

## Arithmetic Operations

### Q. Arithmetic expressions using let

```
kpkm@c99afc4984265e2: ~/scripting
GNU nano 7.2 ar_ops-let.sh
#Arithmetic expression using let
#!/bin/bash

x=10
y=6
z=0
echo "Addition"
let "z=$(( x + y ))"
echo "z=$z"

echo "Subtraction"
let "z=$(( x - y ))"
echo "z=$z"

echo "Multiplication"
let "z=$(( x * y ))"
echo "z=$z"

echo "Division"
let "z=$(( x / y ))"
echo "z=$z"

echo "Exponentiation"
let "z=$(( x ** y ))"
echo "z=$z"

echo "Modular Division"
let "z=$(( x % y ))"
echo "z=$z"

let "x+=5"
echo "Incrementing x by 5, then x="
echo $x

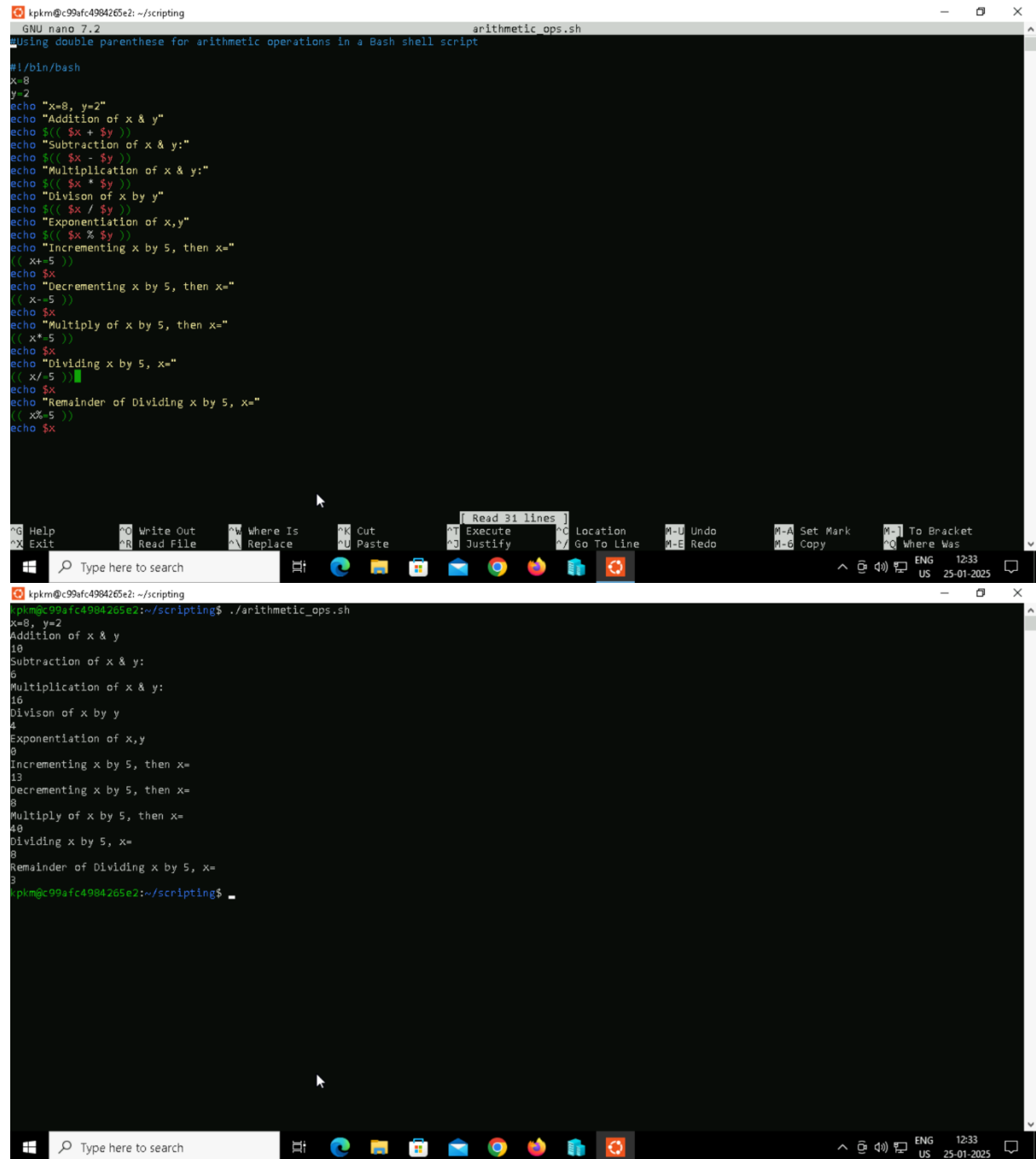
let "x-=5"
echo "Decrementing x by 5, then x="

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

Type here to search

kpkm@c99afc4984265e2: ~/scripting
kpkm@c99afc4984265e2:~/scripting$ nano ar_ops-let.sh
kpkm@c99afc4984265e2:~/scripting$ ./ar_ops-let.sh
Addition
z=16
Subtraction
z=4
Multiplication
z=60
Division
z=1
Exponentiation
z=1000000
Modular Division
z=4
Incrementing x by 5, then x=
15
Decrementing x by 5, then x=
10
Dividing x by 5, x =
2
Remainder of dividing x by 5, x=
2
kpkm@c99afc4984265e2:~/scripting$
```

## Q. Arithmetic operations using double parenthese



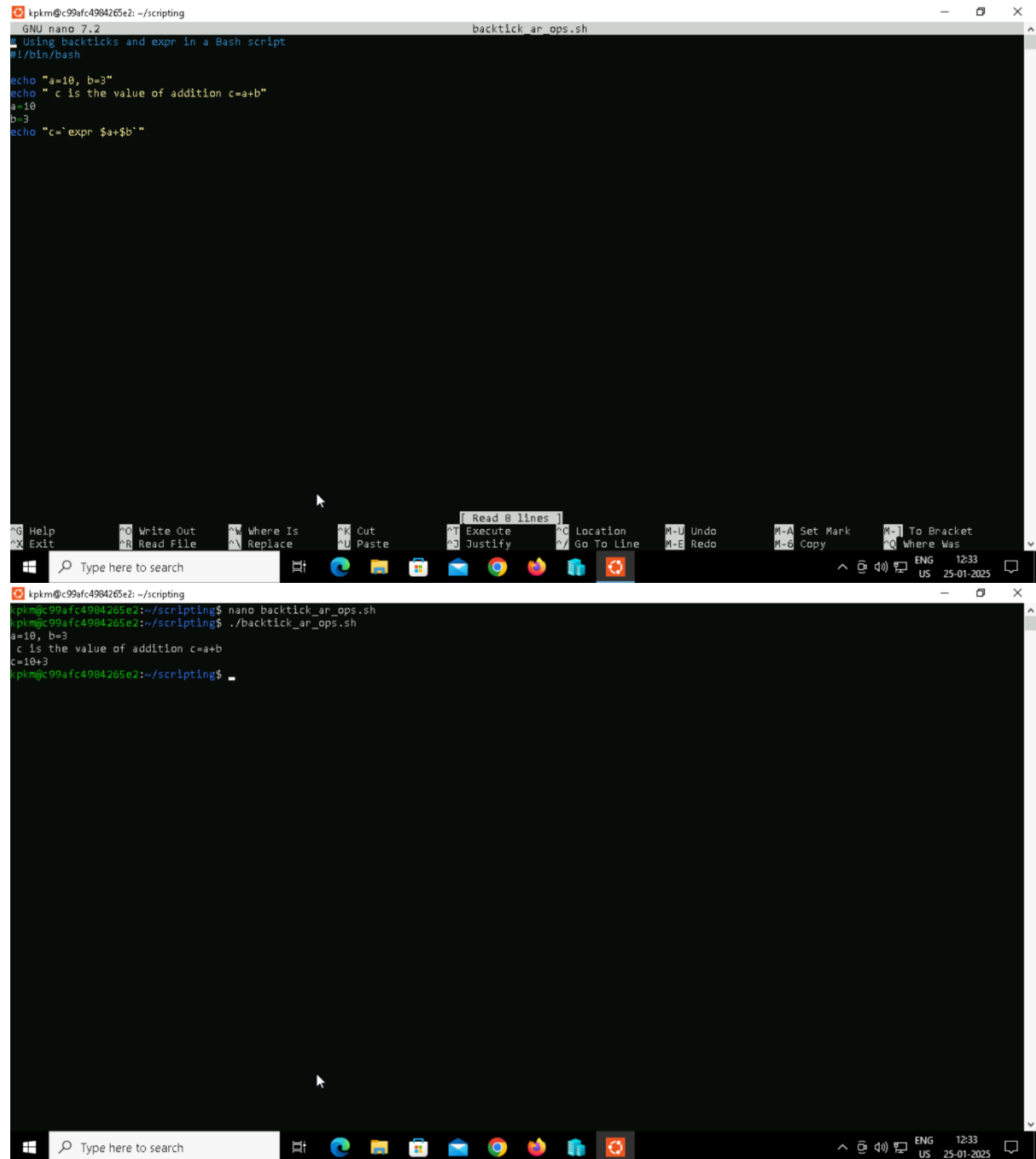
The image displays two terminal windows. The top window shows the creation of a script named `arithmetic_ops.sh` using the `nano` editor. The script defines variables `x=8` and `y=2` and performs various arithmetic operations using double parentheses `(( ))` for calculations. The bottom window shows the execution of this script, displaying the results of each operation.

```
GNU nano 7.2 arithmetic_ops.sh
#using double parenthese for arithmetic operations in a Bash shell script

#!/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 "Divison of x by y"
echo $(( $x / $y ))
echo "Exponentiation of x,y"
echo $(( $x % $y ))
echo "Incrementing x by 5, then x="
(( x+=5 ))
echo $x
echo "Decrementing x by 5, then x="
(( x-=5 ))
echo $x
echo "Multiply of x by 5, then x="
(( x*=5 ))
echo $x
echo "Dividing x by 5, x="
(( x/=5 ))
echo $x
echo "Remainder of Dividing x by 5, x="
(( x%=5 ))
echo $x
```

```
kpkmc99afc4984265e2:~/scripting$ ./arithmetic_ops.sh
x=8, y=2
Addition of x & y
10
Subtraction of x & y:
6
Multiplication of x & y:
16
Divison of x by y
4
Exponentiation of x,y
8
Incrementing x by 5, then x=
13
Decrementing x by 5, then x=
8
Multiply of x by 5, then x=
40
Dividing x by 5, x=
8
Remainder of Dividing x by 5, x=
3
kpkmc99afc4984265e2:~/scripting$
```

