/WindowsPowerShell/v1.0:/mnt/c/Windows/System32/OpenSSH:/mnt/c/Program Files/Java/jdk-11.0.15.1/bin:/mnt/c/apache-maven-3.9.6/bin:/mnt/c/Program Files (x86)/Microsoft SQL Server/160/Tools/Binn:/mnt/c/Program Files/Microsoft SQL Server/160/Tools/Binn:/mnt/c/Program Files/Microsoft SQL Server/Client SDK/ODBC/170/Tools/Binn:/mnt/c/Program Files/Microsoft SQL Server/160/DTS/Binn:/mnt/c/Program Files (x86)/Microsoft SQL Server/160/DTS/Binn:/mnt/c/Program Files/nodejs:/mnt/c/Program Files/Git/cmd:/mnt/c/Program Files/Docker/Docker/resources/bin:/mnt/c/Program Files (x86)/Groovy/bin:/mnt/c/Program Files/MySQL/MySQL Shell 8.0/bin:/mnt/c/Users/Administrator/AppData/Local/Microsoft/WindowsApps:/mnt/c/Users/Administrator/AppData/Local/Programs/Microsoft VS Code/bin:/mnt/c/Users/Administrator/AppData/Roaming/npm:/snap/bin  
karthik@f542ba3b5a78573:~$ nano bashscript2  
karthik@f542ba3b5a78573:~$ chmod +x bashscript2  
karthik@f542ba3b5a78573:~$ ./bashscript2  
a=10, b=3  
c is the value of addition: c = a + b  
c = 13  
karthik@f542ba3b5a78573:~$
```

```
GNU nano 7.2 bashscript2 *  
#!/bin/bash  
  
# Basic arithmetic using expr  
  
echo "a=10, b=3"  
  
# Define variables  
a=10  
b=3  
  
# Perform addition using expr and store in c  
echo "c is the value of addition: c = a + b"  
echo "c = `expr $a + $b`"
```

## GREATER NUMBER

```
karthik@f542ba3b5a78573: ~  
karthik@f542ba3b5a78573:~$ join file1 file2  
join: file1:2: is not sorted:  
join: file2:3: is not sorted:  
1 John Doe  
  
2 Jane Doe  
  
3 Mary Sue  
  
join: input is not in sorted order  
karthik@f542ba3b5a78573:~$ nano greaternum  
karthik@f542ba3b5a78573:~$ chmod a+x greaternum  
karthik@f542ba3b5a78573:~$ ./greaternum  
Enter number: 5  
karthik@f542ba3b5a78573:~$ ./greaternum  
Enter number: 146  
Value is greater than 125  
karthik@f542ba3b5a78573:~$ nano stringcom  
karthik@f542ba3b5a78573:~$ chmod a+x stringcom  
karthik@f542ba3b5a78573:~$ ./stringcom  
true condition  
karthik@f542ba3b5a78573:~$
```

Activate Windows  
Go to Settings to activate Windows.

```
karthik@f542ba3b5a78573: ~  
GNU nano 7.2 greaternum  
#!/bin/bash  
read -p "Enter number: " number  
if [ $number -gt 125 ]; then  
    echo "Value is greater than 125"  
fi
```

[ Read 6 lines ]

Activate Windows  
Go to Settings to activate Windows.

^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location M-U Undo M-A Set Mark ^I-1 To Bracket  
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^\_ Go To Line M-E Redo M-6 Copy ^Q Where Was

# FILE COMPARISON

```
karthik@f542ba3b5a78573: ~  
GNU nano 7.2 stringcom  
#!/bin/bash  
# If condition is true  
if [ "myfile" == "myfile" ]; then  
    echo "true condition"  
fi  
  
