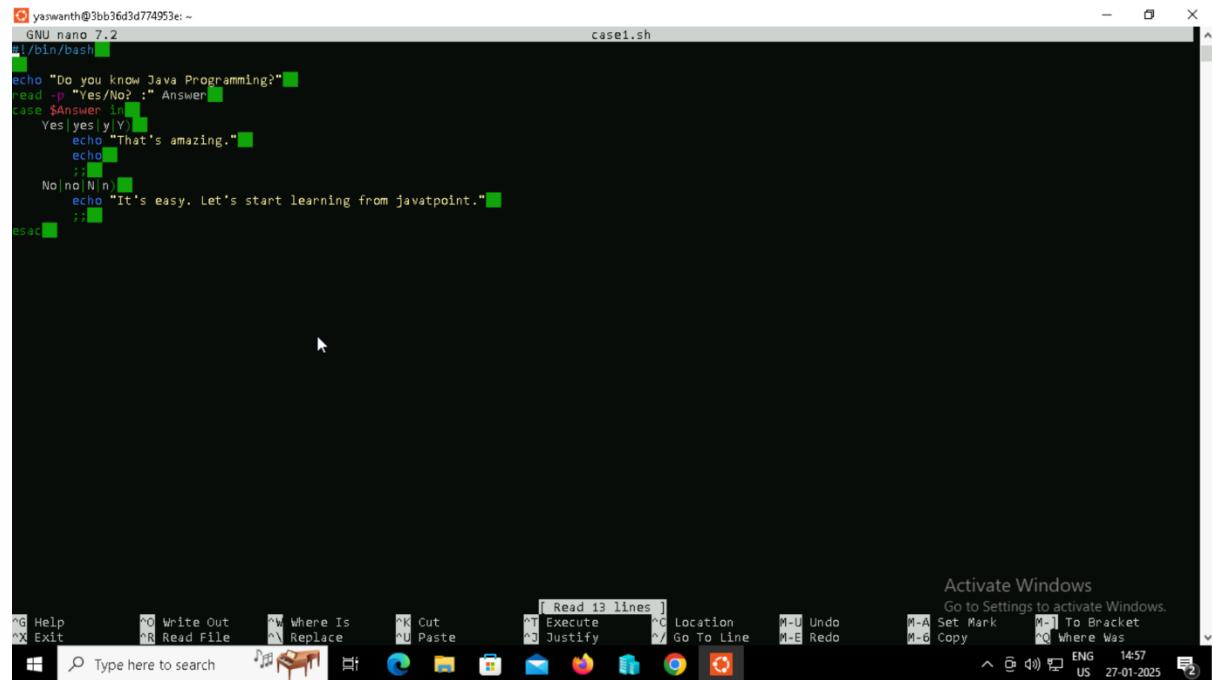
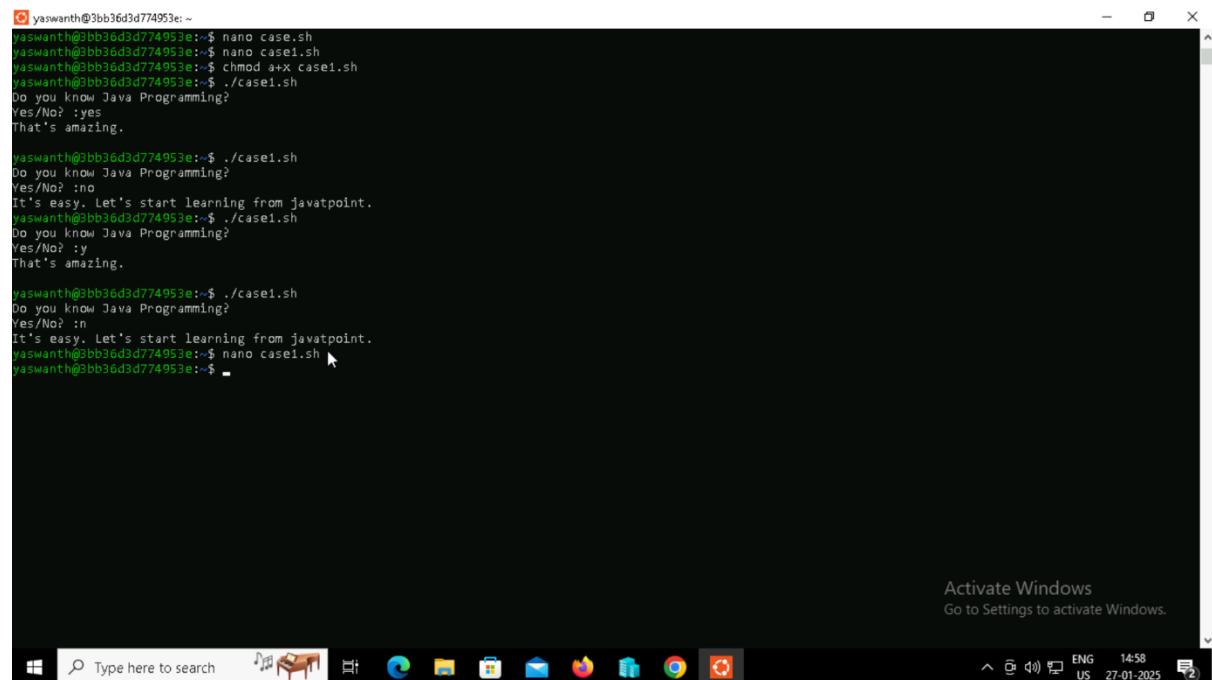


BASH PROGRAM CASE AND FOR LOOPS:

1. In this example, we have defined a simple scenario to demonstrate the use of the Case statement.



```
yaswanth@3bb36d3d774953e:~$ nano 7.2
1/bin/bash
case1.sh
echo "Do you know Java Programming?"
read -p "Yes/No? :" Answer
case $Answer in
  Yes|yes|y|Y)
    echo "That's amazing."
    echo ;;
  No|no|N|n)
    echo "It's easy. Let's start learning from javatpoint."
  ;;
esac
```

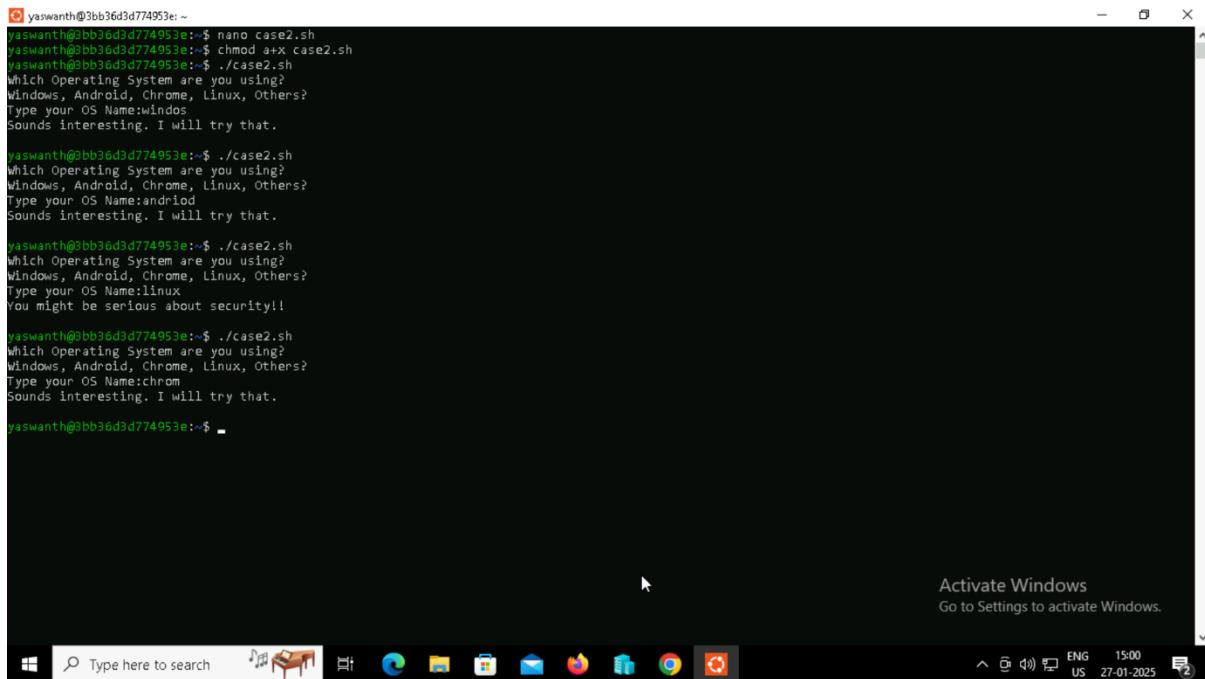


```
yaswanth@3bb36d3d774953e:~$ nano case.sh
yaswanth@3bb36d3d774953e:~$ nano case1.sh
yaswanth@3bb36d3d774953e:~$ chmod a+x case1.sh
yaswanth@3bb36d3d774953e:~$ ./case1.sh
Do you know Java Programming?
Yes/No? :yes
That's amazing.

yaswanth@3bb36d3d774953e:~$ ./case1.sh
Do you know Java Programming?
Yes/No? :no
It's easy. Let's start learning from javatpoint.
yaswanth@3bb36d3d774953e:~$ ./case1.sh
Do you know Java Programming?
Yes/No? :y
That's amazing.

yaswanth@3bb36d3d774953e:~$ ./case1.sh
Do you know Java Programming?
Yes/No? :n
It's easy. Let's start learning from javatpoint.
yaswanth@3bb36d3d774953e:~$ nano case1.sh
```

2.In this example, we have defined a combined scenario where there is also a default case when no previous matched case is found.



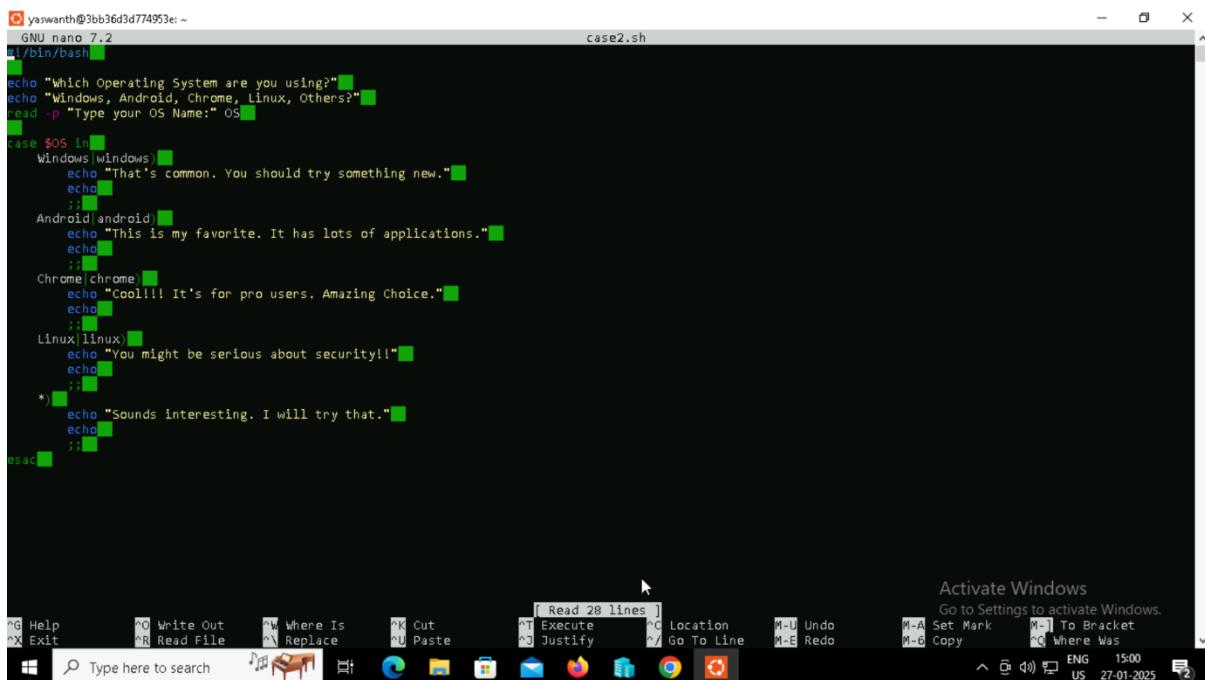
```
yaswanth@3bb36d3d774953e: ~
yaswanth@3bb36d3d774953e: $ nano case2.sh
yaswanth@3bb36d3d774953e: $ chmod a+x case2.sh
yaswanth@3bb36d3d774953e: $ ./case2.sh
Which Operating System are you using?
Windows, Android, Chrome, Linux, Others?
Type your OS Name:windows
Sounds interesting. I will try that.

yaswanth@3bb36d3d774953e: $ ./case2.sh
Which Operating System are you using?
Windows, Android, Chrome, Linux, Others?
Type your OS Name:android
Sounds interesting. I will try that.

yaswanth@3bb36d3d774953e: $ ./case2.sh
Which Operating System are you using?
Windows, Android, Chrome, Linux, Others?
Type your OS Name:linux
You might be serious about security!!

yaswanth@3bb36d3d774953e: $ ./case2.sh
Which Operating System are you using?
Windows, Android, Chrome, Linux, Others?
Type your OS Name:chrom
Sounds interesting. I will try that.

yaswanth@3bb36d3d774953e: ~
```



```
yaswanth@3bb36d3d774953e: ~
GNU nano 7.2
#!/bin/bash

echo "Which Operating System are you using?"
echo "Windows, Android, Chrome, Linux, Others?"
read -p "Type your OS Name:" OS

case $OS in
    Windows)
        echo "That's common. You should try something new."
        echo ;;
    Android)
        echo "This is my favorite. It has lots of applications."
        echo ;;
    Chrome)
        echo "Cool!!! It's for pro users. Amazing Choice."
        echo ;;
    Linux)
        echo "You might be serious about security!!"
        echo ;;
    *)
        echo "Sounds interesting. I will try that."
        echo ;;
esac
```

3. Basic 'For Loop' Example in bash

```
yaswanth@0bb36d3d774953e:~$ nano for1.sh
yaswanth@0bb36d3d774953e:~$ chmod a+x for1.sh
yaswanth@0bb36d3d774953e:~$ ./for1.sh
Start
learning
from
Javatpoint.
Thank You.
yaswanth@0bb36d3d774953e:~$
```

The screenshot shows a terminal window on a Windows desktop. The title bar says "Activate Windows Go to Settings to activate Windows." The taskbar at the bottom has icons for File Explorer, Edge, File Manager, Mail, Firefox, and a Start button.

```
yaswanth@0bb36d3d774953e:~$ GNU nano 7.2
learn="Start learning from Javatpoint."
for learn in $learn
do
echo $learn
done
echo "Thank You."
```

The screenshot shows a terminal window on a Windows desktop. The title bar says "Activate Windows Go to Settings to activate Windows." The taskbar at the bottom has icons for Help, Exit, Write Out, Read File, Where Is, Replace, Cut, Paste, Read 8 lines, Execute, Justify, Location, Undo, Redo, Set Mark, To Bracket, Copy, Where Was, File Explorer, Edge, File Manager, Mail, Firefox, and a Start button.