## Q. Using backticks and expr in Bash script



The image displays two screenshots of a Windows terminal window. The top screenshot shows the nano text editor editing a file named `backtick_ar_ops.sh`. The script content is as follows:

```
GNU nano 7.2 backtick_ar_ops.sh
Using backticks and expr in a Bash script
#!/bin/bash

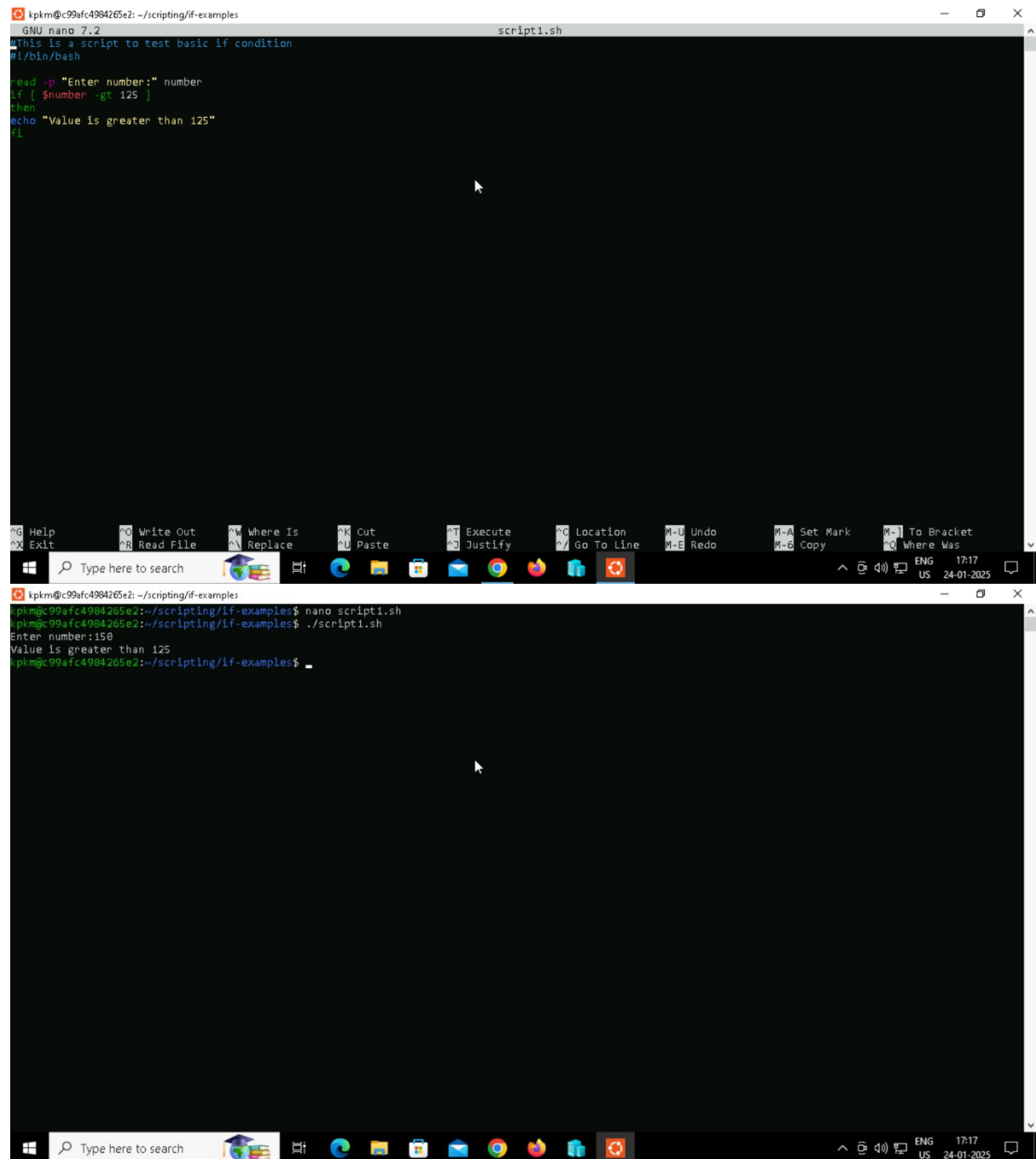
echo "a=10, b=3"
echo "c is the value of addition c=a+b"
a=10
b=3
echo "c=`expr $a+$b`"
```

The bottom screenshot shows the terminal after running the script. The output is:

```
kpkmc99afc4984265e2: ~/scripting
kpkmc99afc4984265e2:~/scripting$ nano backtick_ar_ops.sh
kpkmc99afc4984265e2:~/scripting$ ./backtick_ar_ops.sh
a=10, b=3
c is the value of addition c=a+b
c=10+3
kpkmc99afc4984265e2:~/scripting$
```

## IF Operations

### Q. Basic if operations



The image consists of two screenshots of a terminal window, likely from a Windows Subsystem for Linux (WSL) environment, showing the creation and execution of a basic if-condition script.