# If condition is false  
if [ "myfile" == "yourfile" ]; then  
    echo "false condition"  
fi
```

Read 11 lines

Help Write Out Where Is Cut Execute Location Undo M-A Set Mark M-1 To Bracket  
Exit Read File Replace Paste Justify Go To Line M-E Redo M-G Copy M-Q Where Was

Type here to search

ENG 17:37  
US 24-01-2025

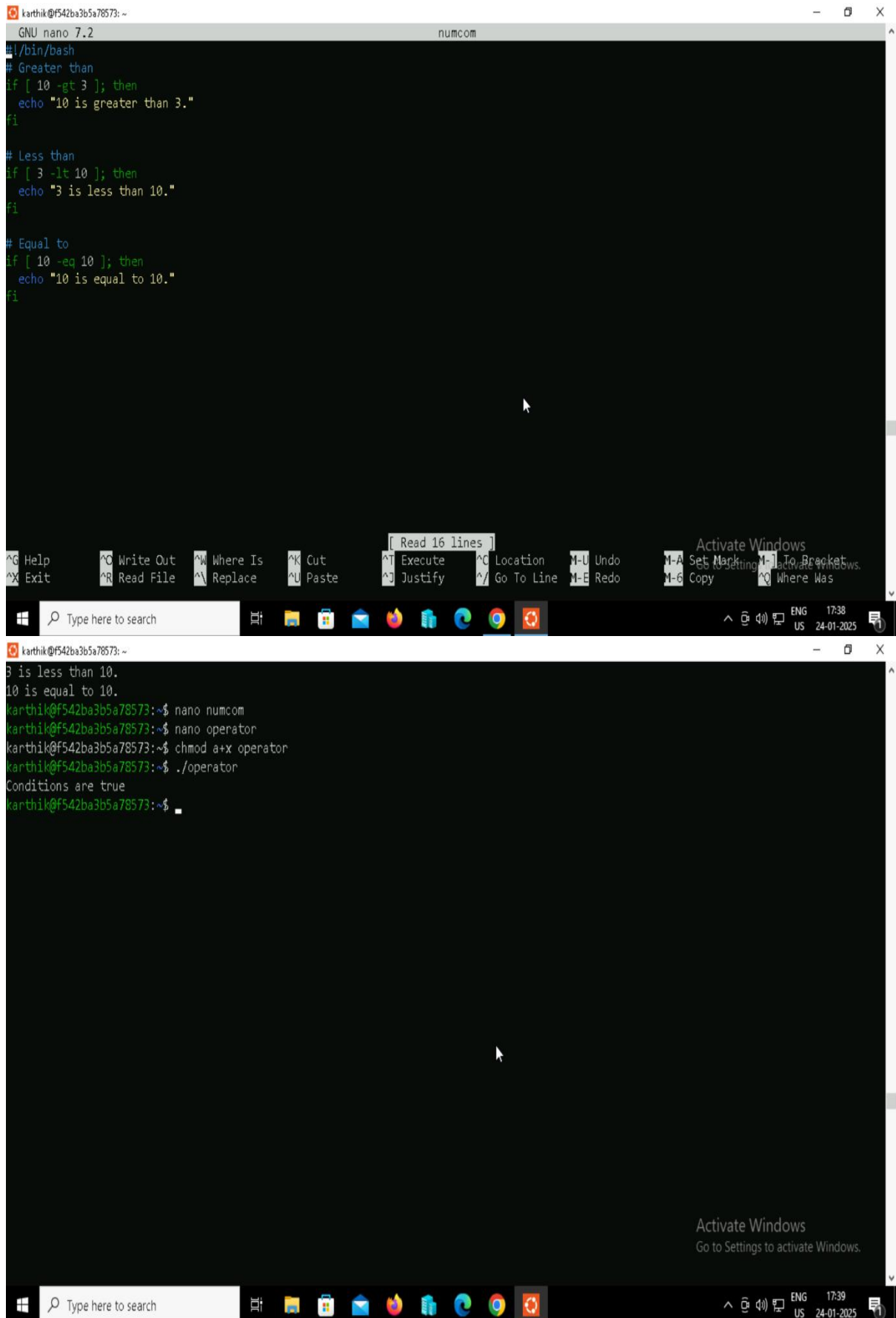
```
karthik@f542ba3b5a78573: ~  
karthik@f542ba3b5a78573:~$ nano stringcom  
karthik@f542ba3b5a78573:~$ nano numcom  
karthik@f542ba3b5a78573:~$ chmod a+x numcom  
karthik@f542ba3b5a78573:~$ ./numcom  
10 is greater than 3.  
3 is less than 10.  
10 is equal to 10.  
karthik@f542ba3b5a78573:~$
```

Activate Windows  
Go to Settings to activate Windows.

Type here to search

ENG 17:38  
US 24-01-2025

# OPERATOR PROGRAM



The image displays two sequential screenshots of a Windows terminal window, illustrating the process of creating and running a shell script.

**Top Screenshot:** The terminal shows the GNU nano 7.2 editor open with a file named 'numcom'. The script content is as follows:

```
#!/bin/bash
# Greater than
if [ 10 -gt 3 ]; then
    echo "10 is greater than 3."
fi

# Less than
if [ 3 -lt 10 ]; then
    echo "3 is less than 10."
fi

# Equal to
if [ 10 -eq 10 ]; then
    echo "10 is equal to 10."
fi
```

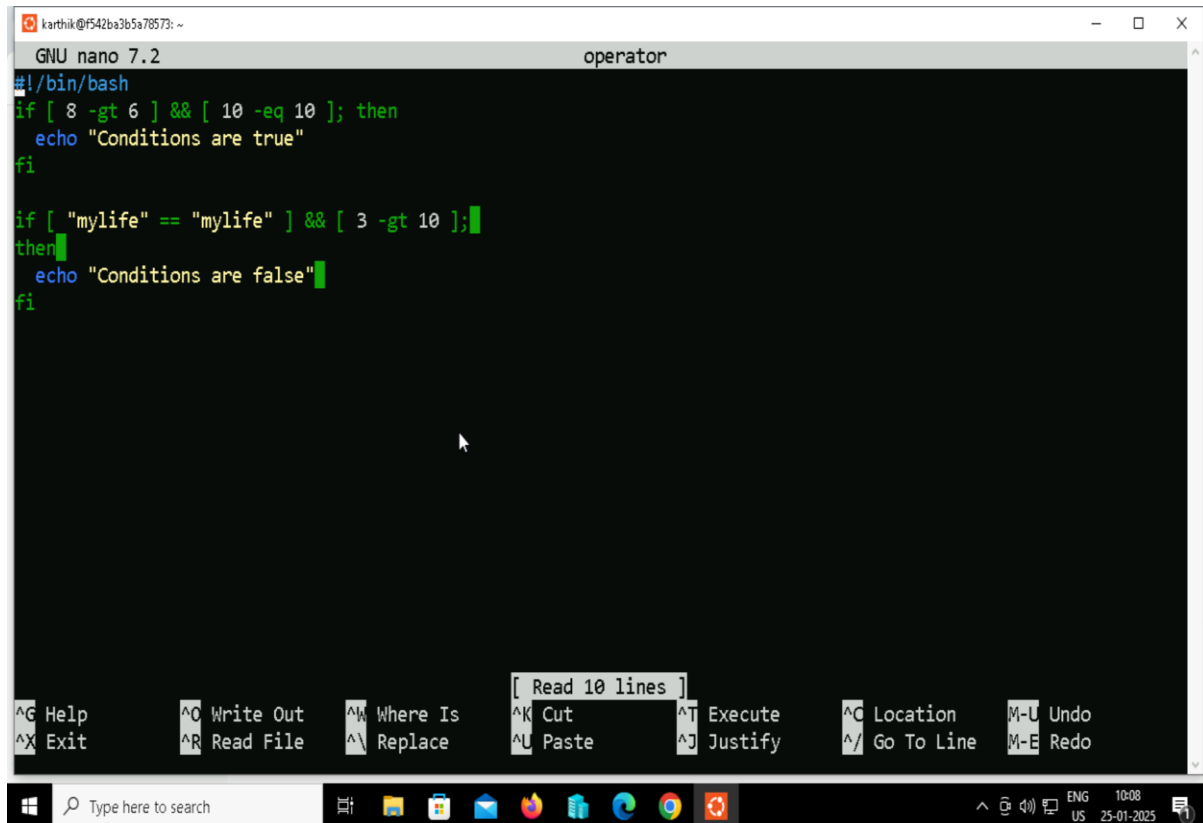
The terminal window includes a menu bar with options like Help, Write Out, Where Is, Cut, Execute, Location, Undo, Set Mark, To Bracket, Exit, Read File, Replace, Paste, Justify, Go To Line, Redo, Copy, and Where Was. The Windows taskbar at the bottom shows the search bar and various application icons.

**Bottom Screenshot:** This screenshot shows the terminal after the script has been executed. The output of the script is visible:

```
3 is less than 10.
10 is equal to 10.
karthik@f542ba3b5a78573:~$ nano numcom
karthik@f542ba3b5a78573:~$ nano operator
karthik@f542ba3b5a78573:~$ chmod a+x operator
karthik@f542ba3b5a78573:~$ ./operator
Conditions are true
karthik@f542ba3b5a78573:~$
```

The terminal window now shows the command prompt, indicating the script execution is complete. The Windows taskbar and system tray are also visible at the bottom.

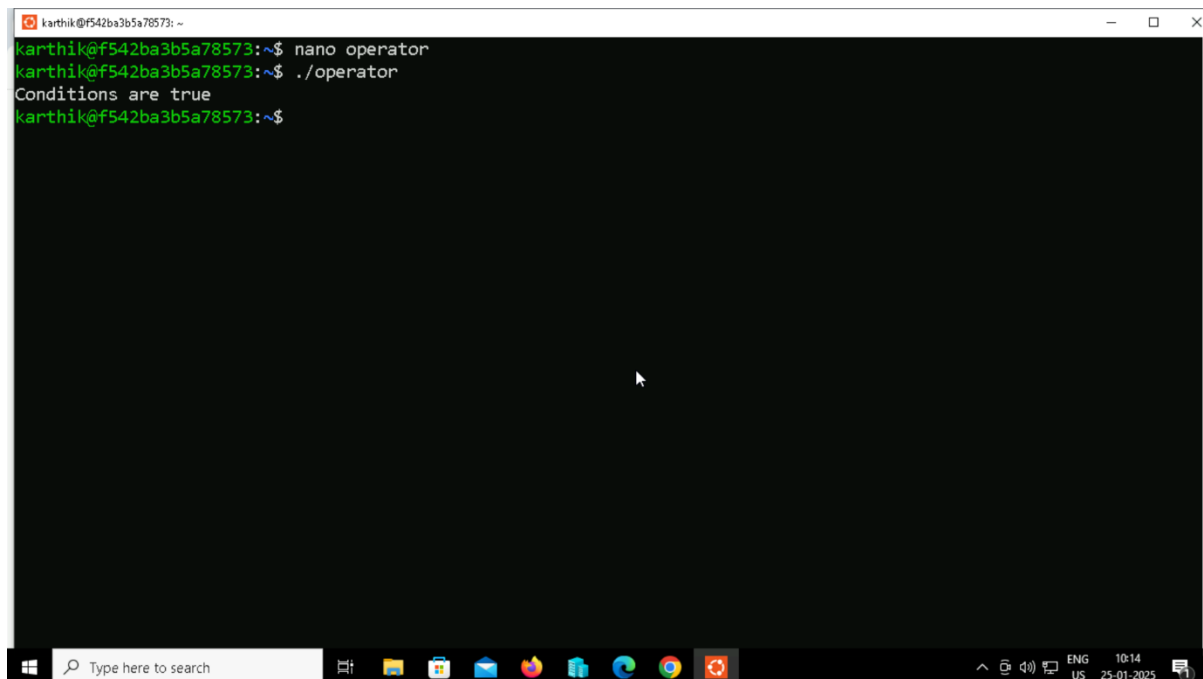
## AND OPERATOR PROGRAM



```
karthik@f542ba3b5a78573: ~  
GNU nano 7.2 operator  
#!/bin/bash  
if [ 8 -gt 6 ] && [ 10 -eq 10 ]; then  
    echo "Conditions are true"  
fi  
  
if [ "mylife" == "mylife" ] && [ 3 -gt 10 ];  
then  
    echo "Conditions are false"  
fi
```

Help Write Out Where Is Cut Execute Location Undo  
Exit Read File Replace Paste Justify Go To Line M-E Redo

Type here to search



```
karthik@f542ba3b5a78573: ~  
karthik@f542ba3b5a78573:~$ nano operator  
karthik@f542ba3b5a78573:~$ ./operator  
Conditions are true  
karthik@f542ba3b5a78573:~$
```

Type here to search

# OROPERATOR

```
karthik@f542ba3b5a78573: ~  
GNU nano 7.2 oroperator  
#!/bin/bash  
if [ 8 -gt 7 ] || [ 10 -eq 3 ]; then  
    echo "Condition is true."  
fi  
if [ "mylife" == "mylife" ] || [ 3 -gt 10 ];  
then  
    echo "Conditions are false"  
fi
```

Help Write Out Where Is Cut Execute Location M-U Undo  
Exit Read File Replace Paste Justify Go To Line M-E Redo

Type here to search

```
karthik@f542ba3b5a78573: ~  
karthik@f542ba3b5a78573:~$ nano operator  
karthik@f542ba3b5a78573:~$ ./operator  
Conditions are true  
karthik@f542ba3b5a78573:~$ nano oroperator  
karthik@f542ba3b5a78573:~$ ./oroperator  
Condition is true.  
Conditions are false  
karthik@f542ba3b5a78573:~$
```

Type here to search

# NESTEDIF PROGRAM

```
karthik@f542ba3b5a78573: ~  
karthik@f542ba3b5a78573:~$ nano nestedif  
karthik@f542ba3b5a78573:~$ chmod a+x nestedif  
karthik@f542ba3b5a78573:~$ ./nestedif 64  
Number is greater than 50.  
and it is an even number.  
karthik@f542ba3b5a78573:~$
```