4. For Loop to Read a Range with Increment/Decrement:

```
yaswanth@3bb36d3d774953e:~  
yaswanth@3bb36d3d774953e:~$ nano for3.sh  
yaswanth@3bb36d3d774953e:~$ chmod +x for3.sh  
yaswanth@3bb36d3d774953e:~$ ./for3.sh  
10  
9  
6  
4  
2  
0  
yaswanth@3bb36d3d774953e:~$
```

The screenshot shows a terminal window on a Windows operating system. The terminal prompt is "yaswanth@3bb36d3d774953e:~". The user runs "nano for3.sh" to edit a script, changes its mode to executable with "chmod +x for3.sh", and then executes it with "./for3.sh". The output of the script is displayed in the terminal: "10", "9", "6", "4", "2", and "0". The taskbar at the bottom shows various application icons like File Explorer, Edge, and Firefox.

```
yaswanth@3bb36d3d774953e:~  
GNU nano 7.2  
for3.sh  
num in {10..0..2}  
do  
echo $num  
done
```

The screenshot shows a terminal window on a Windows operating system. The terminal prompt is "yaswanth@3bb36d3d774953e:~". The user runs "nano for3.sh" to edit a script, which contains a for loop with a range from 10 down to 0 with an increment of 2. The output of the script is displayed in the terminal: "10", "8", "6", "4", "2", and "0". The taskbar at the bottom shows various application icons like File Explorer, Edge, and Firefox.

5. This is the basic example to print a series of numbers from 1 to 10.

```
yaswanth@3bb36d3d774953e:~$ nano for3.sh
yaswanth@3bb36d3d774953e:~$ nano for4.sh
yaswanth@3bb36d3d774953e:~$ chmod a+x for4.sh
yaswanth@3bb36d3d774953e:~$ ./for4.sh
1
2
3
4
5
6
7
8
9
10
yaswanth@3bb36d3d774953e:~$
```



Activate Windows
Go to Settings to activate Windows.

ENG 15:18
US 27-01-2025

```
yaswanth@3bb36d3d774953e:~$ GNU nano 7.2
^M num in {1..10..1}
^M done
^M
```



Activate Windows
Go to Settings to activate Windows.

Help Write Out Where Is Cut Paste Execute Justify Location Go To Line Undo Redo Set Mark To Bracket Copy Where Was

^D Help ^W Write Out ^F Where Is ^X Exit ^R Read File ^C Cut ^V Paste ^E Execute ^J Justify ^L Location ^G Go To Line ^U Undo ^R Redo ^A Set Mark ^T To Bracket ^C Copy ^W Where Was

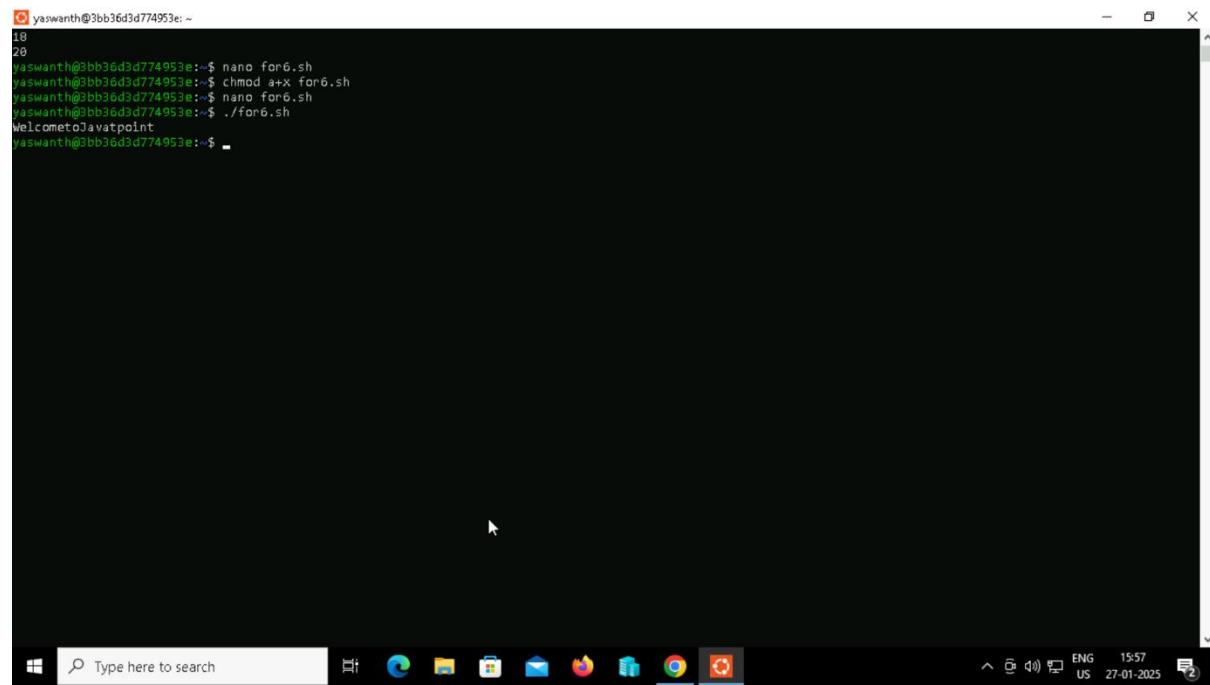
ENG 15:18
US 27-01-2025

6. For Loop with a Break Statement :

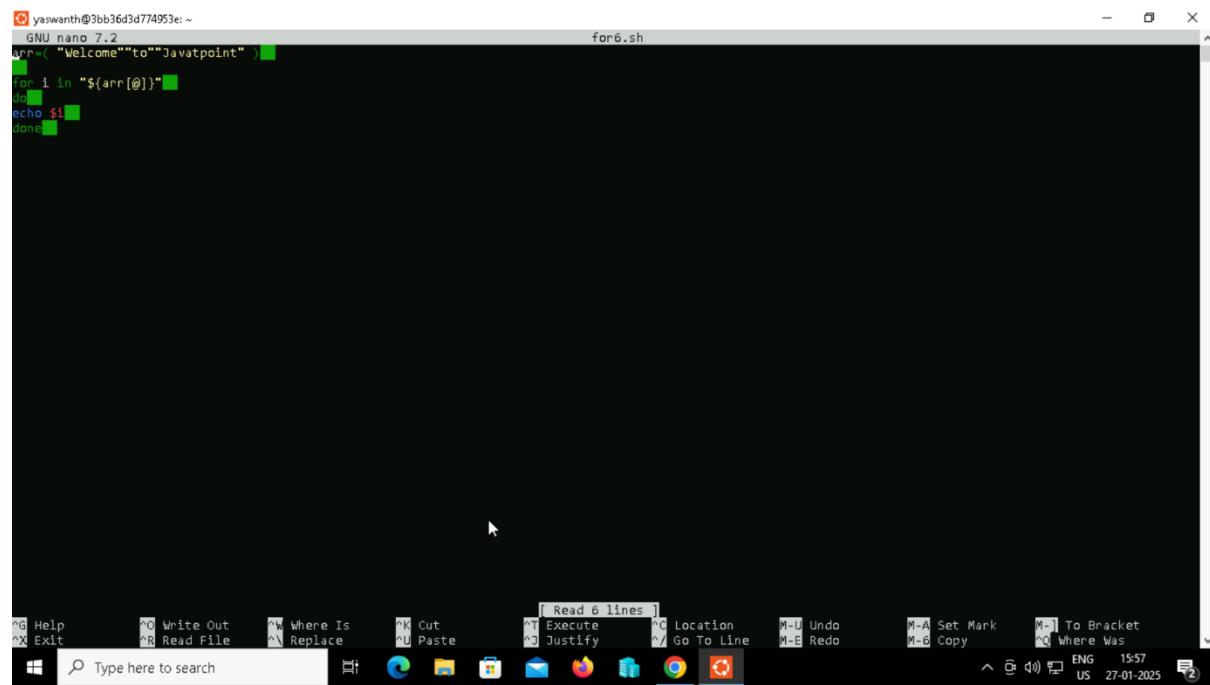
```
yaswanth@3bb36d3d774953e:~$ nano for5.sh
yaswanth@3bb36d3d774953e:~$ chmod a+x for5.sh
yaswanth@3bb36d3d774953e:~$ ./for5.sh
2
4
6
8
10
12
14
16
18
20
yaswanth@3bb36d3d774953e:~$ -
```

```
yaswanth@3bb36d3d774953e:~$ GNU nano 7.2
^_table in (2..100..2
do
echo $table
if [ $table == 20 ]; then
break
fi
done
```

7. For Loop to Read Array Variables :



```
yaswanth@3bb36d3d774953e:~$ nano for6.sh
18
20
yaswanth@3bb36d3d774953e:~$ chmod +x for6.sh
yaswanth@3bb36d3d774953e:~$ nano for6.sh
yaswanth@3bb36d3d774953e:~$ ./for6.sh
Welcome to Javatpoint
yaswanth@3bb36d3d774953e:~$
```



```
yaswanth@3bb36d3d774953e:~$ GNU nano 7.2
arr=( "Welcome" "to" "Javatpoint" )
for i in ${arr[@]}
do
echo $i
done
```

8 . Infinite Bash For Loop :

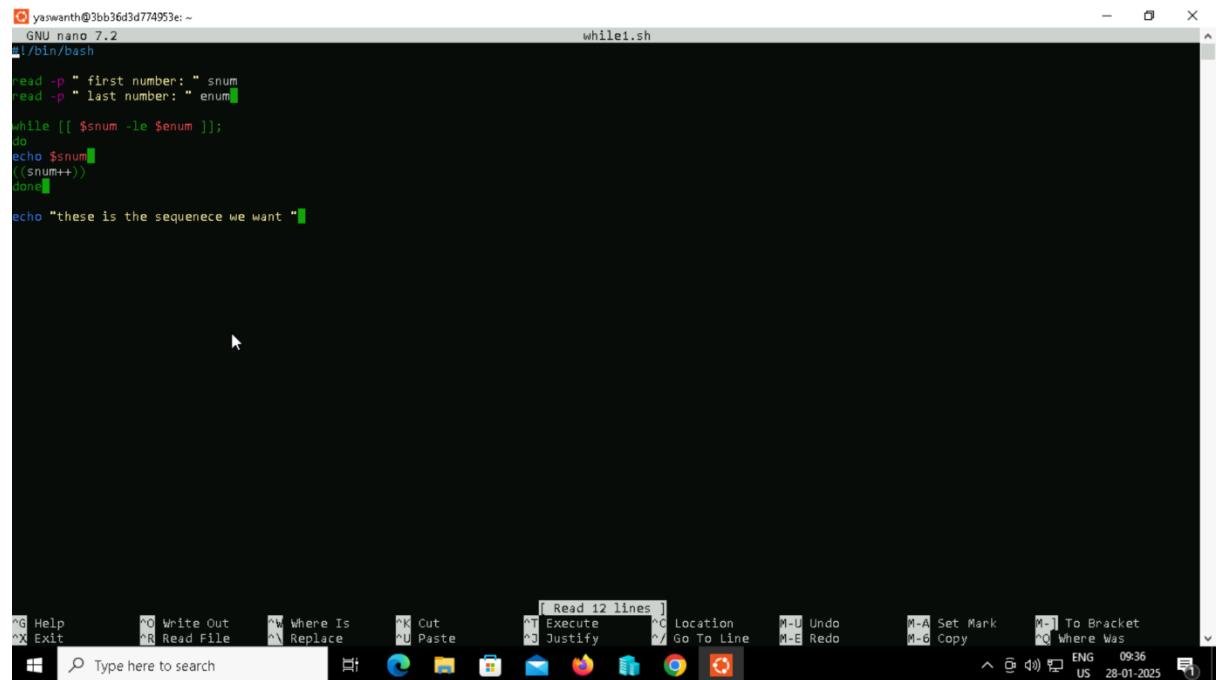
```
yaswanth@3bb36d3d774953e:~$ nano for10.sh
yaswanth@3bb36d3d774953e:~$ chmod a+x for10.sh
yaswanth@3bb36d3d774953e:~$ ./for10.sh
Current Number: 1
Current Number: 2
Current Number: 3
Current Number: 4
Current Number: 5
Current Number: 6
Current Number: 7
Current Number: 8
Current Number: 9
Current Number: 10
Current Number: 11
Current Number: 12
Current Number: 13
Current Number: 14
Current Number: 15
Current Number: 16
Current Number: 17
Current Number: 18
Current Number: 19
Current Number: 20
Current Number: 21
^Cyaswanth@3bb36d3d774953e:~$
```

```
yaswanth@3bb36d3d774953e:~$ nano for10.sh
GNU nano 7.2
1:1
for (( i=1; i<100; i++ ))
do
sleep 1s
echo "Current Number: $((i+1))"
done
^Cyaswanth@3bb36d3d774953e:~$
```