**Top Screenshot:** The terminal shows the user editing a file named `script1.sh` using the `nano` text editor. The script content is as follows:

```
#!/bin/bash

read -p "Enter number:" number
if [ $number -gt 125 ]
then
echo "Value is greater than 125"
fi
```

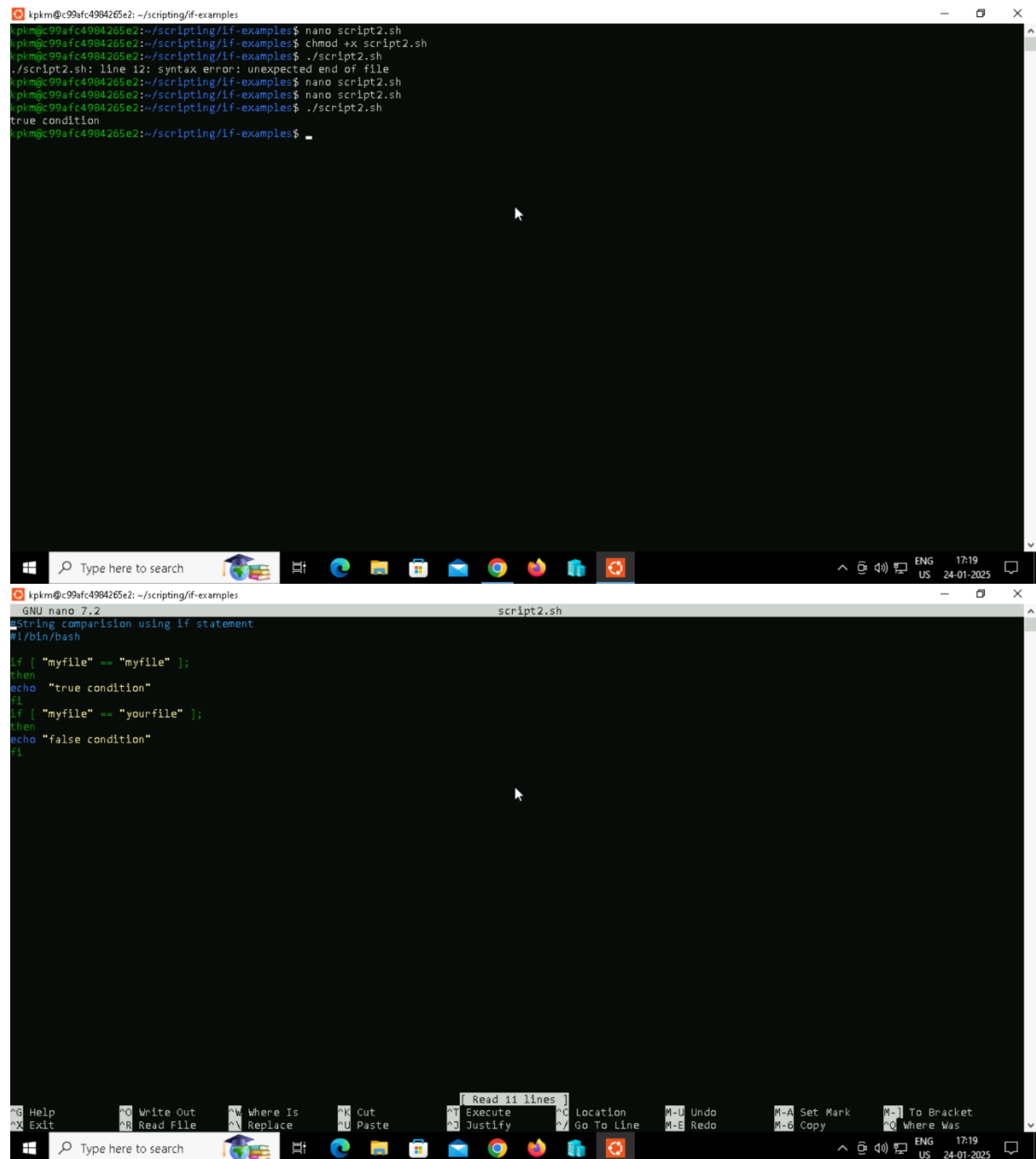
The terminal window title is `GNU nano 7.2 script1.sh`. The bottom status bar shows the time as 17:17 on 24-01-2025.

**Bottom Screenshot:** The terminal shows the user running the script `script1.sh` using the `./script1.sh` command. The output is:

```
kpk@c99afc4984265e2:~/scripting/if-examples$ nano script1.sh
kpk@c99afc4984265e2:~/scripting/if-examples$ ./script1.sh
Enter number:150
Value is greater than 125
kpk@c99afc4984265e2:~/scripting/if-examples$
```

The terminal window title is `kpk@c99afc4984265e2: ~/scripting/if-examples`. The bottom status bar shows the time as 17:17 on 24-01-2025.

## Q. String comparison using if conditions



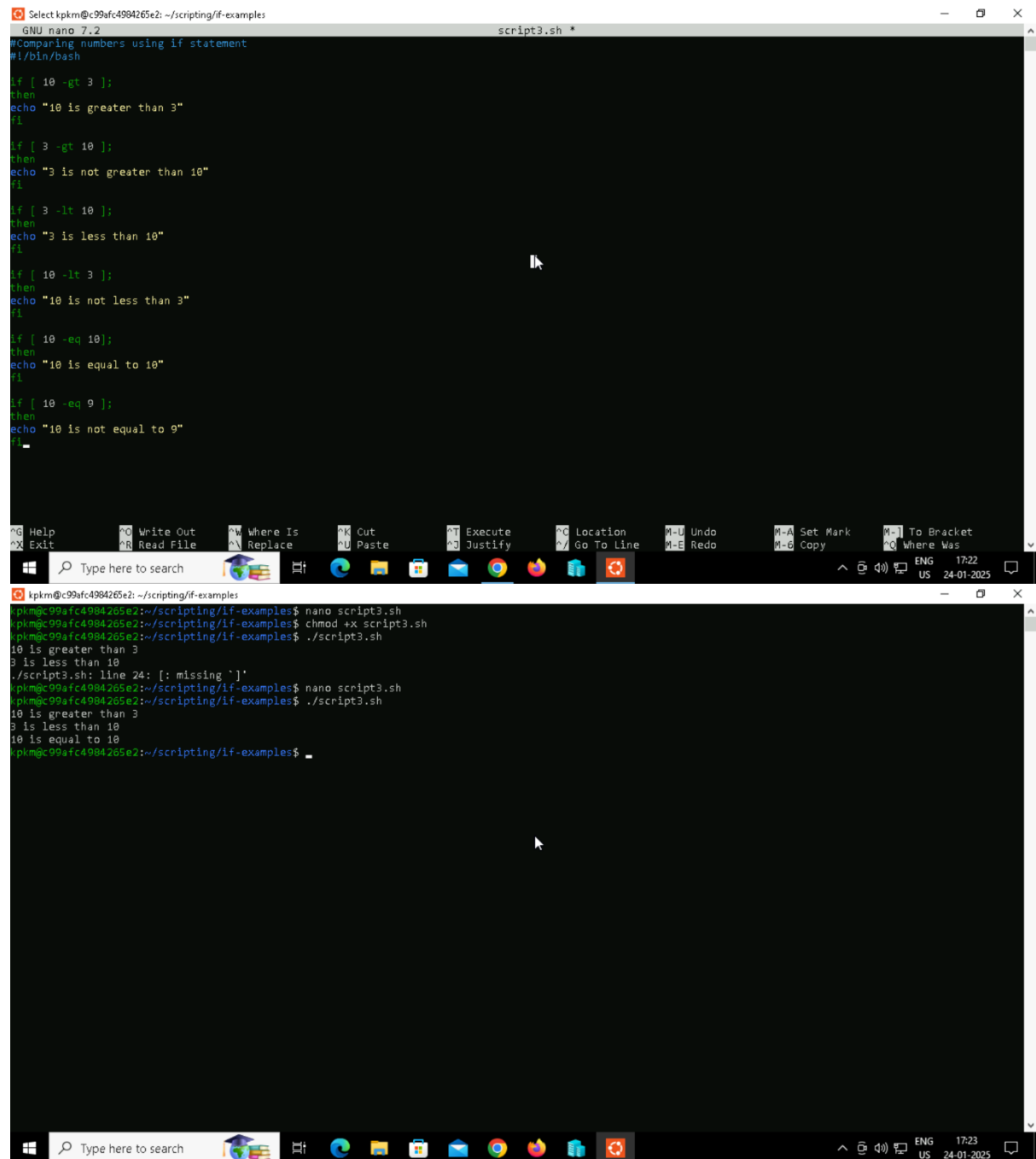
The screenshot displays a Windows desktop environment with two terminal windows open. The top window shows the user editing a script named `script2.sh` in the `nano` editor. The script contains a syntax error on line 12, which is an `if` statement for string comparison. The user runs the script, and it outputs `true condition`. The bottom window shows the user editing the same script, but with a different `if` statement that outputs `false condition`. The terminal windows are titled `kpkmc99afc4984265e2: ~/scripting/if-examples` and `GNU nano 7.2 script2.sh`. The Windows taskbar at the bottom shows the search bar, task view, and several application icons. The system tray on the right indicates the date and time as 24-01-2025, 17:19.

```
kpkmc99afc4984265e2: ~/scripting/if-examples
kpkmc99afc4984265e2:~/scripting/if-examples$ nano script2.sh
kpkmc99afc4984265e2:~/scripting/if-examples$ chmod +x script2.sh
kpkmc99afc4984265e2:~/scripting/if-examples$ ./script2.sh
./script2.sh: line 12: syntax error: unexpected end of file
kpkmc99afc4984265e2:~/scripting/if-examples$ nano script2.sh
kpkmc99afc4984265e2:~/scripting/if-examples$ nano script2.sh
kpkmc99afc4984265e2:~/scripting/if-examples$ ./script2.sh
true condition
kpkmc99afc4984265e2:~/scripting/if-examples$
```

```
GNU nano 7.2 script2.sh
String comparison using if statement
#!/bin/bash

if [ "myfile" == "myfile" ];
then
echo "true condition"
fi
if [ "myfile" == "yourfile" ];
then
echo "false condition"
fi
```