```
GNU nano 7.2 nestedif  
#!/bin/bash  
if [ $1 -gt 50 ]; then  
    echo "Number is greater than 50."  
  
    if (( $1 % 2 == 0 )); then  
        echo "and it is an even number."  
    fi  
fi
```

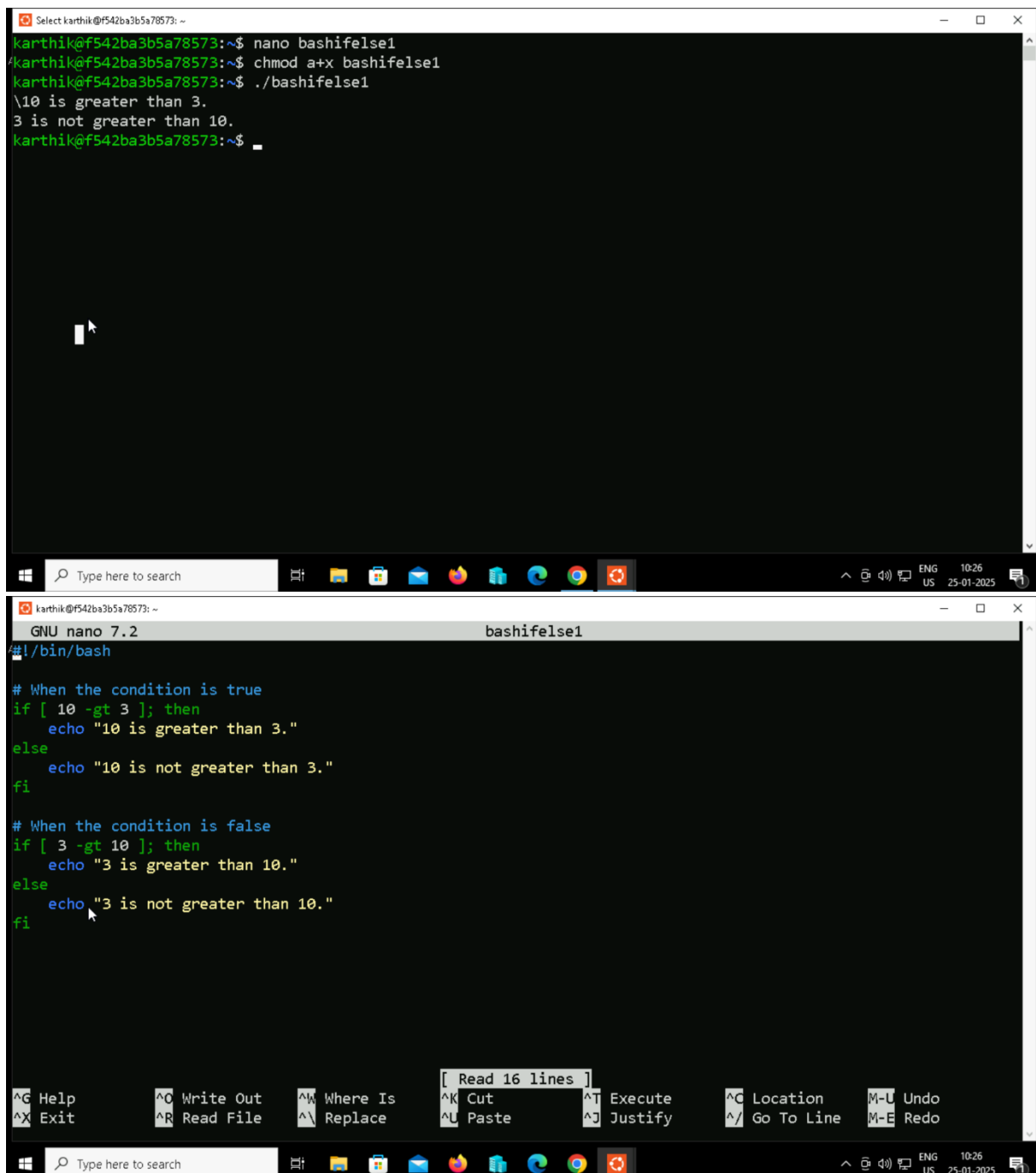
[ Read 9 lines ]

^G Help    ^O Write Out    ^W Where Is    ^K Cut    ^T Execute    ^C Location    ^M Undo  
^X Exit    ^R Read File    ^\ Replace    ^U Paste    ^J Justify    ^\_ Go To Line    ^E Redo



## BASH SCRIPTING 2

### EXAMPLE 1 IF ELSE:



The image consists of two screenshots of a terminal window. The top screenshot shows the execution of a script named 'bashifelse1'. The user runs 'nano bashifelse1', then 'chmod a+x bashifelse1', and finally './bashifelse1'. The script outputs two lines: '\10 is greater than 3.' and '3 is not greater than 10.'. The bottom screenshot shows the source code of 'bashifelse1' in the nano editor. The script uses 'if' and 'else' statements to check conditions and print messages.