File Name to Write: for10.sh

Cancel Help M-D DOS Format M-A Append M-B Backup File
Type here to search M-M Mac Format M-P Prepend M-T Browse
Windows 16:06 ENG US 27-01-2025

9. while loop is used with a single condition in expression. It is the basic example of while loop which will print series of numbers as per user input:

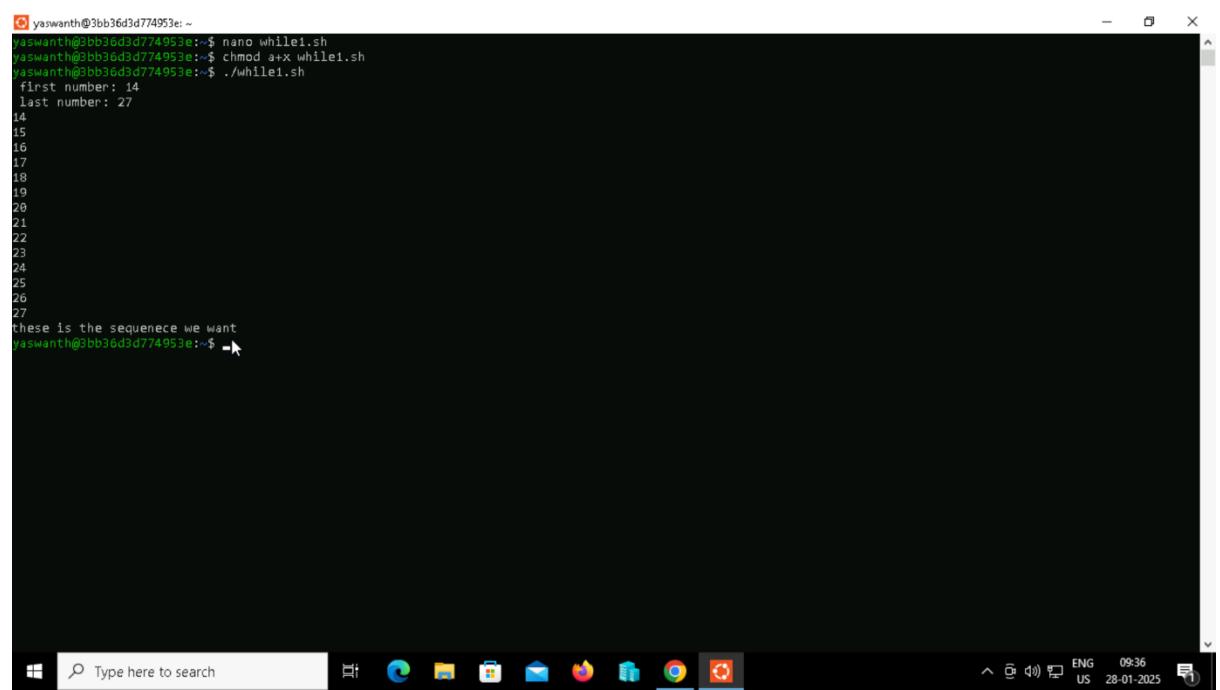


```
yaswanth@3bb36d3d774953e:~$ nano while1.sh
#!/bin/bash

read -p " first number: " snum
read -p " last number: " enum

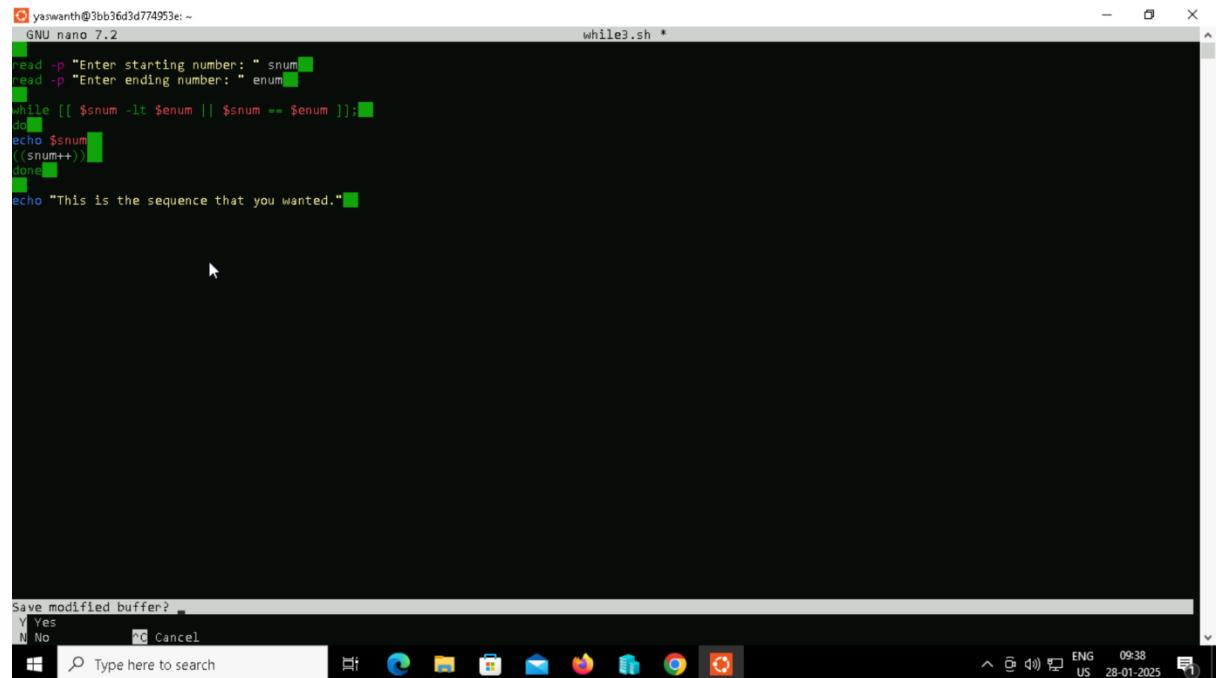
while [[ $snum -le $enum ]];
do
echo $snum
((snum++))
done

echo "these is the sequenec we want "
```

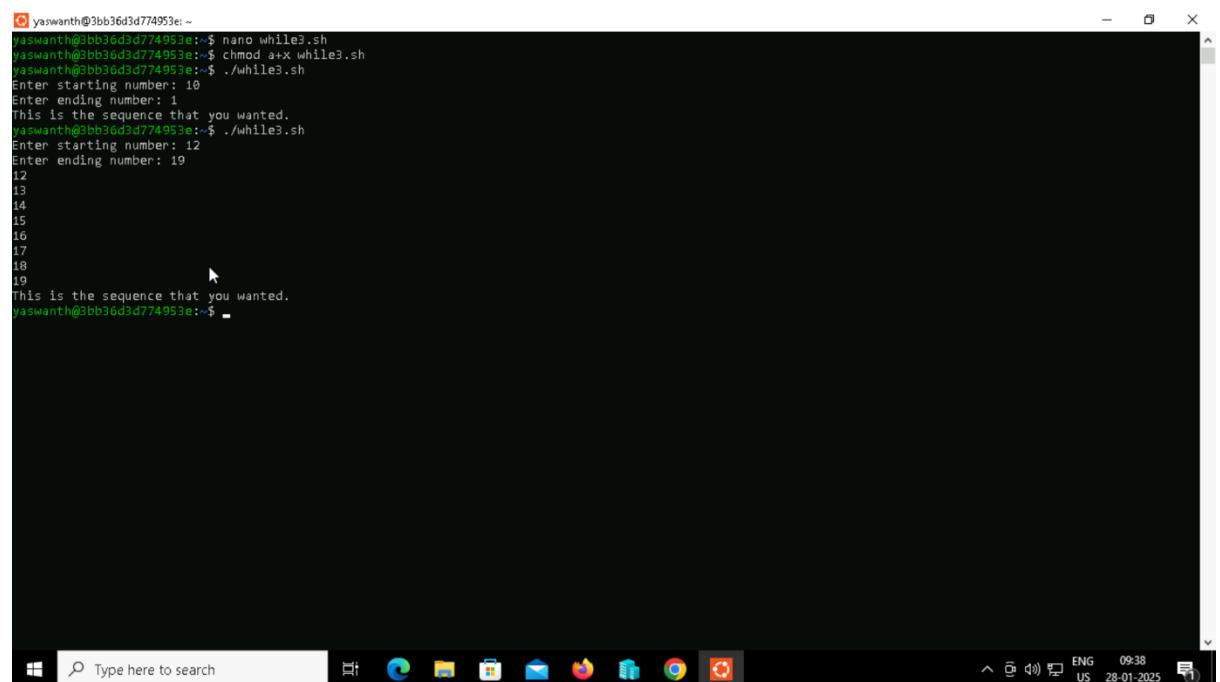


```
yaswanth@3bb36d3d774953e:~$ nano while1.sh
yaswanth@3bb36d3d774953e:~$ chmod a+x while1.sh
yaswanth@3bb36d3d774953e:~$ ./while1.sh
first number: 14
last number: 27
14
15
16
17
18
19
20
21
22
23
24
25
26
27
these is the sequenec we want
yaswanth@3bb36d3d774953e:~$
```

10.while loop with multiple conditions in the expression



```
yaswanth@3bb36d3d774953e: ~
GNU nano 7.2
while3.sh *
read -p "Enter starting number: " snum
read -p "Enter ending number: " enum
while [[ $snum -lt $enum || $snum == $enum ]]; do
echo $snum
((snum++))
done
echo "This is the sequence that you wanted."
Save modified buffer?
Y Yes
N No
C Cancel
```



```
yaswanth@3bb36d3d774953e: ~
yaswanth@3bb36d3d774953e: ~$ nano while3.sh
yaswanth@3bb36d3d774953e: ~$ chmod a+x while3.sh
yaswanth@3bb36d3d774953e: ~$ ./while3.sh
Enter starting number: 10
Enter ending number: 1
This is the sequence that you wanted.
yaswanth@3bb36d3d774953e: ~$ ./while3.sh
Enter starting number: 12
Enter ending number: 19
12
13
14
15
16
17
18
19
This is the sequence that you wanted.
yaswanth@3bb36d3d774953e: ~$
```

11.A break statement can be used to stop the loop as per the applied condition.



GNU nano 7.2

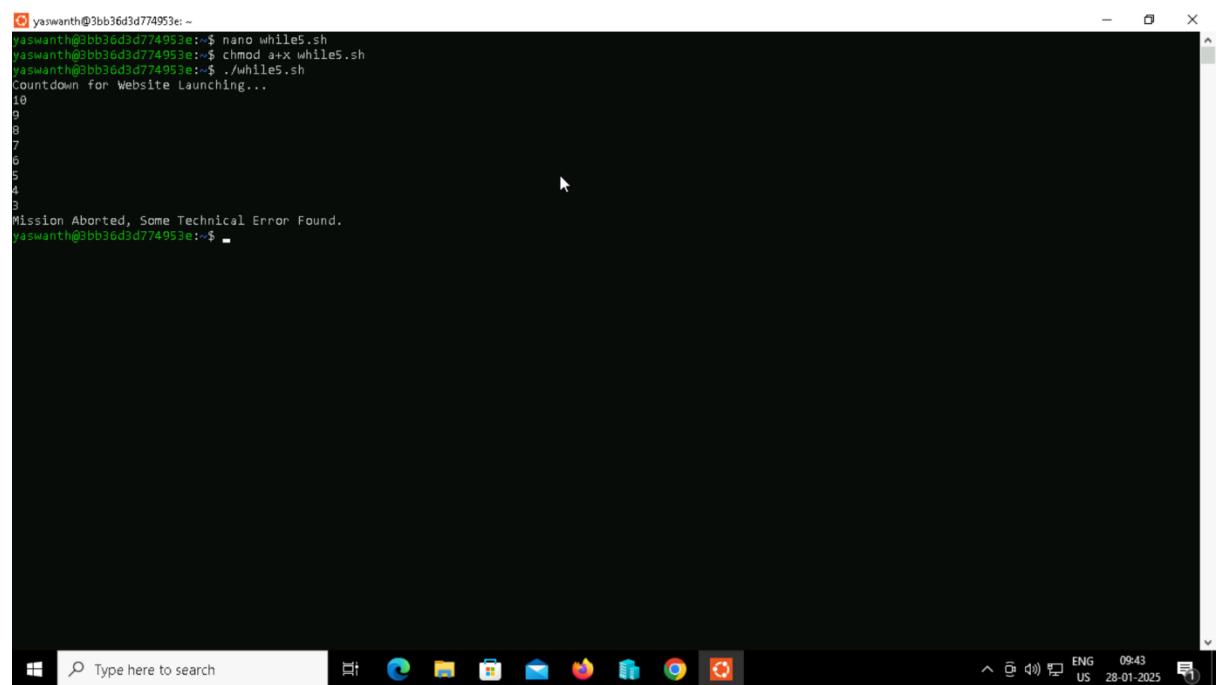
```
echo "Countdown for Website Launching..."  
i=10  
while [ $i -ge 1 ]  
do  
if [ $i == 2 ]  
then  
    echo "Mission Aborted, Some Technical Error Found."  
    break  
fi  
echo "$i"  
(( i-- ))  
done
```

File menu: Help, Exit, White Out, Read File, Where Is, Replace, Cut, Paste, Execute, Justify, Location, Undo, Redo, Set Mark, Copy, To Bracket, Where Was.