## Q. Comparing numbers using if conditions



The image consists of two screenshots of a terminal window. The top screenshot shows the nano text editor editing a file named script3.sh. The script contains several if statements for comparing the numbers 10 and 3. The bottom screenshot shows the terminal after running the script, displaying the output of the if statements. The terminal window has a title bar that reads "Select kpkmc99afc4984265e2: ~/scripting/if-examples".

```
GNU nano 7.2 script3.sh
#Comparing numbers using if statement
#!/bin/bash

if [ 10 -gt 3 ];
then
echo "10 is greater than 3"
fi

if [ 3 -gt 10 ];
then
echo "3 is not greater than 10"
fi

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

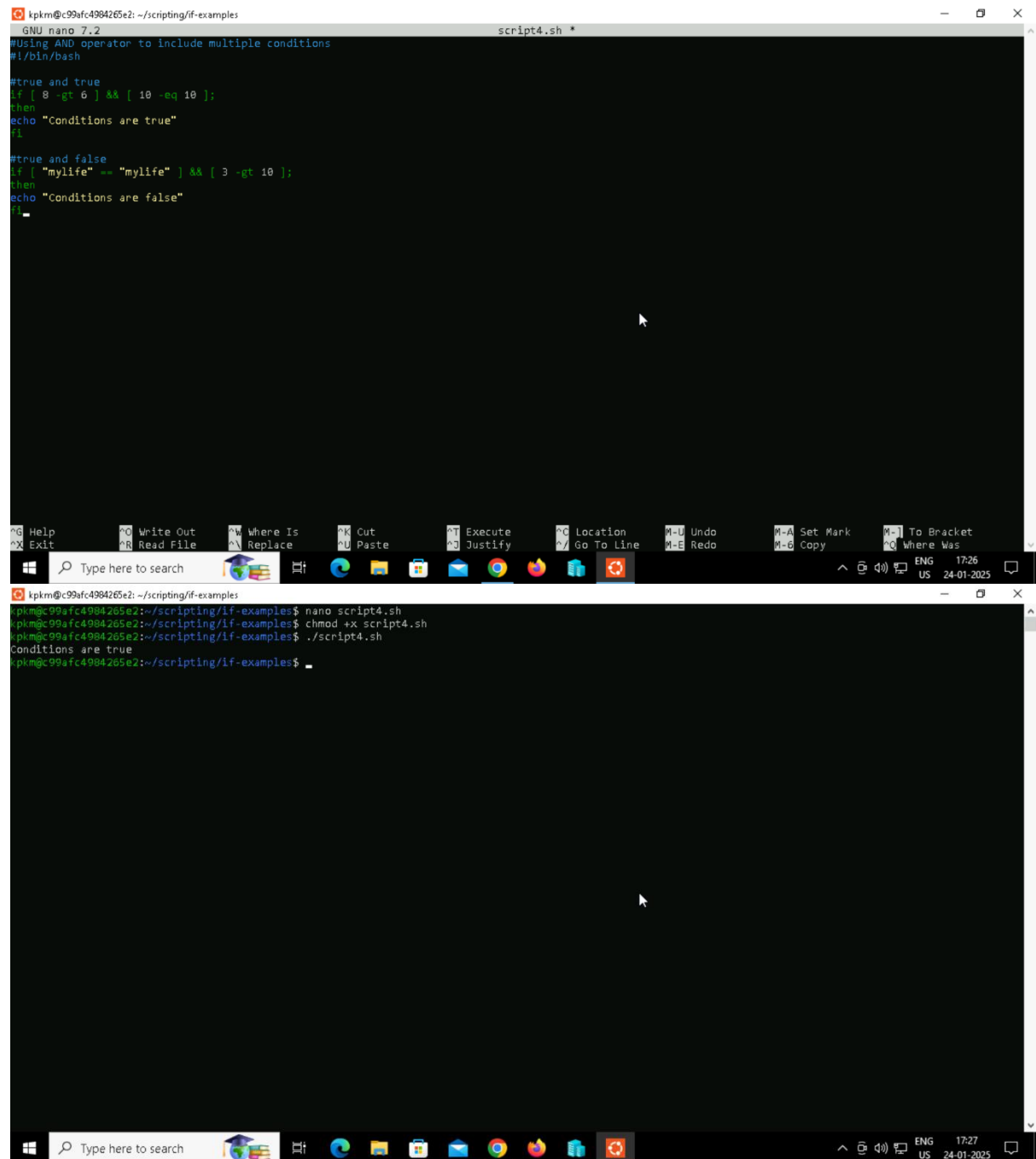
if [ 10 -lt 3 ];
then
echo "10 is not less than 3"
fi

if [ 10 -eq 10 ];
then
echo "10 is equal to 10"
fi

if [ 10 -eq 9 ];
then
echo "10 is not equal to 9"
fi
_
```

```
kpkmc99afc4984265e2:~/scripting/if-examples$ nano script3.sh
kpkmc99afc4984265e2:~/scripting/if-examples$ chmod +x script3.sh
kpkmc99afc4984265e2:~/scripting/if-examples$ ./script3.sh
10 is greater than 3
3 is less than 10
./script3.sh: line 24: [: missing `]'
kpkmc99afc4984265e2:~/scripting/if-examples$ nano script3.sh
kpkmc99afc4984265e2:~/scripting/if-examples$ ./script3.sh
10 is greater than 3
3 is less than 10
10 is equal to 10
kpkmc99afc4984265e2:~/scripting/if-examples$ _
```

## Q. Using AND operator



The image consists of two screenshots of a terminal window. The top screenshot shows the nano text editor editing a file named script4.sh. The script contains two conditional blocks. The first block checks if 8 is greater than 6 and 10 is equal to 10, which is true, and prints "Conditions are true". The second block checks if the variable mylife is equal to mylife and 3 is greater than 10, which is false, and prints "Conditions are false". The bottom screenshot shows the terminal after the script has been executed. The user runs 'nano script4.sh' to edit the file, then 'chmod +x script4.sh' to make it executable, and finally './script4.sh' to run it. The output shows "Conditions are true" followed by a prompt character.