```
karthik@f542ba3b5a78573: ~$ nano bashifelse1
karthik@f542ba3b5a78573: ~$ chmod a+x bashifelse1
karthik@f542ba3b5a78573: ~$ ./bashifelse1
\10 is greater than 3.
3 is not greater than 10.
karthik@f542ba3b5a78573: ~$
```

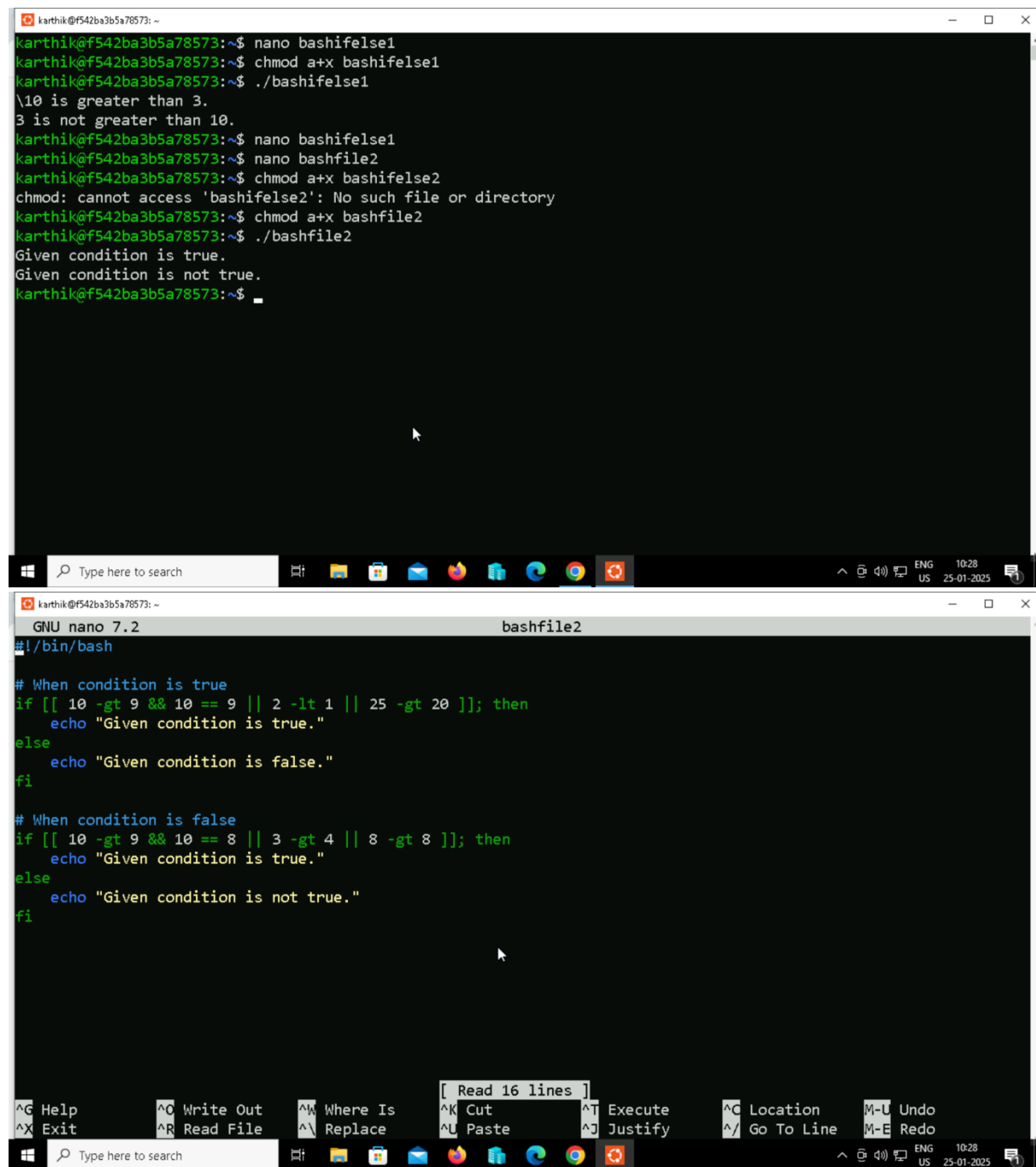
```
GNU nano 7.2 bashifelse1
#!/bin/bash

# When the condition is true
if [ 10 -gt 3 ]; then
    echo "10 is greater than 3."
else
    echo "10 is not greater than 3."
fi

# When the condition is false
if [ 3 -gt 10 ]; then
    echo "3 is greater than 10."
else
    echo "3 is not greater than 10."
fi
```

Help Exit Write Out Read File Where Is Replace [ Read 16 lines ] Cut Paste Execute Justify Location Go To Line M-U Undo M-E Redo

## BASH IFELSE EXAMPLE 2



The image consists of two screenshots of a terminal window. The top screenshot shows the creation and execution of a script named 'bashifelse1'. The user creates the file with 'nano bashifelse1', sets permissions with 'chmod a+x bashifelse1', and runs it with './bashifelse1'. The script outputs two lines: '\10 is greater than 3.' and '3 is not greater than 10.'. The bottom screenshot shows the creation and execution of a script named 'bashfile2'. The user creates the file with 'nano bashfile2', sets permissions with 'chmod a+x bashfile2', and runs it with './bashfile2'. The script outputs two lines: 'Given condition is true.' and 'Given condition is not true.'. The terminal window has a Windows taskbar at the bottom with various icons and a search bar.

```
karthik@f542ba3b5a78573: ~  
karthik@f542ba3b5a78573:~$ nano bashifelse1  
karthik@f542ba3b5a78573:~$ chmod a+x bashifelse1  
karthik@f542ba3b5a78573:~$ ./bashifelse1  
\10 is greater than 3.  
3 is not greater than 10.  
karthik@f542ba3b5a78573:~$ nano bashifelse1  
karthik@f542ba3b5a78573:~$ nano bashfile2  
karthik@f542ba3b5a78573:~$ chmod a+x bashifelse2  
chmod: cannot access 'bashifelse2': No such file or directory  
karthik@f542ba3b5a78573:~$ chmod a+x bashfile2  
karthik@f542ba3b5a78573:~$ ./bashfile2  
Given condition is true.  
Given condition is not true.  
karthik@f542ba3b5a78573:~$
```

```
GNU nano 7.2 bashfile2  
#!/bin/bash  
  
# When condition is true  
if [[ 10 -gt 9 && 10 == 9 || 2 -lt 1 || 25 -gt 20 ]]; then  
    echo "Given condition is true."  
else  
    echo "Given condition is false."  
fi  
  
# When condition is false  
if [[ 10 -gt 9 && 10 == 8 || 3 -gt 4 || 8 -gt 8 ]]; then  
    echo "Given condition is true."  
else  
    echo "Given condition is not true."  
fi
```

## BASH IFELSE EXAMPLE 3