Type here to search

Windows taskbar: Search bar, Task View, File Explorer, Mail, Edge, Photos, OneDrive, Firefox, Chrome, Task View, Start button.

System tray: ENG 09:43, US 28-01-2025, Battery icon.



```
yaswanth@3bb36d3d774953e:~$ nano while5.sh  
yaswanth@3bb36d3d774953e:~$ chmod a+x while5.sh  
yaswanth@3bb36d3d774953e:~$ ./while5.sh  
Countdown for Website Launching...  
10  
9  
8  
7  
6  
5  
4  
3  
Mission Aborted, Some Technical Error Found.  
yaswanth@3bb36d3d774953e:~$
```

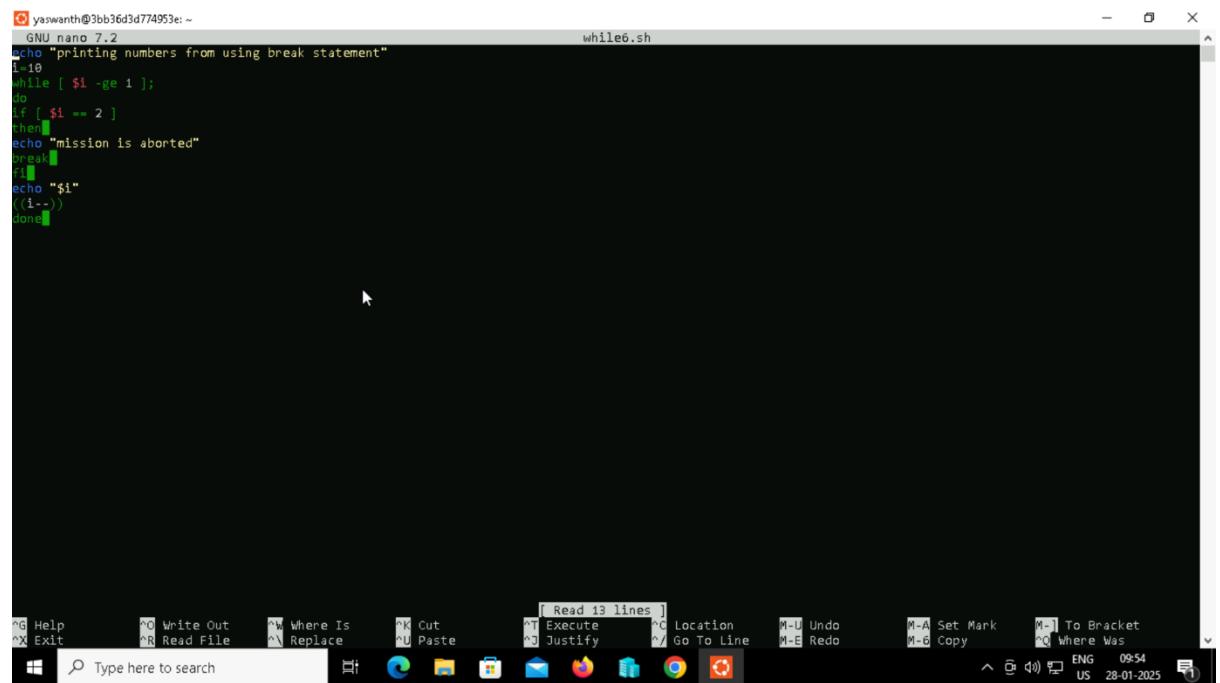
File menu: Help, Exit, White Out, Read File, Where Is, Replace, Cut, Paste, Execute, Justify, Location, Undo, Redo, Set Mark, Copy, To Bracket, Where Was.

Type here to search

Windows taskbar: Search bar, Task View, File Explorer, Mail, Edge, Photos, OneDrive, Firefox, Chrome, Task View, Start button.

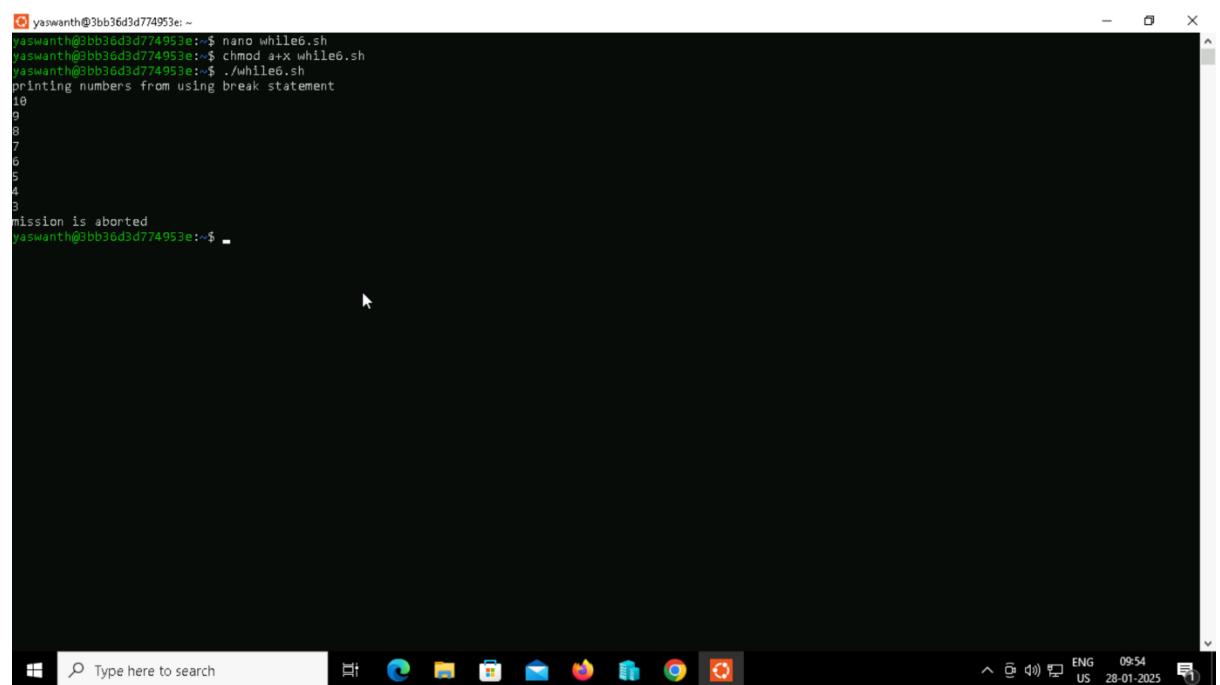
System tray: ENG 09:43, US 28-01-2025, Battery icon.

11.A break statement can be used to stop the loop as per the applied condition



GNU nano 7.2
echo "printing numbers from using break statement"
i=10
while [\$1 -ge 1];
do
if [\$1 == 2];
then
echo "mission is aborted"
break
fi
echo "\$i"
(i--)
done

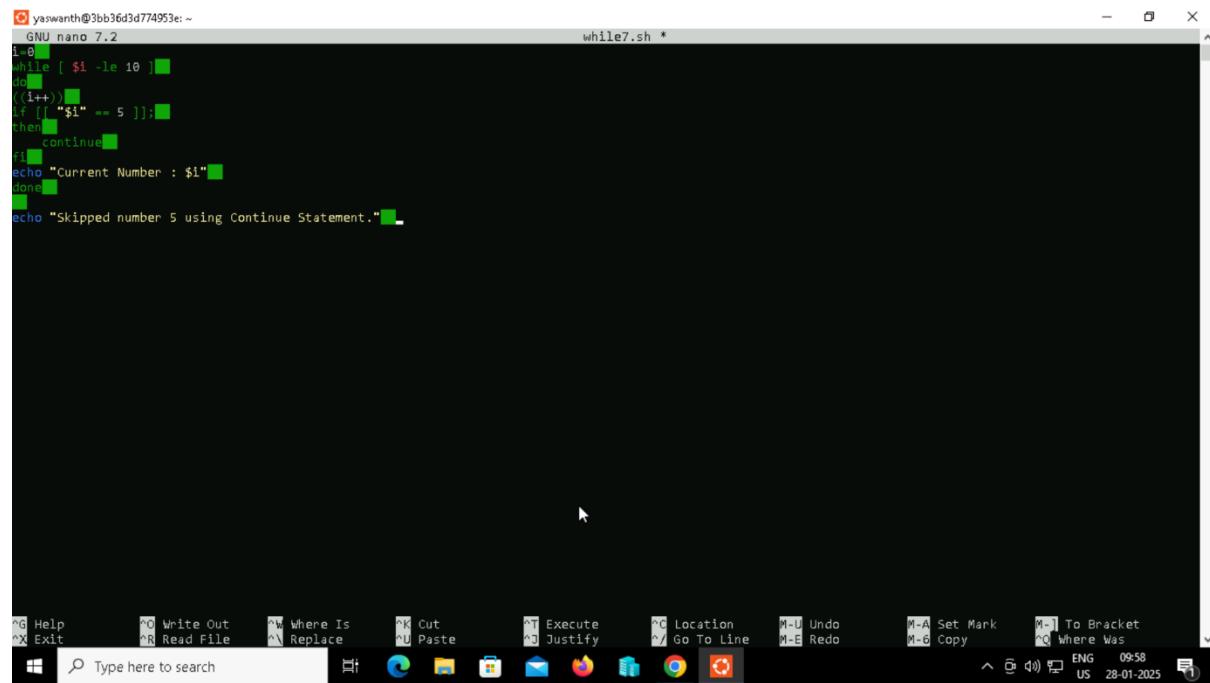
The screenshot shows a Windows desktop environment. In the center is a terminal window titled "while6.sh" containing a shell script. The script uses a while loop to print numbers from 10 down to 1. It includes a break statement to exit the loop when the value of \$1 is 2. The terminal window has a menu bar with options like Help, Exit, Write Out, Read File, Where Is, Replace, Cut, Paste, Execute, Location, Undo, Redo, Set Mark, To Bracket, and Where Was. Below the menu is a toolbar with icons for file operations. At the bottom of the terminal window is a status bar showing "Read 13 lines". The taskbar at the bottom of the screen shows various application icons, including Microsoft Edge, File Explorer, Mail, and others. The system tray on the right shows the date and time as 28-01-2025 09:54.



```
printing numbers from using break statement
10
9
8
7
6
5
4
3
mission is aborted
```

The screenshot shows a Windows desktop environment. In the center is a terminal window displaying the output of the shell script. The output shows the numbers 10 through 3 being printed, followed by the message "mission is aborted". The terminal window has a menu bar with options like Help, Exit, Write Out, Read File, Where Is, Replace, Cut, Paste, Execute, Location, Undo, Redo, Set Mark, To Bracket, and Where Was. Below the menu is a toolbar with icons for file operations. At the bottom of the terminal window is a status bar showing "Read 13 lines". The taskbar at the bottom of the screen shows various application icons, including Microsoft Edge, File Explorer, Mail, and others. The system tray on the right shows the date and time as 28-01-2025 09:54.

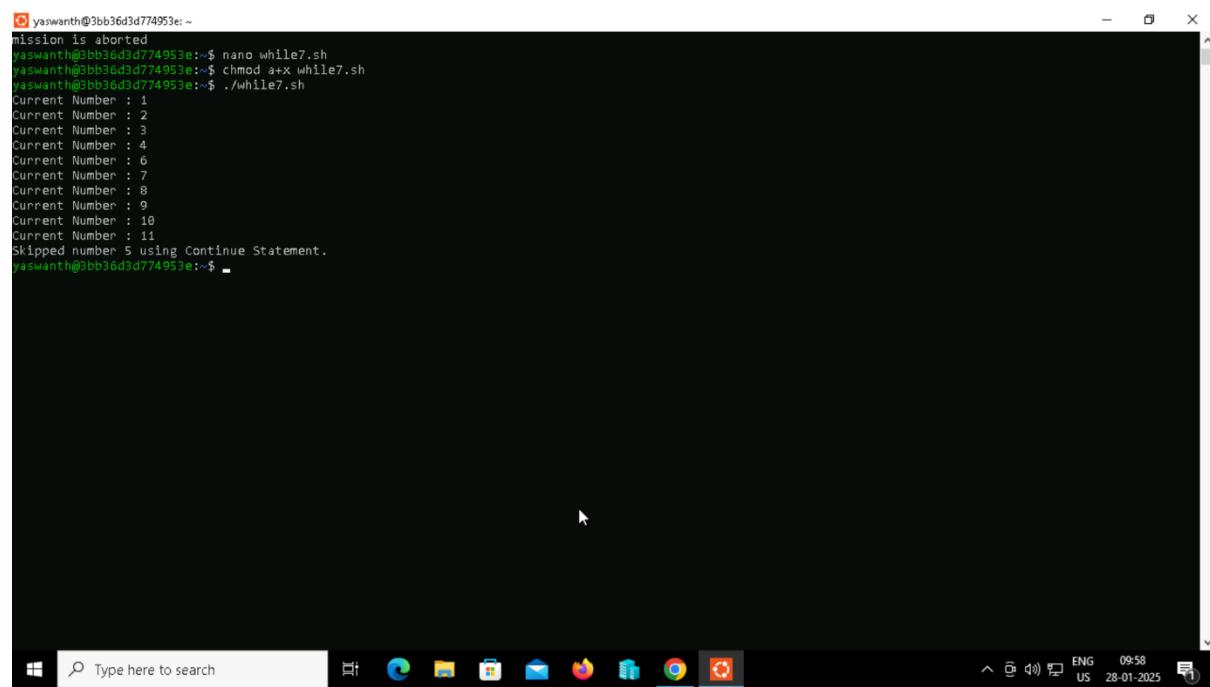
12.A continue statement can be used to skip the iteration for a specific condition inside the while loop.