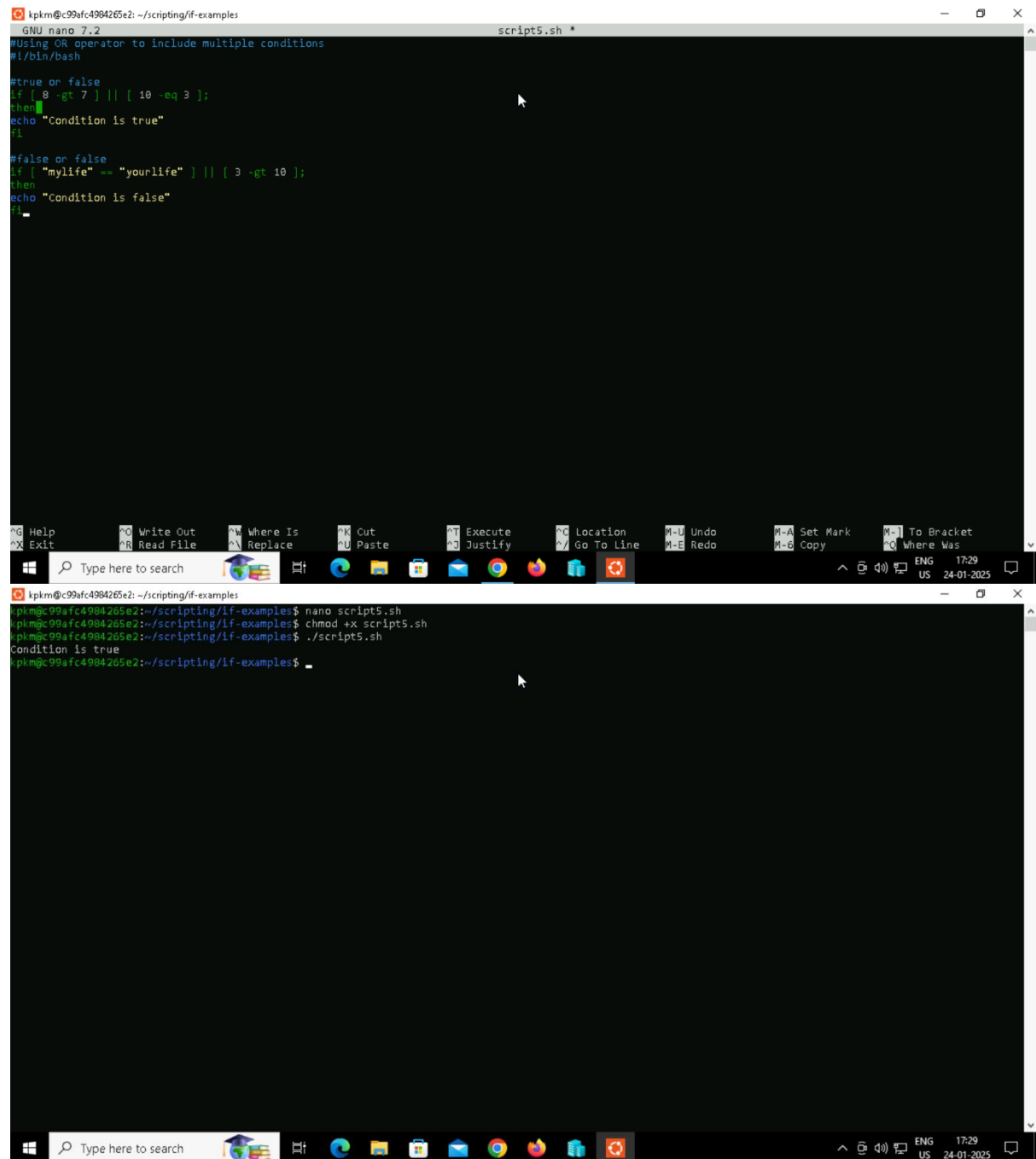
```
kpk@c99afc4984265e2: ~/scripting/lf-examples
GNU nano 7.2 script4.sh
#Using AND operator to include multiple conditions
#!/bin/bash

#true and true
if [ 8 -gt 6 ] && [ 10 -eq 10 ];
then
echo "Conditions are true"
fi

#true and false
if [ "mylife" == "mylife" ] && [ 3 -gt 10 ];
then
echo "Conditions are false"
fi

kpk@c99afc4984265e2:~/scripting/lf-examples$ nano script4.sh
kpk@c99afc4984265e2:~/scripting/lf-examples$ chmod +x script4.sh
kpk@c99afc4984265e2:~/scripting/lf-examples$ ./script4.sh
Conditions are true
kpk@c99afc4984265e2:~/scripting/lf-examples$
```

## Q. Using OR operator



The image shows a Windows terminal window with two stacked screenshots. The top screenshot shows the nano text editor editing a file named `script5.sh`. The script content is as follows:

```
GNU nano 7.2 script5.sh *
#Using OR operator to include multiple conditions
#!/bin/bash

#true or false
if [ 8 -gt 7 ] || [ 10 -eq 3 ];
then
echo "Condition is true"
fi

#false or false
if [ "mylife" == "yourlife" ] || [ 3 -gt 10 ];
then
echo "Condition is false"
fi
_
```

The bottom screenshot shows the terminal after running the script. The user has executed `chmod +x script5.sh` and then `./script5.sh`. The output of the script is:

```
kpk@c99afc4984265e2: ~/scripting/if-examples
kpk@c99afc4984265e2:~/scripting/if-examples$ nano script5.sh
kpk@c99afc4984265e2:~/scripting/if-examples$ chmod +x script5.sh
kpk@c99afc4984265e2:~/scripting/if-examples$ ./script5.sh
Condition is true
kpk@c99afc4984265e2:~/scripting/if-examples$ _
```

The Windows taskbar at the bottom of the terminal window shows the search bar, taskbar icons for various applications, and system tray information including the date and time (17:29, 24-01-2025).



## Q. Using AND and OR operator

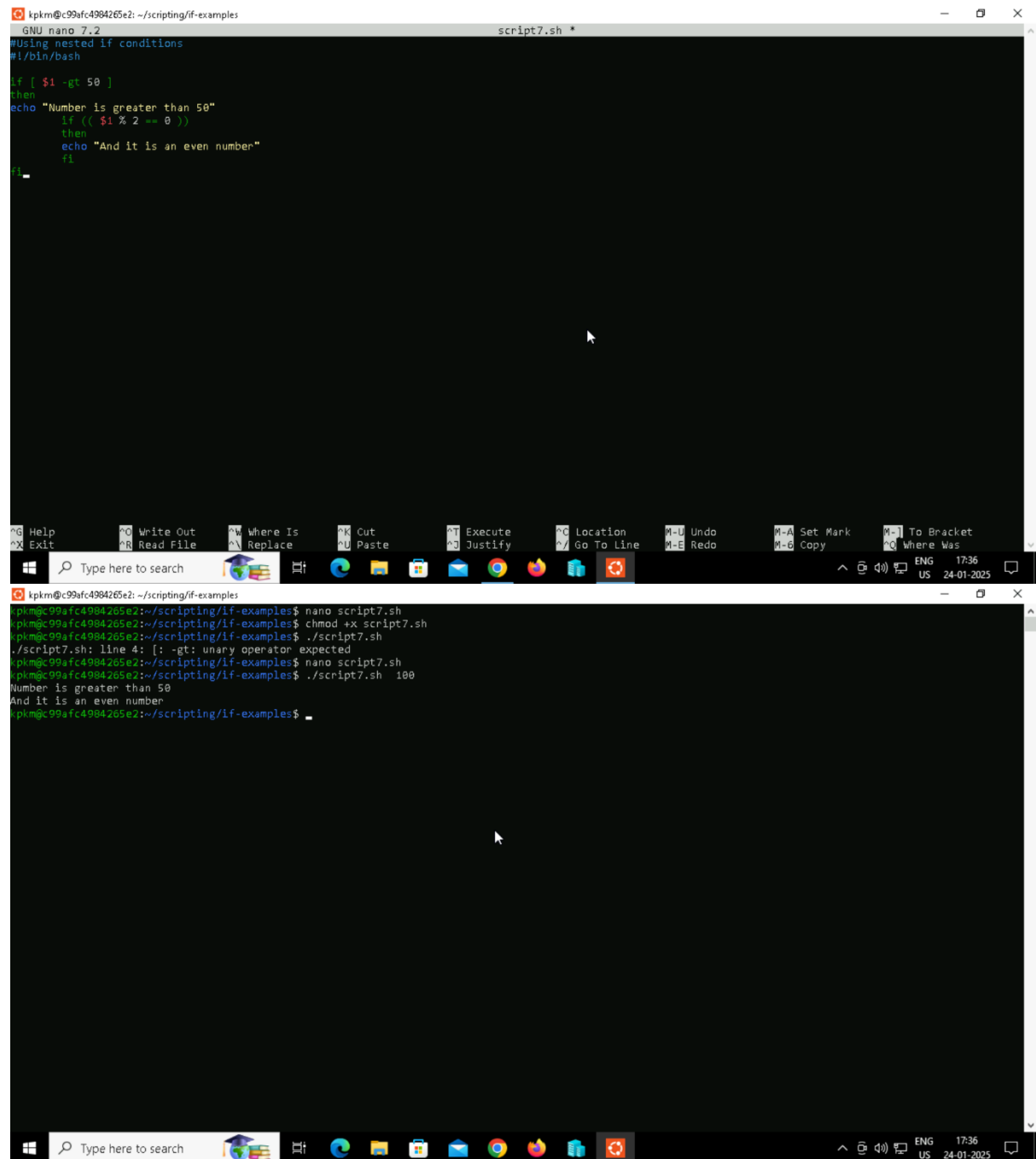
```
GNU nano 7.2 script6.sh
#Using AND and OR together
#!/bin/bash

# T && F || F || T
if [[ 10 -eq 10 && 5 -gt 4 || 3 -eq 4 || 3 -lt 6 ]];
then
echo "Condition is true"
fi

# T && F || F
if [[ 8 -eq 8 && 8 -gt 10 || -lt 5 ]];
then
echo "Condition is false"
fi
```

```
kpkmc99afc4984265e2: ~/scripting/lf-examples
kpkmc99afc4984265e2:~/scripting/lf-examples$ nano script6.sh
kpkmc99afc4984265e2:~/scripting/lf-examples$ chmod +x script6.sh
kpkmc99afc4984265e2:~/scripting/lf-examples$ ./script6.sh
Condition is true
./script6.sh: line 11: conditional binary operator expected
./script6.sh: line 11: syntax error near `5'
./script6.sh: line 11: `if [[ 8 -eq 8 && 8 -gt 10 || -lt 5 ]];'
kpkmc99afc4984265e2:~/scripting/lf-examples$ nano script6.sh
kpkmc99afc4984265e2:~/scripting/lf-examples$ ./script6.sh
Condition is true
kpkmc99afc4984265e2:~/scripting/lf-examples$
```