```
karthik@f542ba3b5a78573: ~  
karthik@f542ba3b5a78573:~$ nano bashifelse1  
karthik@f542ba3b5a78573:~$ chmod a+x bashifelse1  
karthik@f542ba3b5a78573:~$ ./bashifelse1  
10 is greater than 3.  
3 is not greater than 10.  
karthik@f542ba3b5a78573:~$ nano bashifelse1  
karthik@f542ba3b5a78573:~$ nano bashfile2  
karthik@f542ba3b5a78573:~$ chmod a+x bashifelse2  
chmod: cannot access 'bashifelse2': No such file or directory  
karthik@f542ba3b5a78573:~$ chmod a+x bashfile2  
karthik@f542ba3b5a78573:~$ ./bashfile2  
Given condition is true.  
Given condition is not true.  
karthik@f542ba3b5a78573:~$ nano bashfile2  
karthik@f542ba3b5a78573:~$ nano bashfile3  
karthik@f542ba3b5a78573:~$ chmod a+x bashfile3  
karthik@f542ba3b5a78573:~$ ./bashfile3  
Enter a value: 5  
The value you typed is not greater than 9.  
karthik@f542ba3b5a78573:~$
```

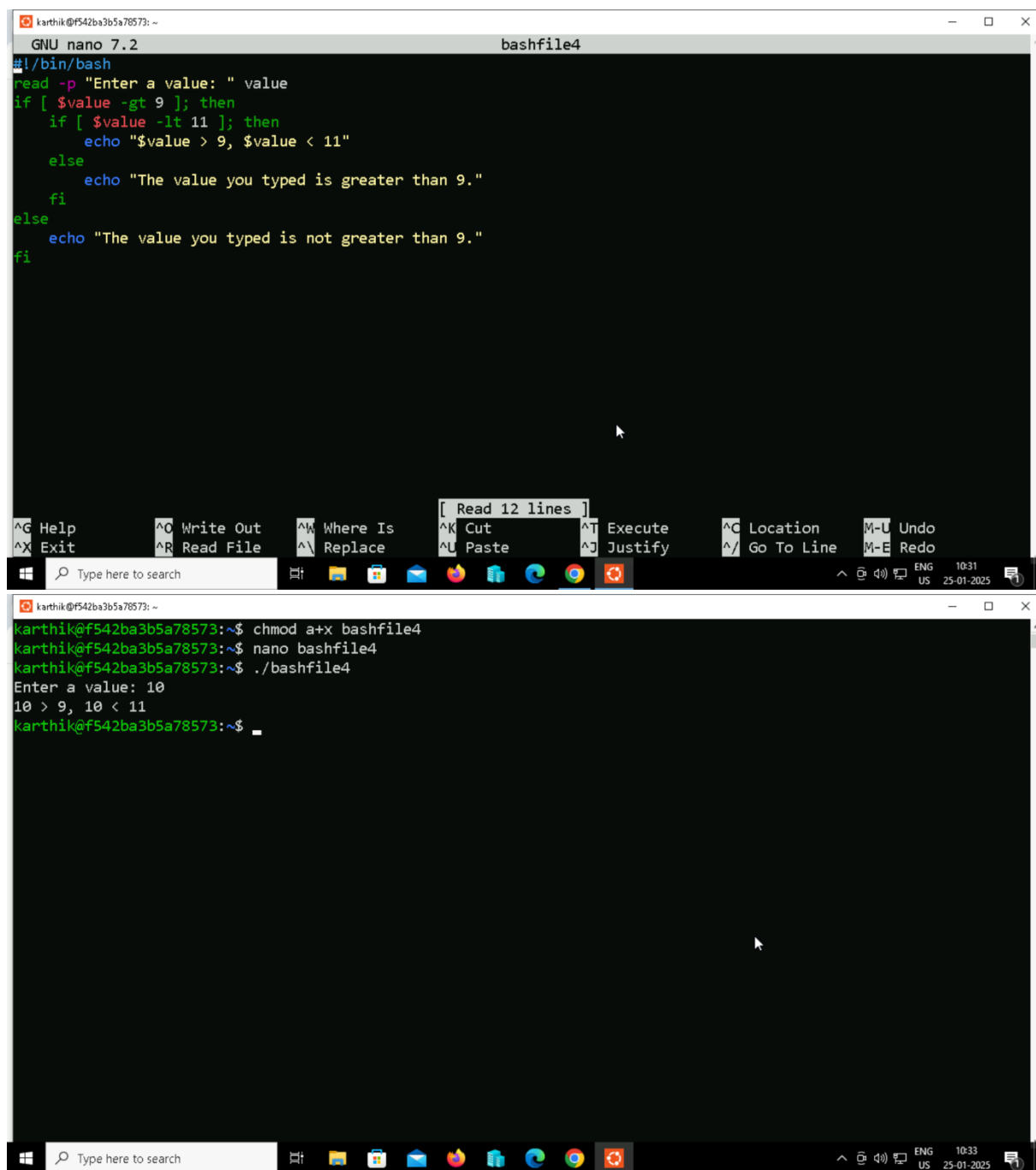
GNU nano 7.2 bashfile3