yaswanth@3bb36d3d774953e:~

```
GNU nano 7.2
1.0
while [ $i -le 10 ]
do
((i++))
if [[ "$i" == 5 ]]; then
    continue
fi
echo "Current Number : $i"
done
echo "Skipped number 5 using Continue Statement."
```

The screenshot shows a Windows desktop environment. In the center is a terminal window titled "while7.sh *". It contains a shell script named "while7.sh" written in the GNU nano 7.2 editor. The script uses a while loop to iterate from 1 to 10. Inside the loop, it checks if the current value is 5. If it is, it uses the "continue" keyword to skip the rest of the loop body and move to the next iteration. Otherwise, it prints the current number. After the loop, it prints a message indicating that the number 5 was skipped using the continue statement. At the bottom of the terminal window, there is 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, To Bracket, Where Was, and a status bar showing ENG 09:58 US 28-01-2025. Below the terminal window is a taskbar with various icons for common applications like File Explorer, Edge, Mail, and Google Chrome. A search bar is also visible on the taskbar.



```
mission is aborted
yaswanth@3bb36d3d774953e:~$ nano while7.sh
yaswanth@3bb36d3d774953e:~$ chmod a+x while7.sh
yaswanth@3bb36d3d774953e:~$ ./while7.sh
Current Number : 1
Current Number : 2
Current Number : 3
Current Number : 4
Current Number : 6
Current Number : 7
Current Number : 8
Current Number : 9
Current Number : 10
Current Number : 11
Skipped number 5 using Continue Statement.
yaswanth@3bb36d3d774953e:~$
```

The screenshot shows a Windows desktop environment. In the center is a terminal window showing the execution of the "while7.sh" script. The output shows the script running from 1 to 11, skipping the number 5 due to the "continue" statement. At the top of the terminal window, there is a command history showing the commands entered: "mission is aborted", "nano while7.sh", "chmod a+x while7.sh", and "./while7.sh". Below the terminal window is a taskbar with various icons for common applications like File Explorer, Edge, Mail, and Google Chrome. A search bar is also visible on the taskbar.

13. the until loop contains a single condition in expression. It is the basic example of until loop which will print series of numbers from 1 to 10

Save modified buffer?

Y Yes N No C Cancel

Activate Windows
Go to Settings to activate Windows.

1.1
until [\$i -gt 10]
do
echo \$i
((i++))
done

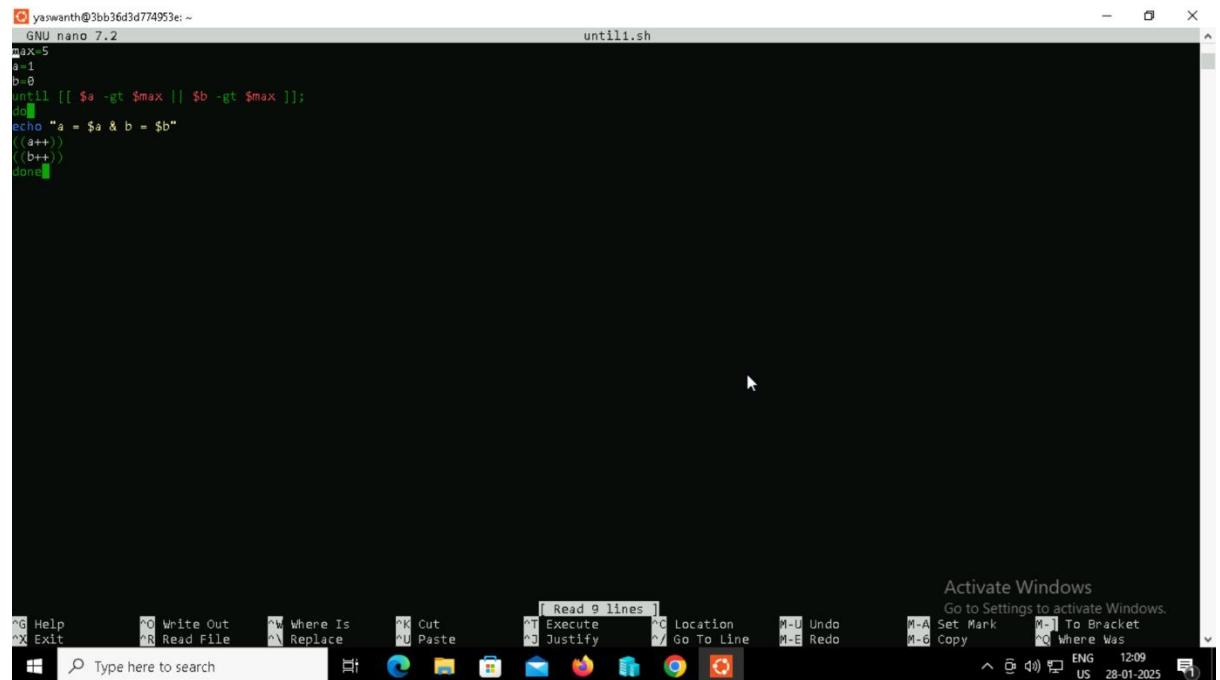
11:58 28-01-2025

Activate Windows
Go to Settings to activate Windows.

1
2
3
4
5
6
7
8
9
10

11:58 28-01-2025

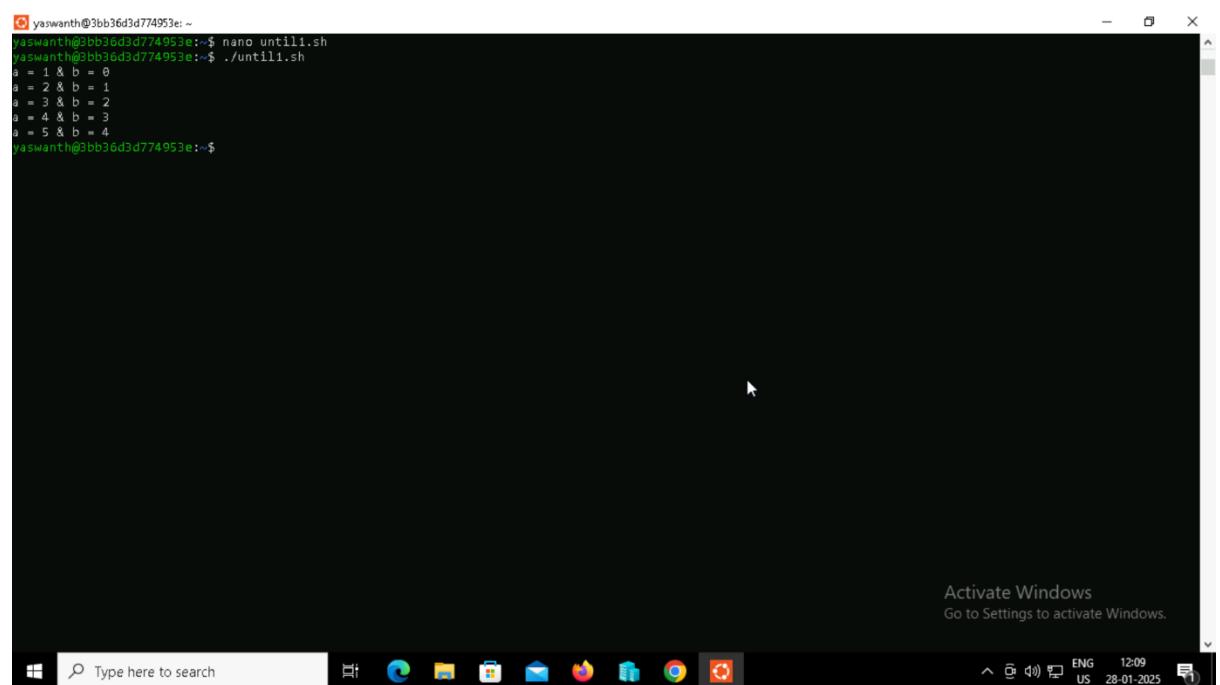
14. with multiple conditions in an expression



Terminal window showing the nano editor with the script content:

```
max=5
a=1
b=0
until [[ $a -gt $max || $b -gt $max ]];
do
echo "a = $a & b = $b"
((a++))
((b++))
done
```

The terminal window has a title bar "until1.sh". The menu bar includes Help, Exit, Write Out, Read File, Where Is, Replace, Cut, Paste, Execute, Justify, Location, Go To Line, Undo, Redo, Set Mark, Copy, and Where Was. The status bar shows "Activate Windows", "Go to Settings to activate Windows.", "M-A Set Mark", "M-T To Bracket", "M-B Copy", "M-C Where Was", "ENG 12:09 US 28-01-2025". The taskbar at the bottom shows icons for File Explorer, Edge, File, Mail, Firefox, Google Chrome, and Task View.



Terminal window showing the output of the script execution:

```
max=5
a=1 & b=0
a=2 & b=1
a=3 & b=2
a=4 & b=3
a=5 & b=4
```

The terminal window has a title bar "until1.sh". The menu bar includes Help, Exit, Write Out, Read File, Where Is, Replace, Cut, Paste, Execute, Justify, Location, Go To Line, Undo, Redo, Set Mark, Copy, and Where Was. The status bar shows "Activate Windows", "Go to Settings to activate Windows.", "M-A Set Mark", "M-T To Bracket", "M-B Copy", "M-C Where Was", "ENG 12:09 US 28-01-2025". The taskbar at the bottom shows icons for File Explorer, Edge, File, Mail, Firefox, Google Chrome, and Task View.

15. An equal operator (=) is used to check whether two strings are equal.

GNU nano 7.2

```
str1="Welcometojavatpoint."
str2="javatpoint"
if [ $str1 = $str2 ]; then
echo "Both the strings are equal."
else
echo "Strings are not equal."
fi
```

string1.sh *

Activate Windows
Save modified buffer? Yes No Cancel

Type here to search

14:08 28-01-2025 ENG US

```
a = 5 & b = 4
yaswanth@3bb36d3d774953e:~$ nano untili.sh
yaswanth@3bb36d3d774953e:~$ nano string1.sh
yaswanth@3bb36d3d774953e:~$ chmod a+x string1.sh
yaswanth@3bb36d3d774953e:~$ ./string1.sh
Strings are not equal.
yaswanth@3bb36d3d774953e:~$
```

Activate Windows
Go to Settings to activate Windows.

Type here to search

14:08 28-01-2025 ENG US

16. Not equal operator (!=) is used to define that strings are not equal.

```
yaswanth@3bb36d3d774953e:~$ nano string2.sh
yaswanth@3bb36d3d774953e:~$ chmod a+x string2.sh
yaswanth@3bb36d3d774953e:~$ nano string3.sh
yaswanth@3bb36d3d774953e:~$ ./string2.sh
Strings are not equal.
yaswanth@3bb36d3d774953e:~$
```

The screenshot shows a Windows terminal window with a black background. It displays a command-line session where a user named 'yaswanth' is comparing two strings. The user runs 'nano string2.sh' to edit a script, adds 'chmod a+x string2.sh' to make it executable, edits 'string3.sh', and then runs the script with './string2.sh'. The output shows that the strings are not equal. The taskbar at the bottom includes icons for File Explorer, Edge, File Manager, Mail, and others. A system tray icon for a battery is visible on the right.

```
yaswanth@3bb36d3d774953e:~$ GNU nano 7.2
GNU nano 7.2                                         string2.sh *
str1="WelcometoJavatpoint."
str2="javatpoint"
if [[ $str1 != $str2 ]]; then
echo "Strings are not equal."
else
echo "Strings are equal."
fi
```

The screenshot shows a Windows terminal window with a black background. It displays a command-line session where a user named 'yaswanth' is editing a script named 'string2.sh' using 'nano 7.2'. The script contains a comparison between two strings: 'str1' and 'str2'. If they are not equal, it prints 'Strings are not equal.'; otherwise, it prints 'Strings are equal.'. The taskbar at the bottom includes icons for File Explorer, Edge, File Manager, Mail, and others. A system tray icon for a battery is visible on the right. A modal dialog box is open in the foreground asking 'Save modified buffer?' with options 'Yes', 'No', and 'Cancel'.