## Q. Using nested if conditions



The image consists of two screenshots of a terminal window, likely from a Windows Subsystem for Linux (WSL) environment, showing the process of creating and running a shell script.

**Top Screenshot:** The terminal shows the GNU nano 7.2 editor editing a file named `script7.sh`. The script content is as follows:

```
#!/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
```

The terminal window includes a menu bar with options like Help, Exit, Write Out, Read File, Where Is, Replace, Cut, Paste, Execute, Justify, Location, Go To Line, Undo, Redo, Set Mark, Copy, and To Bracket. The system tray at the bottom shows the time as 17:36 on 24-01-2025.

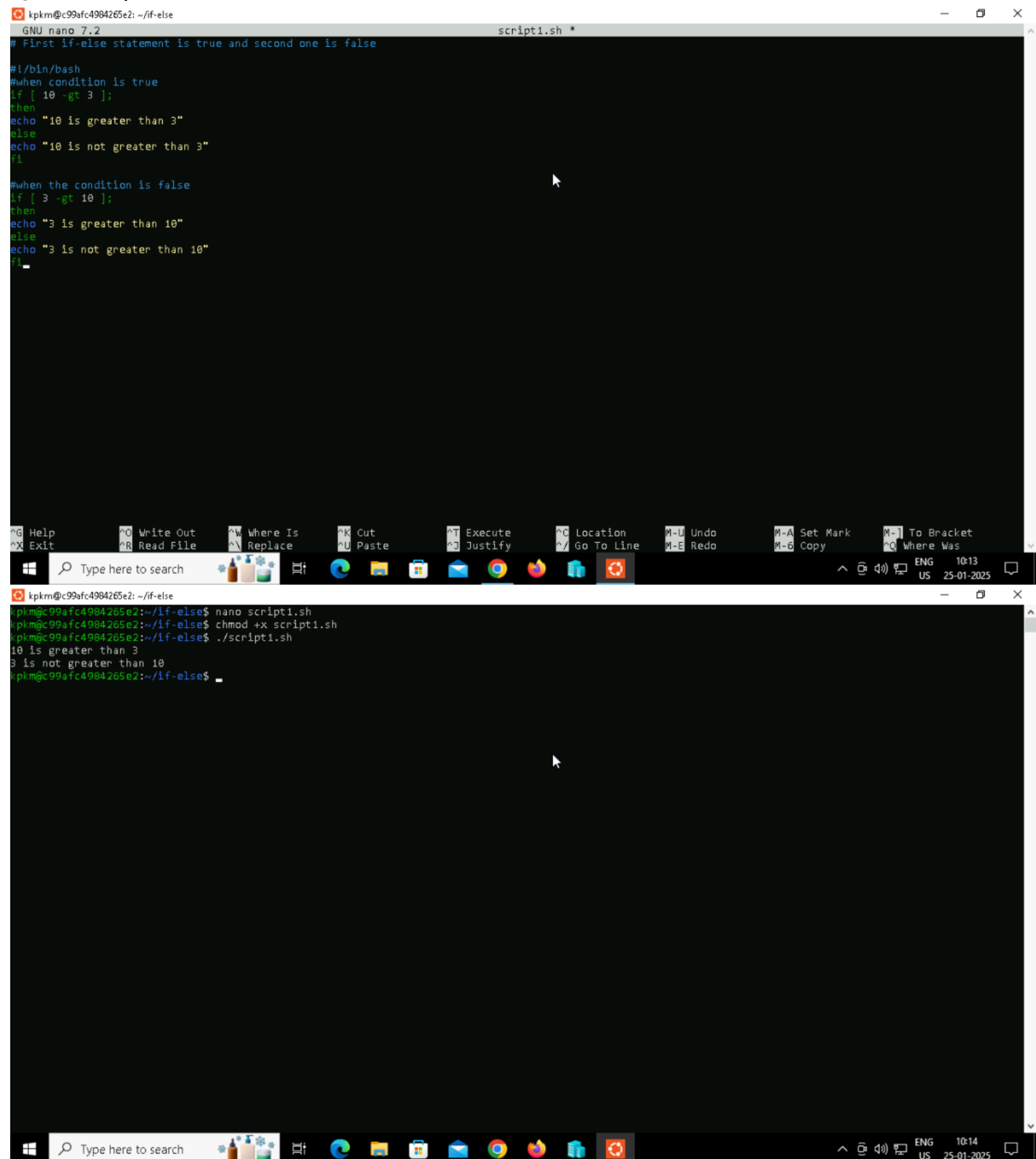
**Bottom Screenshot:** This screenshot shows the terminal after the script has been executed. The commands and their outputs are:

```
kpkmc99afc4984265e2: ~/scripting/if-examples$ nano script7.sh
kpkmc99afc4984265e2: ~/scripting/if-examples$ chmod +x script7.sh
kpkmc99afc4984265e2: ~/scripting/if-examples$ ./script7.sh
./script7.sh: line 4: [: -gt: unary operator expected
kpkmc99afc4984265e2: ~/scripting/if-examples$ nano script7.sh
kpkmc99afc4984265e2: ~/scripting/if-examples$ ./script7.sh 100
Number is greater than 50
And it is an even number
kpkmc99afc4984265e2: ~/scripting/if-examples$
```

The error message `./script7.sh: line 4: [: -gt: unary operator expected` indicates a syntax error in the original script. The corrected script should use `if [ $1 -gt 50 ]` instead of `if (( $1 % 2 == 0 ))` for the first condition, and the second condition should be `if (( $1 % 2 == 0 ))` instead of `if (( $1 % 2 == 0 ))` (which is correct in the original image). The system tray at the bottom shows the time as 17:36 on 24-01-2025.

## If-Else and Elif operations

### Q. If else operations



The image consists of two screenshots of a terminal window. The top screenshot shows the nano text editor editing a file named script1.sh. The script contains two if-else blocks. The first block checks if 10 is greater than 3, and the second block checks if 3 is greater than 10. The bottom screenshot shows the terminal after running the script, displaying the output of the if-else operations.