```
#!/bin/bash  
read -p "Enter a value: " value  
if [ $value -gt 9 ]; then  
    echo "The value you typed is greater than 9."  
else  
    echo "The value you typed is not greater than 9."  
fi
```

Read 8 lines

Help Write Out Where Is Cut Execute Location M-U Undo  
Exit Read File Replace Paste Justify Go To Line M-E Redo

## BASH IFELSE EXAMPLE 4



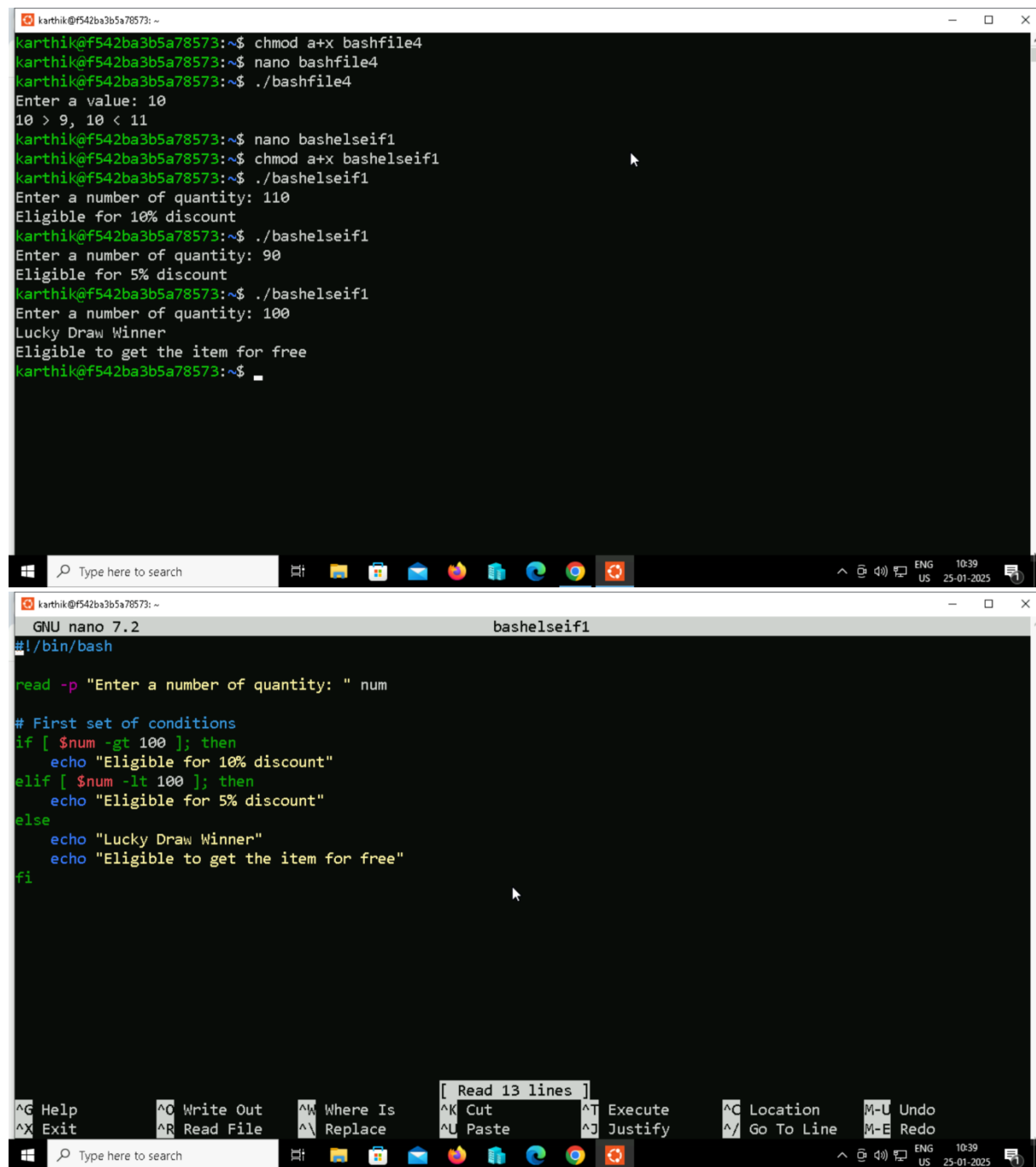
The image consists of two terminal window screenshots. The top screenshot shows the nano text editor editing a file named 'bashfile4'. The script content is as follows:

```
#!/bin/bash
read -p "Enter a value: " value
if [ $value -gt 9 ]; then
    if [ $value -lt 11 ]; then
        echo "$value > 9, $value < 11"
    else
        echo "The value you typed is greater than 9."
    fi
else
    echo "The value you typed is not greater than 9."
fi
```

The bottom screenshot shows the terminal execution of the script. The user runs 'chmod a+x bashfile4', 'nano bashfile4', and './bashfile4'. The prompt 'Enter a value:' is shown, followed by the input '10'. The script then outputs '10 > 9, 10 < 11'.