17. The 'greater than operator (`\>`)' is used to check if string1 is greater than string2.

```
GNU nano 7.2                                     string3.sh *
stri="WelcometoJavatpoint"
str2="Javatpoint"
if [ $stri \> $str2 ]; then
then
    echo "$stri is greater then $str2"
else
    echo "$stri is less then $str2"
fi
```

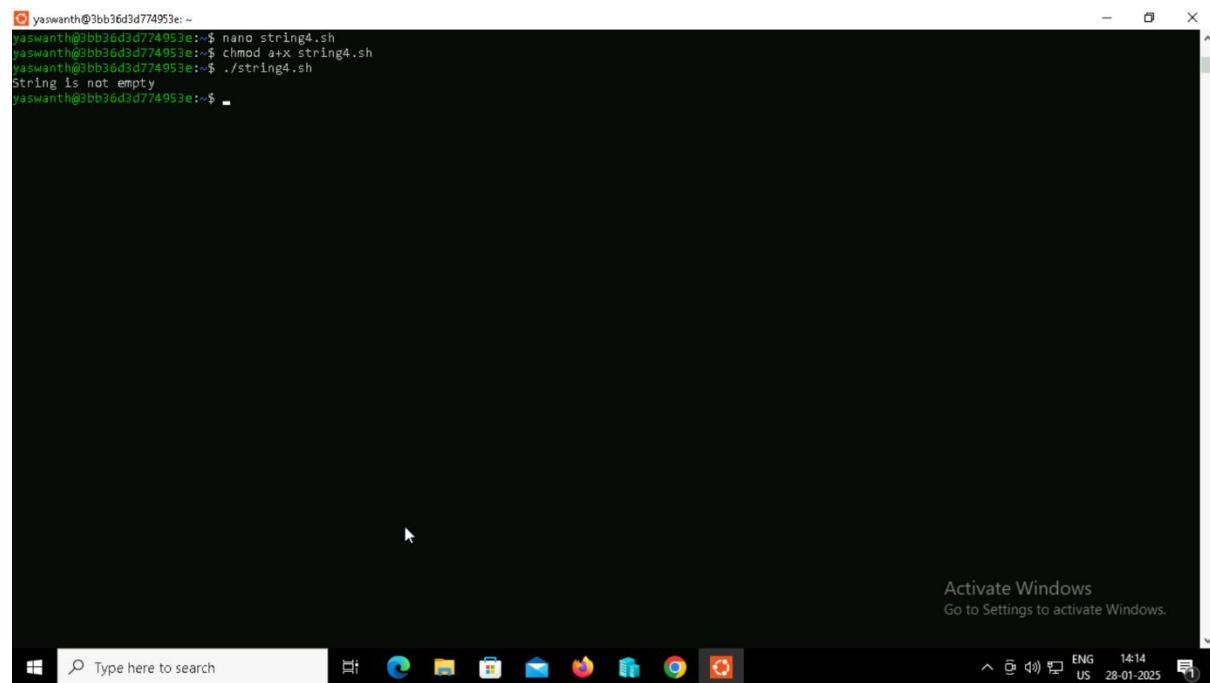
The screenshot shows a terminal window titled "string3.sh". It contains a shell script with two strings, "stri" and "str2", defined. An "if" condition checks if "stri" is greater than "str2". If true, it prints "\$stri is greater then \$str2"; otherwise, it prints "\$stri is less then \$str2". The terminal window has a standard Windows-style title bar and a taskbar at the bottom.

```
yaswanth@3bb36d3d774953e:~$ ./string3.sh
Strings are not equal.
yaswanth@3bb36d3d774953e:~$ nano string3.sh
yaswanth@3bb36d3d774953e:~$ chmod a+x string3.sh
yaswanth@3bb36d3d774953e:~$ ./string3.sh
WelcometoJavatpoint is greater then Javatpoint
yaswanth@3bb36d3d774953e:~$
```

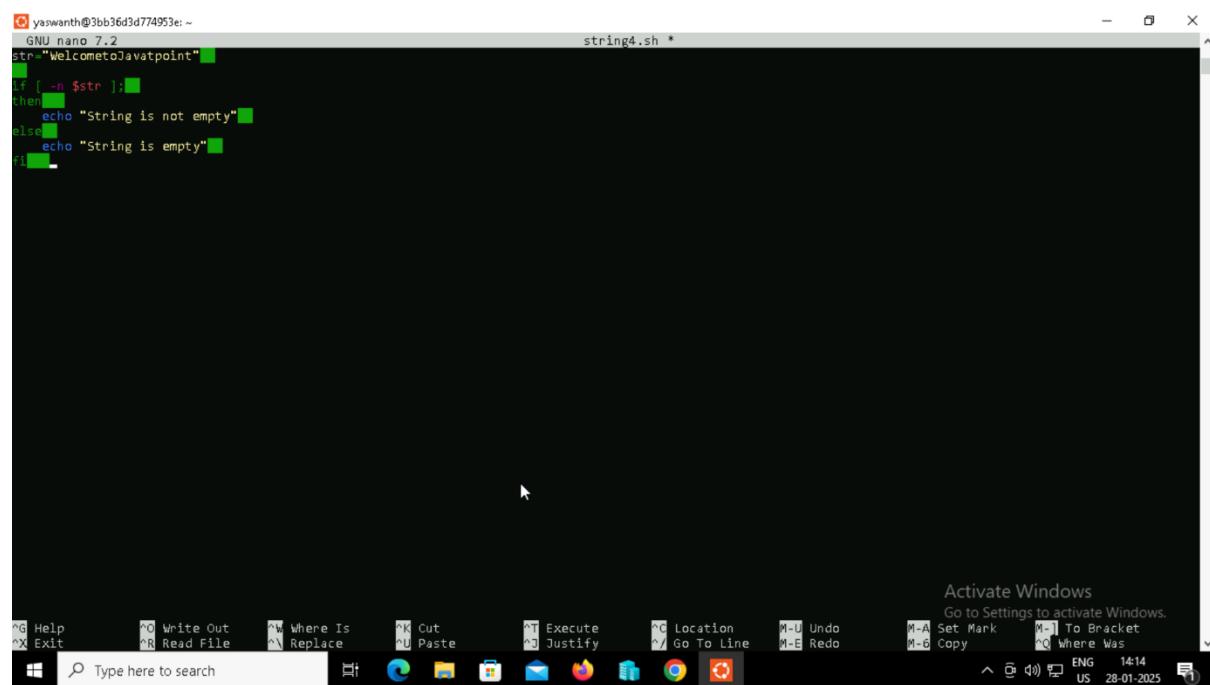
The screenshot shows a terminal window executing the script. It first runs the command `./string3.sh`, which outputs "Strings are not equal.". Then, it edits the script with `nano string3.sh`, changes its mode to executable with `chmod a+x string3.sh`, and finally runs it again, this time outputting "WelcometoJavatpoint is greater then Javatpoint". The terminal window has a standard Windows-style title bar and a taskbar at the bottom.

18. This operator is used to check if the string is zero or greater than zero

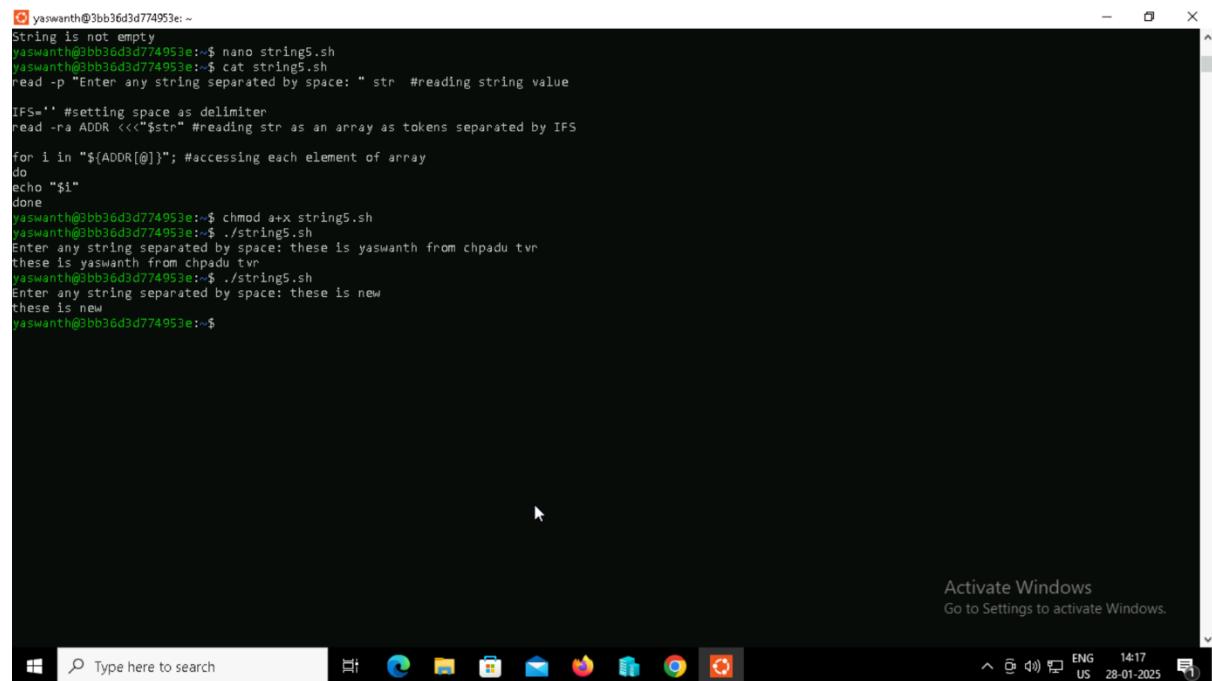
```
yaswanth@3bb36d3d774953e:~$ nano string4.sh
yaswanth@3bb36d3d774953e:~$ chmod a+x string4.sh
yaswanth@3bb36d3d774953e:~$ ./string4.sh
String is not empty
yaswanth@3bb36d3d774953e:~$
```



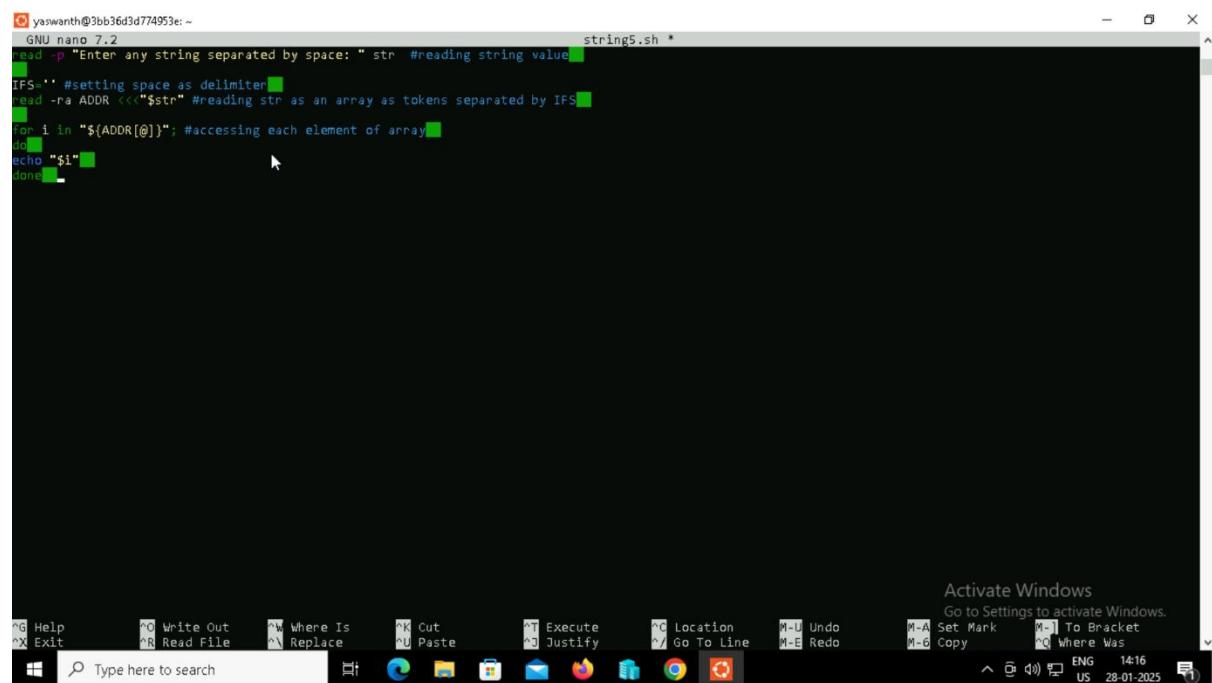
```
yaswanth@3bb36d3d774953e:~$ GNU nano 7.2
str="WelcometoJavatpoint"
if [ -n $str ]; then
echo "String is not empty"
else
echo "String is empty"
fi
```



19. This example, a string is **split** using a space character delimiter.



```
yaswanth@0bb36d3d774953e:~ String is not empty
yaswanth@0bb36d3d774953e:~$ nano string5.sh
yaswanth@0bb36d3d774953e:~$ cat string5.sh
read -p "Enter any string separated by space: " str #reading string value
IFS=' ' #setting space as delimiter
read -ra ADDR <<"$str" #reading str as an array as tokens separated by IFS
for i in "${ADDR[@]}"; #accessing each element of array
do
echo "$i"
done
yaswanth@0bb36d3d774953e:~$ chmod a+x string5.sh
yaswanth@0bb36d3d774953e:~$ ./string5.sh
Enter any string separated by space: these is yaswanth from chpadu tvr
these is yaswanth from chpadu tvr
yaswanth@0bb36d3d774953e:~$ ./string5.sh
Enter any string separated by space: these is new
these is new
yaswanth@0bb36d3d774953e:~$
```



```
yaswanth@0bb36d3d774953e:~ string5.sh *
GNU nano 7.2
read -p "Enter any string separated by space: " str #reading string value
IFS=' ' #setting space as delimiter
read -ra ADDR <<"$str" #reading str as an array as tokens separated by IFS
for i in "${ADDR[@]}"; #accessing each element of array
do
echo "$i"
done
```

20 . This example, a string is **split** using a comma (,) symbol character as a delimiter.

```
yawanth@3bb36d3d774953e:~\nthese is new\nyawanth@3bb36d3d774953e:~$ nano string6.sh\nyawanth@3bb36d3d774953e:~$ chmod a+x string6.sh\nyawanth@3bb36d3d774953e:~$ ./string6.sh\nEnter Name, State and Age separated by a comma: yash,ap,22\nName : yash\nState : ap\nAge : 22\nyawanth@3bb36d3d774953e:~$ cat string6.sh\n#!/bin/bash\n#Example for bash split string by Symbol (comma)\n\nread -p "Enter Name, State and Age separated by a comma: " entry #reading string value\n\nIFS=',' #setting comma as delimiter\nread -a strarr <<< "$entry" #reading str as an array as tokens separated by IFS\n\necho "Name : ${strarr[0]}"\necho "State : ${strarr[1]}"\necho "Age : ${strarr[2]}"\nyawanth@3bb36d3d774953e:~$
```

The screenshot shows a Windows terminal window with a black background. It displays the execution of a Bash script named 'string6.sh'. The script prompts the user to enter 'Name, State and Age separated by a comma', reads the input ('yash,ap,22'), and then prints each token on a new line. The terminal window has a title bar, a scroll bar on the right, and a taskbar at the bottom with various icons like File Explorer, Edge, and Task View.

```
yawanth@3bb36d3d774953e:~\n  GNU nano 7.2\n#!/bin/bash\n#Example for bash split string by Symbol (comma)\n\nread -p "Enter Name, State and Age separated by a comma: " entry #reading string value\n\nIFS=',' #setting comma as delimiter\nread -a strarr <<< "$entry" #reading str as an array as tokens separated by IFS\n\necho "Name : ${strarr[0]}"\necho "State : ${strarr[1]}"\necho "Age : ${strarr[2]}"\nyawanth@3bb36d3d774953e:~$
```

This screenshot shows the same Windows terminal window, but now it's displaying the contents of the 'string6.sh' file using the nano text editor. The file contains the same script code as the previous screenshot. A save dialog box is visible at the top, asking 'Save modified buffer?'. The taskbar at the bottom remains the same.