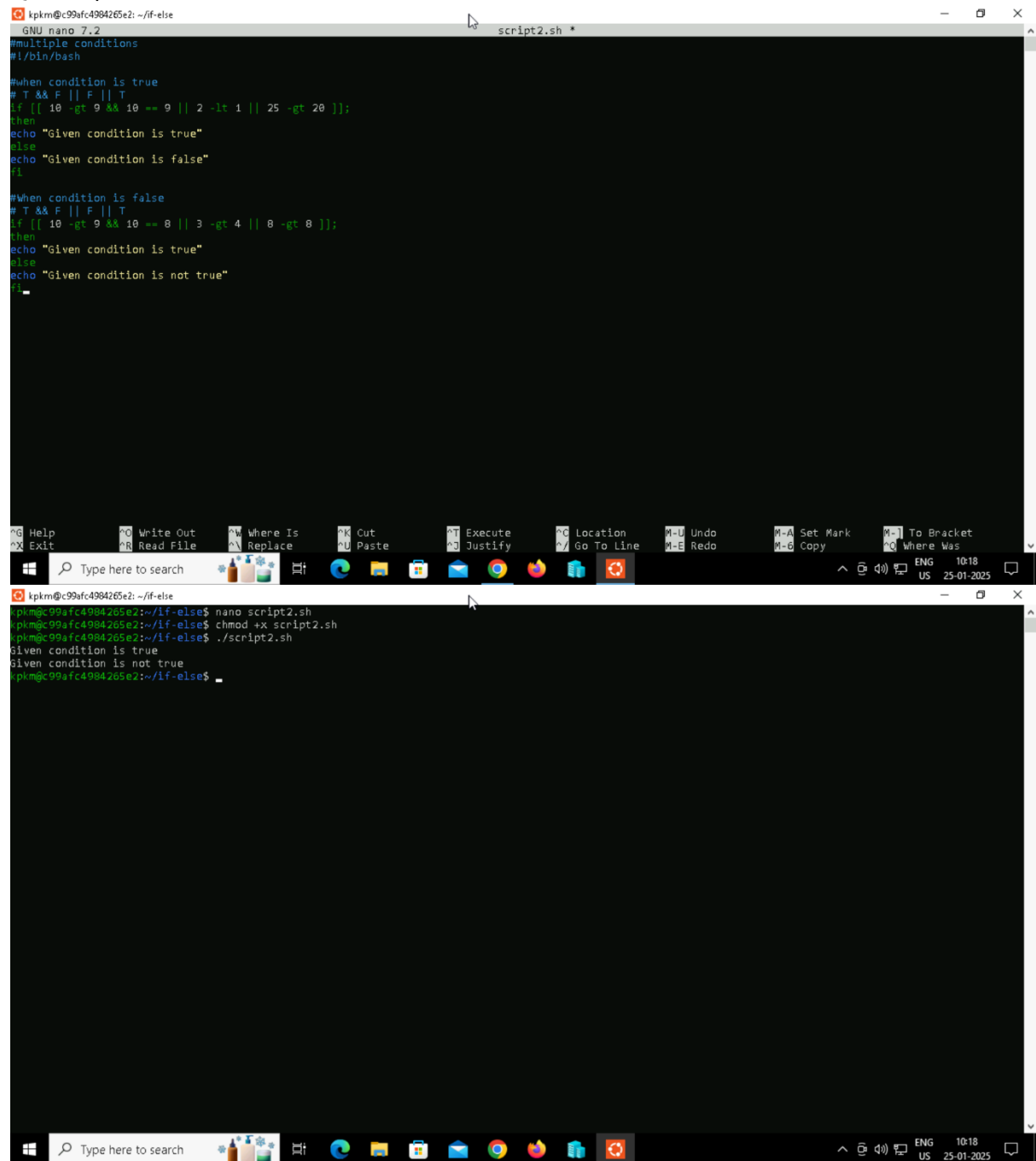
```
GNU nano 7.2 script1.sh
# First if/else statement is true and second one is false

#!/bin/bash
#when 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
_
```

```
kpk@k99af4984265e2: ~/if-else
kpk@k99af4984265e2:~/if-else$ nano script1.sh
kpk@k99af4984265e2:~/if-else$ chmod +x script1.sh
kpk@k99af4984265e2:~/if-else$ ./script1.sh
10 is greater than 3
3 is not greater than 10
kpk@k99af4984265e2:~/if-else$
```

## Q. Multiple if-else conditions



The screenshot displays a Windows desktop environment with two terminal windows. The top window is a nano editor editing a file named 'script2.sh'. The script contains two if-else blocks. The first block checks if the variable 'T' is true (T == 9) and prints 'Given condition is true' if true, or 'Given condition is false' if false. The second block checks if 'T' is false (T == 8) and prints 'Given condition is true' if true, or 'Given condition is not true' if false. The bottom terminal window shows the execution of the script. It first runs 'nano script2.sh' to create the file, then 'chmod +x script2.sh' to make it executable, and finally './script2.sh' to run it. The output of the script is 'Given condition is true' followed by 'Given condition is not true'.

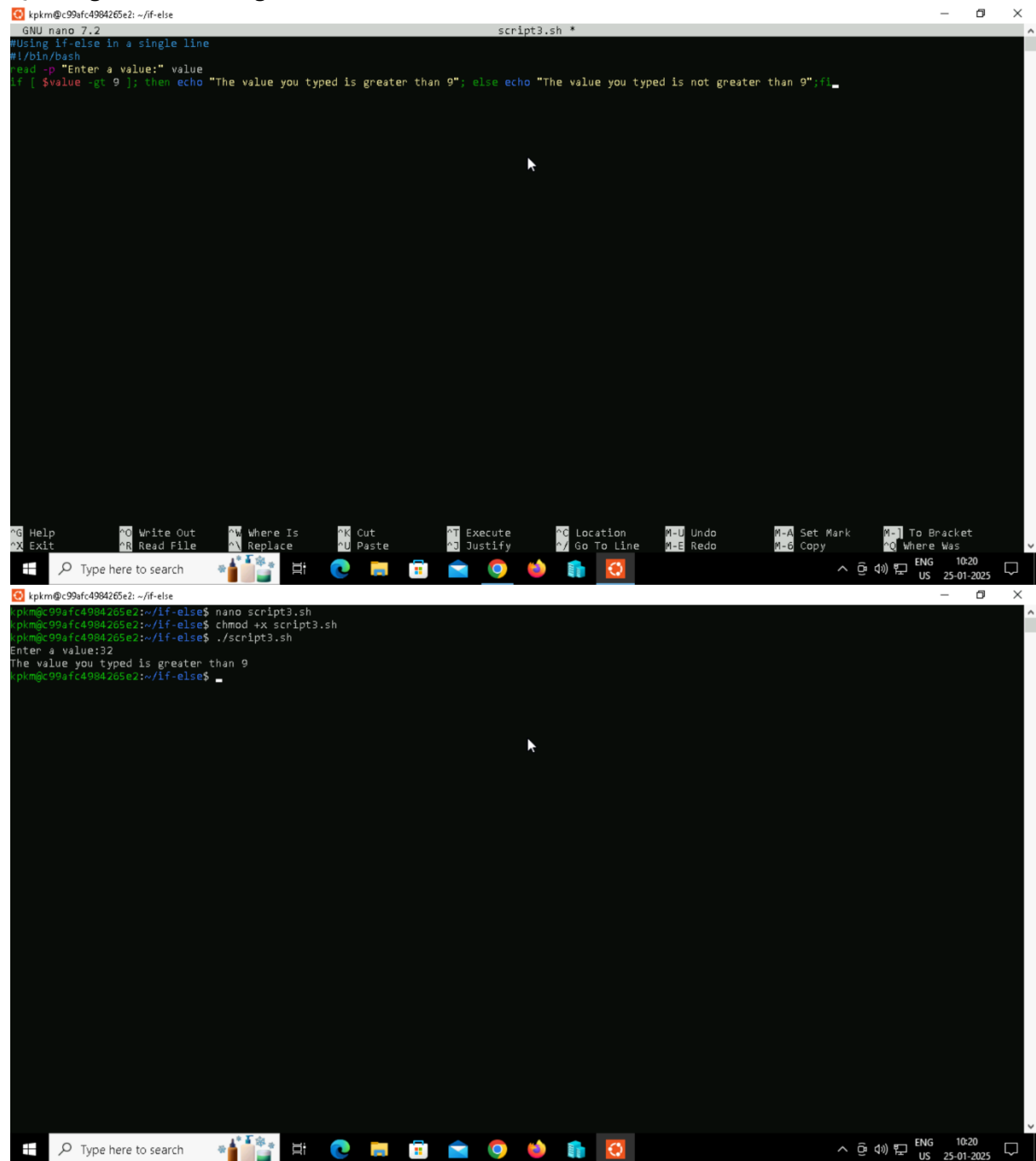
```
GNU nano 7.2 script2.sh
#multiple conditions
#!/bin/bash

#when condition is true
# T == 9 || F || T
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
# T == 8 || F || T
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
_
```

```
kpkmc99afc4984265e2: ~/if-else
kpkmc99afc4984265e2:~/if-else$ nano script2.sh
kpkmc99afc4984265e2:~/if-else$ chmod +x script2.sh
kpkmc99afc4984265e2:~/if-else$ ./script2.sh
Given condition is true
Given condition is not true
kpkmc99afc4984265e2:~/if-else$
```