```
karthik@f542ba3b5a78573: ~$ chmod a+x bashfile4
karthik@f542ba3b5a78573: ~$ nano bashfile4
karthik@f542ba3b5a78573: ~$ ./bashfile4
Enter a value: 10
10 > 9, 10 < 11
karthik@f542ba3b5a78573: ~$
```

## BASH ELSE-IF EXAMPLE 1



The image consists of two screenshots of a terminal window. The top screenshot shows the execution of a Bash script named `bashfile4` and `bashelseif1`. The bottom screenshot shows the source code of `bashelseif1` in the `nano` editor.

**Top Screenshot: Terminal Execution**

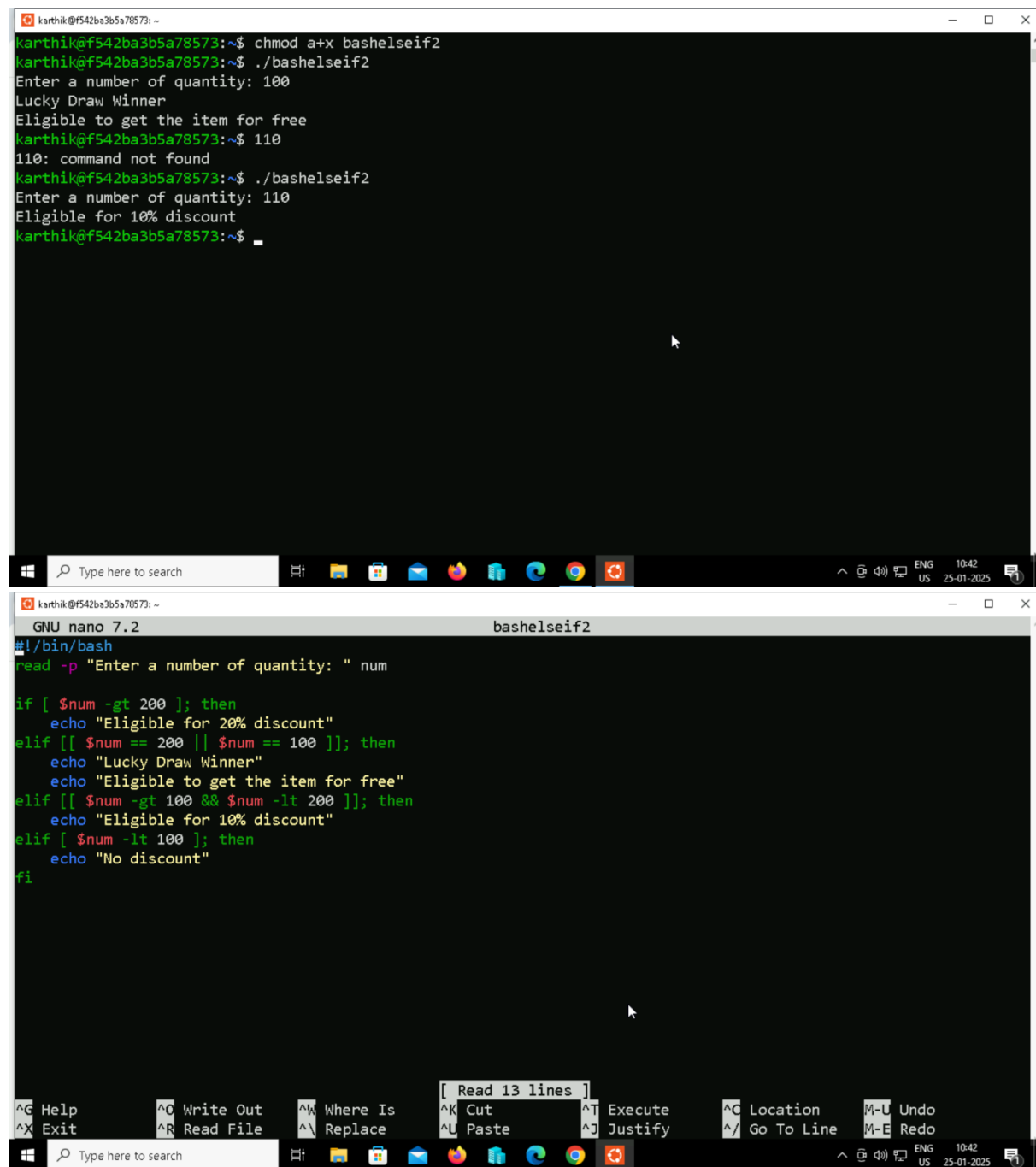
```
karthik@f542ba3b5a78573: ~  
karthik@f542ba3b5a78573:~$ chmod a+x bashfile4  
karthik@f542ba3b5a78573:~$ nano bashfile4  
karthik@f542ba3b5a78573:~$ ./bashfile4  
Enter a value: 10  
10 > 9, 10 < 11  
karthik@f542ba3b5a78573:~$ nano bashelseif1  
karthik@f542ba3b5a78573:~$ chmod a+x bashelseif1  
karthik@f542ba3b5a78573:~$ ./bashelseif1  
Enter a number of quantity: 110  
Eligible for 10% discount  
karthik@f542ba3b5a78573:~$ ./bashelseif1  
Enter a number of quantity: 90  
Eligible for 5% discount  
karthik@f542ba3b5a78573:~$ ./bashelseif1  
Enter a number of quantity: 100  
Lucky Draw Winner  
Eligible to get the item for free  
karthik@f542ba3b5a78573:~$
```

**Bottom Screenshot: nano Editor**

```
GNU nano 7.2 bashelseif1  
#!/bin/bash  
  
read -p "Enter a number of quantity: " num  
  
# First set of conditions  
if [ $num -gt 100 ]; then  
    echo "Eligible for 10% discount"  
elif [ $num -lt 100 ]; then  
    echo "Eligible for 5% discount"  
else  
    echo "Lucky Draw Winner"  
    echo "Eligible to get the item for free"  
fi
```

The bottom screenshot also shows a menu bar with the following options: Help, Write Out, Where Is, Cut, Execute, Location, M-U, Undo, Exit, Read File, Replace, Paste, Justify, Go To Line, M-E, Redo. The status bar at the bottom indicates the time is 10:39 and the date is 25-01-2025.

## BASH ELSE-IF EXAMPLE 2



The image consists of two screenshots of a Windows terminal window. The top screenshot shows the execution of a script named 'bashelseif2'. The user runs 'chmod a+x bashelseif2' and then './bashelseif2'. The script prompts for a quantity. When the user enters '100', the script outputs 'Lucky Draw Winner' and 'Eligible to get the item for free'. When the user enters '110', the script outputs '110: command not found'. When the user enters '110' again, the script outputs 'Eligible for 10% discount'. The bottom screenshot shows the source code of the script 'bashelseif2' in the GNU nano 7.2 editor. The script starts with a shebang '#!/bin/bash' and a prompt 'read -p "Enter a number of quantity: " num'. It then uses a series of 'if' and 'elif' statements to check the value of 'num' and output the appropriate message based on the conditions.

```
karthik@f542ba3b5a78573: ~  
karthik@f542ba3b5a78573:~$ chmod a+x bashelseif2  
karthik@f542ba3b5a78573:~$ ./bashelseif2  
Enter a number of quantity: 100  
Lucky Draw Winner  
Eligible to get the item for free  
karthik@f542ba3b5a78573:~$ 110  
110: command not found  
karthik@f542ba3b5a78573:~$ ./bashelseif2  
Enter a number of quantity: 110  
Eligible for 10% discount  
karthik@f542ba3b5a78573:~$
```

```
GNU nano 7.2 bashelseif2  
#!/bin/bash  
read -p "Enter a number of quantity: " num  
  
if [ $num -gt 200 ]; then  
    echo "Eligible for 20% discount"  
elif [[ $num == 200 || $num == 100 ]]; then  
    echo "Lucky Draw Winner"  
    echo "Eligible to get the item for free"  
elif [[ $num -gt 100 && $num -lt 200 ]]; then  
    echo "Eligible for 10% discount"  
elif [ $num -lt 100 ]; then  
    echo "No discount"  
fi
```