21. To calculate the length of a string is to use '#' symbol. In this example, we have used \${#string_variable_name} to find the length of a string.

```
yaswanth@3bb36d3d774953e:~$ nano split1.sh
yaswanth@3bb36d3d774953e:~$ chmod a+x split1.sh
yaswanth@3bb36d3d774953e:~$ ./split1.sh
Length of 'Welcome to Javatpoint' is 21
yaswanth@3bb36d3d774953e:~$
```

The screenshot shows a Windows terminal window with a black background. At the top, there's a title bar with the command prompt "yaswanth@3bb36d3d774953e:~\$". Below the title bar, the terminal displays a series of commands: "nano split1.sh", "chmod a+x split1.sh", "./split1.sh", and finally the output "Length of 'Welcome to Javatpoint' is 21". The bottom of the window shows the Windows taskbar with various icons like File Explorer, Edge, and Google Chrome. On the right side of the taskbar, there's a system tray with icons for battery, signal, and date/time (28-01-2025). A watermark for "Activate Windows" is visible in the center of the screen.

```
yaswanth@3bb36d3d774953e:~$ GNU nano 7.2
str="Welcome to Javatpoint"
length=${#str}
echo "Length of '$str' is $length"
```

The screenshot shows a Linux terminal window with a black background. At the top, there's a title bar with the command prompt "yaswanth@3bb36d3d774953e:~\$". Below the title bar, the terminal displays a series of commands: "GNU nano 7.2", "str='Welcome to Javatpoint'", "length=\${#str}", and "echo 'Length of '\$str' is \$length''. The bottom of the window shows a menu bar with options like Help, Exit, White Out, Read File, Where Is, Replace, Cut, Paste, Execute, Justify, Location, Go To Line, Undo, Redo, Set Mark, To Bracket, Copy, and Where Was. Below the menu bar is the terminal's search bar with the placeholder "Type here to search". The bottom of the window shows a standard Linux desktop interface with icons for File Explorer, Terminal, and other applications. A watermark for "Activate Windows" is visible in the center of the screen.

22. to calculate the length of a string is to use `expr` command with the 'length' keyword. In this example, we have used `expr length "\$str"` to find the length of a string.

```
yaswanth@3bb36d3d774953e:~$ nano split2.sh
Length of 'Welcome to Javatpoint' is 21
yaswanth@3bb36d3d774953e:~$ chmod a+x split2.sh
yaswanth@3bb36d3d774953e:~$ ./split2.sh
Length of 'Welcome to Javatpoint' is 21
yaswanth@3bb36d3d774953e:~$ cat split2.sh
str="Welcome to Javatpoint"
length=`expr "$str" : '\.*'
```

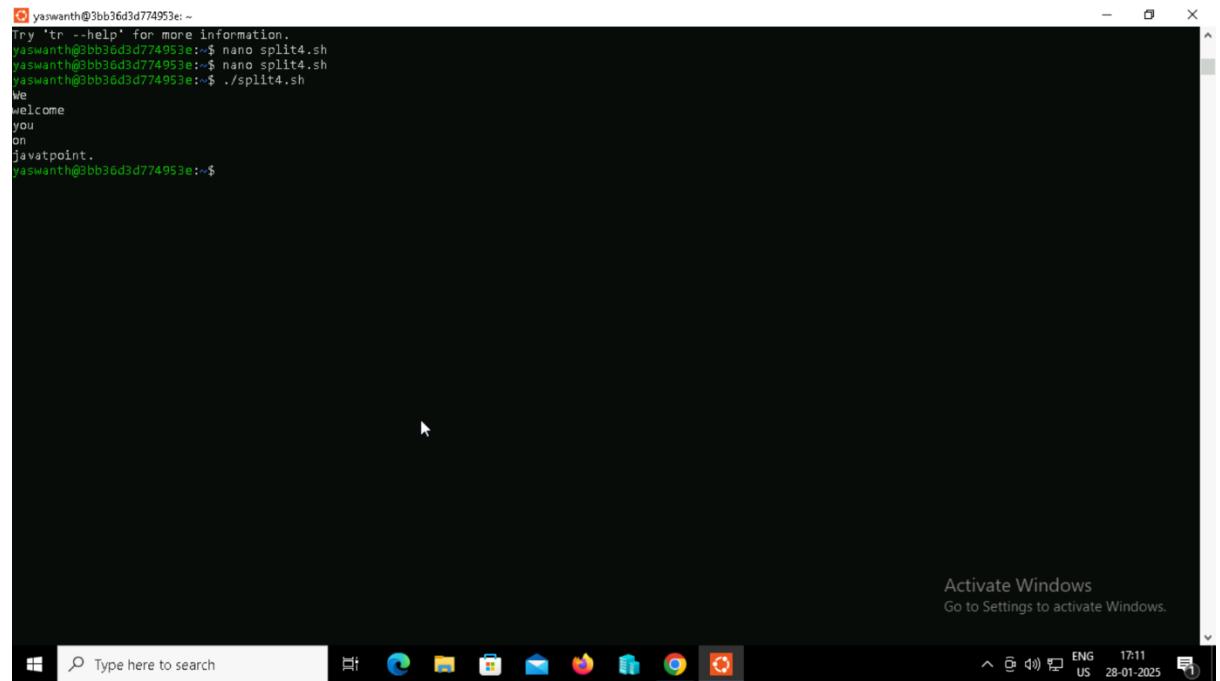
```
yaswanth@3bb36d3d774953e:~$ nano 7.2
GNU nano 7.2
str="Welcome to Javatpoint"
length=`expr "$str" : '\.*'
echo "Length of '$str' is $length"
```

23. we have used `awk` command to find the length of a string.

```
yaswanth@3bb36d3d774953e:~ echo "Length of '$str' is $length" yaswanth@3bb36d3d774953e:~$ nano split3.sh yaswanth@3bb36d3d774953e:~$ chmod a+x split3.sh yaswanth@3bb36d3d774953e:~$ ./split3.sh Length of 'Welcome to Javatpoint' is 21 Length of 'Welcome to Javatpoint' is 21 yaswanth@3bb36d3d774953e:~$ cat split3.sh str="Welcome to Javatpoint" length=`echo $str | awk '{print length}'` echo "Length of '$str' is $length" str="Welcome to Javatpoint" length=`echo $str | awk '{print length}'` echo "Length of '$str' is $length" yaswanth@3bb36d3d774953e:~
```

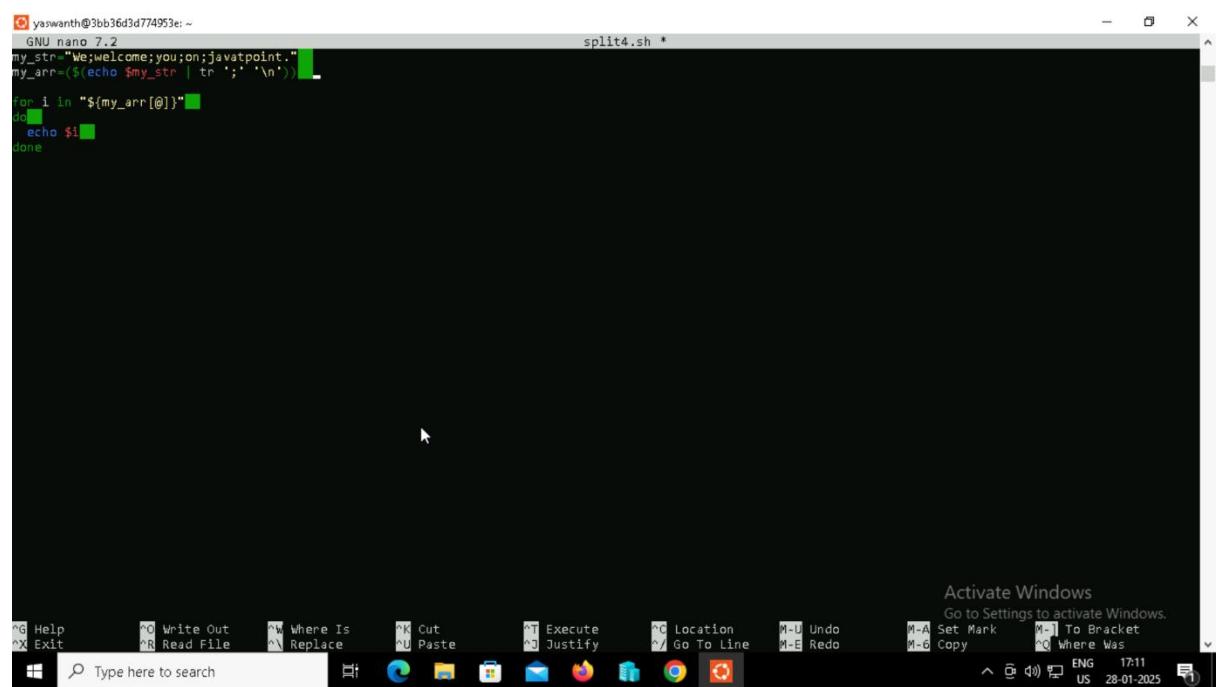
```
yaswanth@3bb36d3d774953e:~ GNU nano 7.2 split3.sh * str="Welcome to Javatpoint" length=`echo $str | awk '{print length}'` echo "Length of '$str' is $length" str="Welcome to Javatpoint" length=`echo $str | awk '{print length}'` echo "Length of '$str' is $length"
```

24. Bash **Split** String using Trim Command



```
yaswanth@3bb36d3d774953e:~  
Try 'tr --help' for more information.  
yaswanth@3bb36d3d774953e:~$ nano split4.sh  
yaswanth@3bb36d3d774953e:~$ nano split4.sh  
we  
welcome  
you  
on  
javatpoint.  
yaswanth@3bb36d3d774953e:~$
```

The screenshot shows a Windows terminal window with a black background and white text. It displays the command 'tr' being used to trim whitespace from the string 'we welcome you on javatpoint.'. The output shows the original string followed by a prompt '\$'. The terminal window has a title bar, a scroll bar on the right, and a taskbar at the bottom with various icons like File Explorer, Edge, and Google Chrome.