## Q. Using if-else in a single line



The image consists of two screenshots of a terminal window. The top screenshot shows the nano text editor editing a file named script3.sh. The content of the file is a shell script that uses a single-line if-else statement to check if a value is greater than 9. The bottom screenshot shows the terminal after the script has been executed with the input value 32, resulting in the output: "The value you typed is greater than 9".

```
GNU nano 7.2 script3.sh
#Using if-else in a single line
#!/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
```

```
kpkmc99afc4984265e2: ~/if-else
kpkmc99afc4984265e2:~/if-else$ nano script3.sh
kpkmc99afc4984265e2:~/if-else$ chmod +x script3.sh
kpkmc99afc4984265e2:~/if-else$ ./script3.sh
Enter a value:32
The value you typed is greater than 9
kpkmc99afc4984265e2:~/if-else$
```

## Q. Nested if-else statement

```
kpkm@c99afc4984265e2: ~/if-else
kpkm@c99afc4984265e2:~/if-else$ nano script4.sh
kpkm@c99afc4984265e2:~/if-else$ chmod +x script4.sh
kpkm@c99afc4984265e2:~/if-else$ ./script4.sh
Enter a value:32
The value you typed is greater than 9
kpkm@c99afc4984265e2:~/if-else$ ./script4.sh
Enter a value:8
The value you typed is not greater than 9
kpkm@c99afc4984265e2:~/if-else$ ./script4.sh
Enter a value:10
10>9, 10<11
kpkm@c99afc4984265e2:~/if-else$
```

```
GNU nano 7.2 script4.sh
Nested if-else statement
#!/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
```

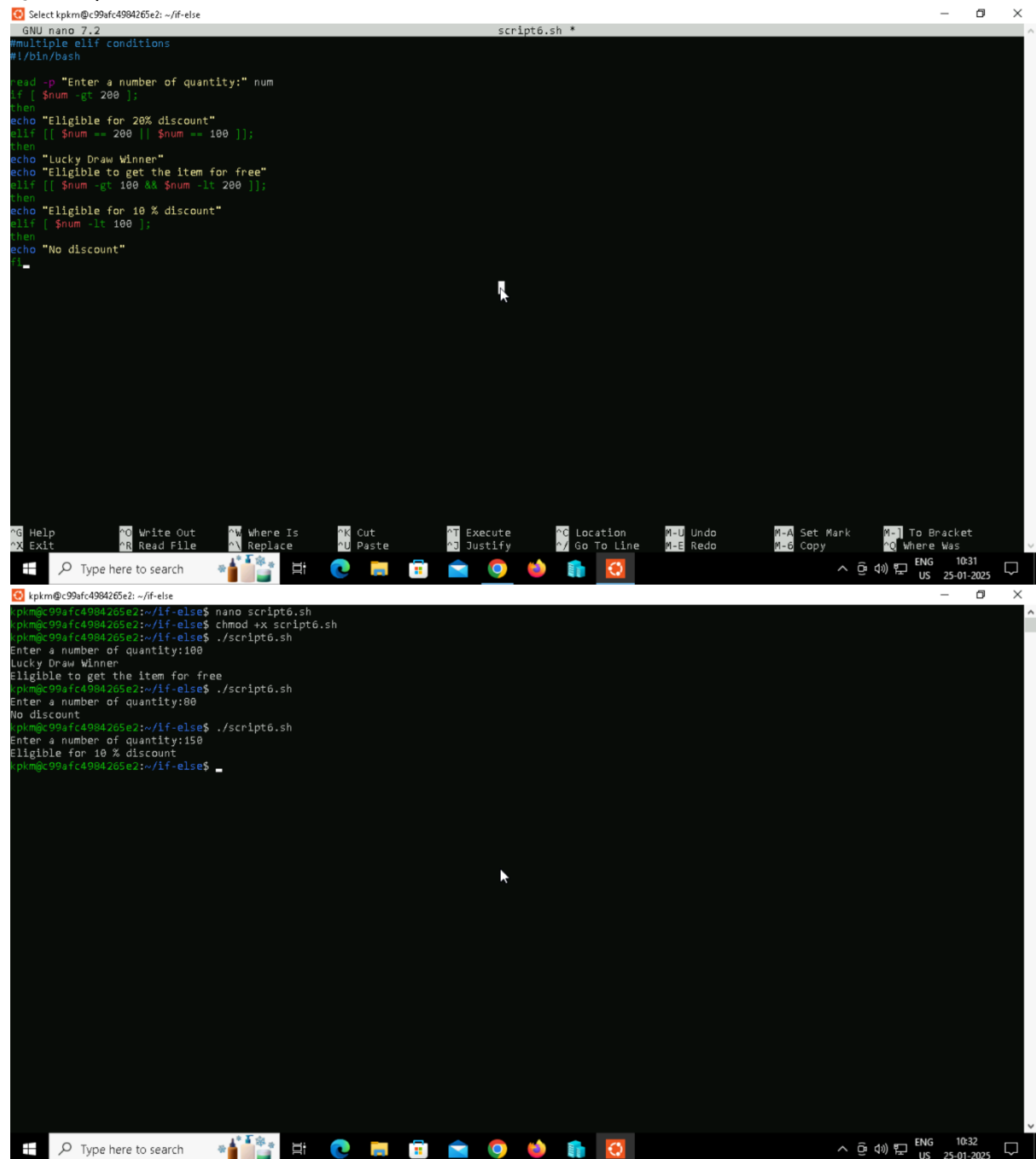
## Q. El-if expressions

```
kpkm@c99afc4984265e2: ~/if-else
GNU nano 7.2 script5.sh
Using elif conditions
#!/bin/bash

read -p "Enter a number of quantity:" num
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
```

```
kpkm@c99afc4984265e2: ~/if-else
kpkm@c99afc4984265e2:~/if-else$ nano script5.sh
kpkm@c99afc4984265e2:~/if-else$ chmod +x script5.sh
kpkm@c99afc4984265e2:~/if-else$ ./script5.sh
Enter a number of quantity:100
./script5.sh: line 13: unexpected EOF while looking for matching `"'
kpkm@c99afc4984265e2:~/if-else$ nano script5.sh
kpkm@c99afc4984265e2:~/if-else$ ./script5.sh
Enter a number of quantity:100
Lucky draw winner
Eligible to get the item for free
kpkm@c99afc4984265e2:~/if-else$ ./script5.sh
Enter a number of quantity:101
Eligible for 10% discount
kpkm@c99afc4984265e2:~/if-else$ ./script5.sh
Enter a number of quantity:99
Eligible for 5% discount
kpkm@c99afc4984265e2:~/if-else$
```

## Q. Multiple el-if condition



```
GNU nano 7.2 script6.sh *
#multiple elif conditions
#!/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

kpkmc99afc4984265e2: ~/if-else
kpkmc99afc4984265e2:~/if-else$ nano script6.sh
kpkmc99afc4984265e2:~/if-else$ chmod +x script6.sh
kpkmc99afc4984265e2:~/if-else$ ./script6.sh
Enter a number of quantity:100
Lucky Draw Winner
Eligible to get the item for free
kpkmc99afc4984265e2:~/if-else$ ./script6.sh
Enter a number of quantity:80
No discount
kpkmc99afc4984265e2:~/if-else$ ./script6.sh
Enter a number of quantity:150
Eligible for 10 % discount
kpkmc99afc4984265e2:~/if-else$
```