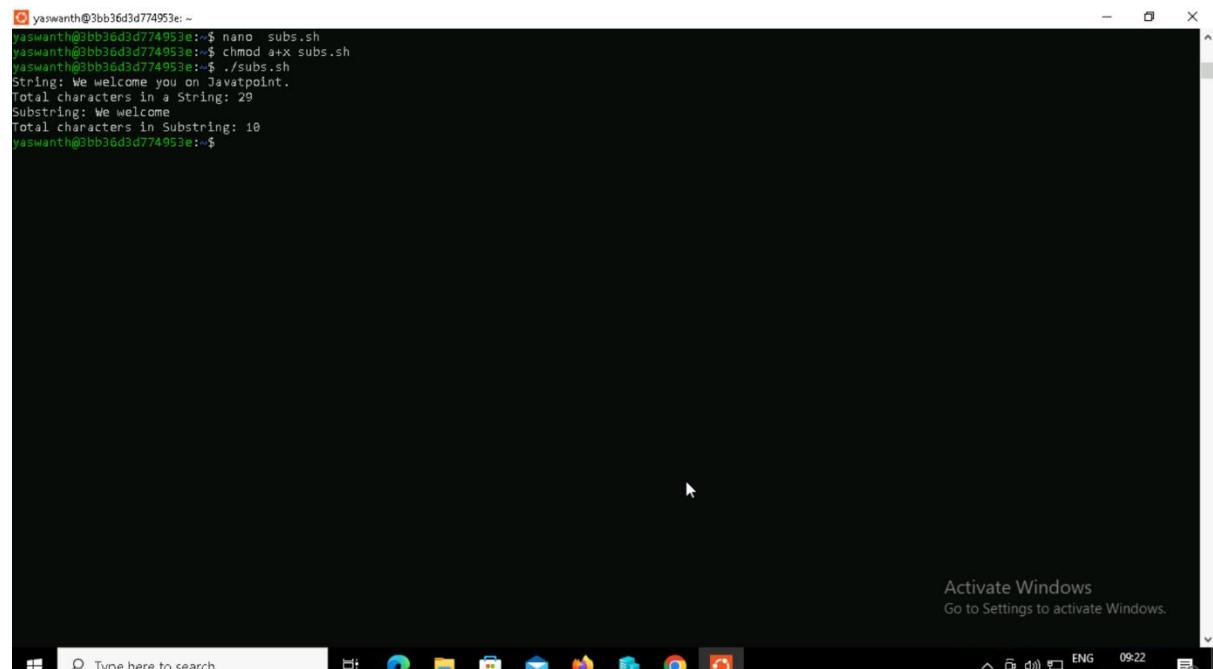


```
yaswanth@3bb36d3d774953e:~  
GNU nano 7.2  
my_str="we;welcome;you;on;javatpoint."  
my_arr=($(echo $my_str | tr ';' '\n'))  
for i in "${my_arr[@]}"  
do  
    echo $i  
done
```

```
split4.sh *
```

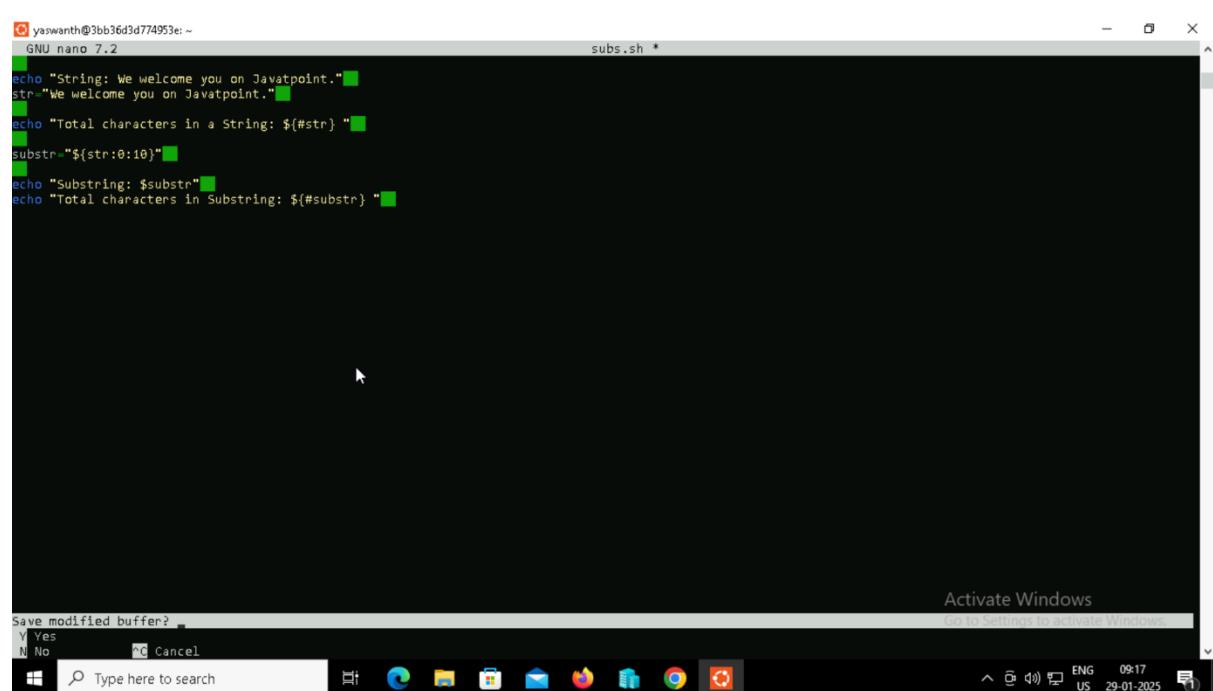
The screenshot shows a Windows terminal window with a black background and white text. It displays the execution of a script named 'split4.sh'. The script uses the 'tr' command to replace semicolons with newlines, then splits the resulting string into an array 'my_arr'. A 'for' loop iterates over the array, printing each element. The terminal window has a title bar, a scroll bar on the right, and a taskbar at the bottom with various icons like File Explorer, Edge, and Google Chrome.

25. To Extract till Specific Characters from Starting :**SUB STRING**



```
yaswanth@3bb36d3d774953e:~$ nano subs.sh
yaswanth@3bb36d3d774953e:~$ chmod a+x subs.sh
yaswanth@3bb36d3d774953e:~$ ./subs.sh
String: We welcome you on Javatpoint.
Total characters in a String: 29
Substring: We welcome
Total characters in Substring: 10
yaswanth@3bb36d3d774953e:~$
```

Activate Windows
Go to Settings to activate Windows.



```
yaswanth@3bb36d3d774953e:~$ GNU nano 7.2
echo "String: We welcome you on Javatpoint."
str="We welcome you on Javatpoint."
echo "Total characters in a String: ${#str} "
substr="\${str:0:10}"
echo "Substring: $substr"
echo "Total characters in Substring: ${#substr} "
subs.sh *
```

Save modified buffer?

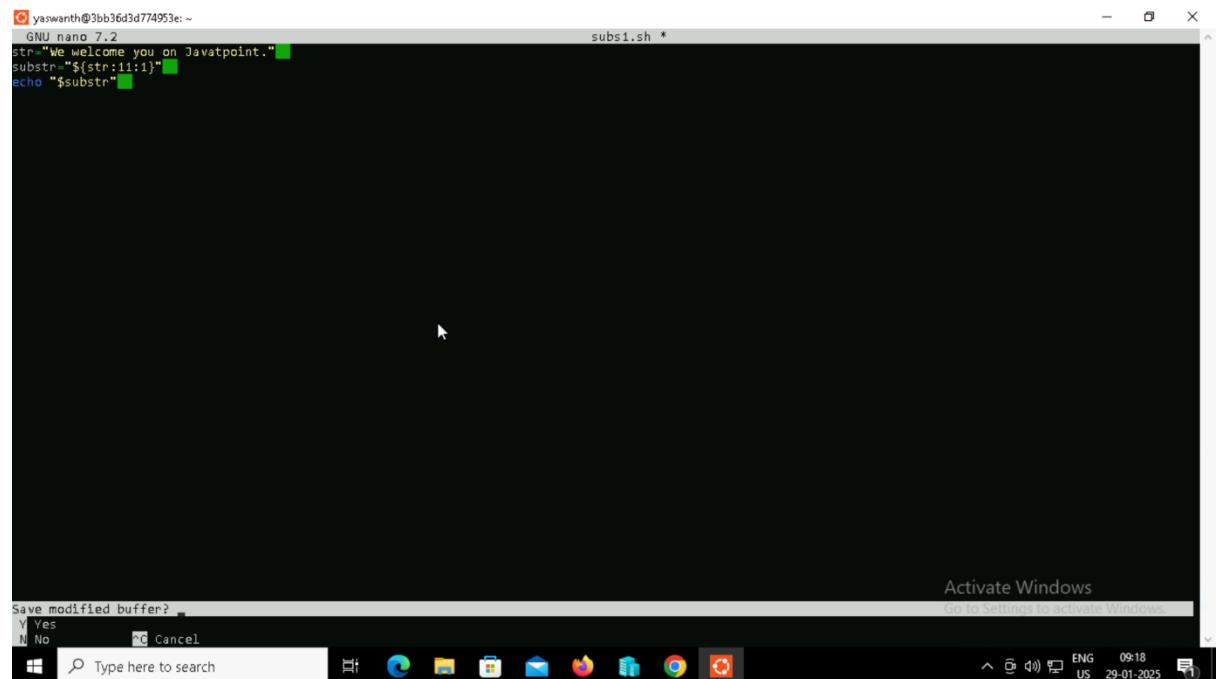
Y Yes N No C Cancel

Activate Windows
Go to Settings to activate Windows.

Type here to search Taskbar icons ENG US 09:22 29-01-2025

26 . To Extract a Single Character

```
aswanth@3bb36d3d774953e:~$ nano subs1.sh
aswanth@3bb36d3d774953e:~$ chmod a+x subs1.sh
aswanth@3bb36d3d774953e:~$ ./subs1.sh
aswanth@3bb36d3d774953e:~$ cat subs1.sh
str="We welcome you on Javatpoint."
substr=${str:11:1}
echo "$substr"
aswanth@3bb36d3d774953e:~$
```



The screenshot shows a Windows terminal window with a black background. At the top, it displays the command prompt: `aswanth@3bb36d3d774953e:~$`. Below the prompt, the terminal shows the contents of the `subs1.sh` file, which contains a single line of code: `echo "$substr"`. After the file content, there is a small asterisk (*) indicating the file has been modified. At the bottom of the terminal, a save dialog box is open, asking "Save modified buffer?". It contains three options: "Y Yes", "N No", and "C Cancel". The "Y Yes" option is highlighted with a red box. The terminal window has a standard Windows title bar with minimize, maximize, and close buttons. The taskbar at the bottom of the screen shows various pinned icons, including File Explorer, Edge browser, File History, Mail, and others. The system tray on the right side of the taskbar shows the date and time as "29-01-2025 09:18", along with icons for battery level, signal strength, and language settings.

27. To Extract the specific characters from last

```
yaswanth@3bb36d3d774953e:~$ nano subs2.sh
yaswanth@3bb36d3d774953e:~$ chmod a+x subs2.sh
yaswanth@3bb36d3d774953e:~$ cat subs2.sh
str="We welcome you on Javatpoint."
substr="${str: -11}"
echo "$substr"
yaswanth@3bb36d3d774953e:~$ ./subs2.sh
Javatpoint.
yaswanth@3bb36d3d774953e:~$
```

The screenshot shows a Windows terminal window titled "subs2.sh *". The command "nano subs2.sh" is run to edit the script. The script contains the following code:

```
str="We welcome you on Javatpoint."
substr="${str: -11}"
echo "$substr"
```

The command "chmod a+x subs2.sh" is used to make the script executable. Finally, the command "./subs2.sh" is run, outputting the string "Javatpoint.".

The taskbar at the bottom of the screen includes icons for File Explorer, Edge browser, File Explorer, Mail, and Task View. The system tray shows the date and time as "29-01-2025 09:20".

28. The basic example of String Concatenation, and we do not need any extra operator or function in this method.

```
yaswanth@3bb36d3d774953e:~$ nano cat1.sh
yaswanth@3bb36d3d774953e:~$ chmod a+x cat1.sh
yaswanth@3bb36d3d774953e:~$ ./cat1.sh
We welcome you on Javatpoint.
yaswanth@3bb36d3d774953e:~$
```

This screenshot shows a Windows terminal window. The command `nano cat1.sh` is run to create a new file named `cat1.sh`. The file contains the text "We welcome you on Javatpoint.". The command `chmod a+x cat1.sh` is used to make the script executable. Finally, `./cat1.sh` is executed, outputting the welcome message to the terminal.

```
yaswanth@3bb36d3d774953e:~$ nano 7.2
GNU nano 7.2                                     cat1.sh *
#Declaring the first String
stri="We welcome you"
#Declaring the Second String
str2=" on Javatpoint."
#Combining first and second string
str3="$stri$str2"
#Printing a new string by combining both
echo $str3
```

This screenshot shows a Windows terminal window with the nano editor open. The script `cat1.sh` contains the following code:

```
#Declaring the first String
stri="We welcome you"
#Declaring the Second String
str2=" on Javatpoint."
#Combining first and second string
str3="$stri$str2"
#Printing a new string by combining both
echo $str3
```

The terminal shows the command `nano 7.2` to open the file, the file content itself, and the command `cat1.sh` to execute the script.

29. Using Append Operator with Loop

```
yaswanth@3bb36d3d774953e:~$ nano cat2.sh
yaswanth@3bb36d3d774953e:~$ chmod a+x cat2.sh
yaswanth@3bb36d3d774953e:~$ ./cat2.sh
Printing the name of the programming languages
javapythonCC++
yaswanth@3bb36d3d774953e:~$
```

The screenshot shows a Windows desktop environment. At the top, there is a taskbar with various icons for applications like File Explorer, Edge, and Google Chrome. Below the taskbar is a search bar with the placeholder "Type here to search". The main area of the desktop contains two windows: a terminal window and a file editor window.

The terminal window (top) displays the command-line session from the previous screenshot:

```
yaswanth@3bb36d3d774953e:~$ nano cat2.sh
yaswanth@3bb36d3d774953e:~$ chmod a+x cat2.sh
yaswanth@3bb36d3d774953e:~$ ./cat2.sh
Printing the name of the programming languages
javapythonCC++
yaswanth@3bb36d3d774953e:~$
```

The file editor window (bottom) shows the content of the file "cat2.sh" in the nano text editor:

```
GNU nano 7.2
#!/bin/bash
echo "Printing the name of the programming languages"
#Initializing the variable before combining
lang=""
#for loop for reading the list
for value in 'java' 'python' 'C' 'C++';
do
lang+="$value "
done
#Printing the combined values
echo "$lang"
```

The file editor window has 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, To Bracket, and Where Was. The status bar at the bottom shows "ENG 10:47 US 29-01-2025".

30. Using Literal Strings

```
>Select yaswanth@3bb36d3d774953e: ~
yaswanth@3bb36d3d774953e:~$ nano cat3.sh
yaswanth@3bb36d3d774953e:~$ chmod a+x cat2.sh
yaswanth@3bb36d3d774953e:~$ chmod a+x cat3.sh
yaswanth@3bb36d3d774953e:~$ ./cat3.sh
Welcome to Javatpoint.
yaswanth@3bb36d3d774953e:~$
```

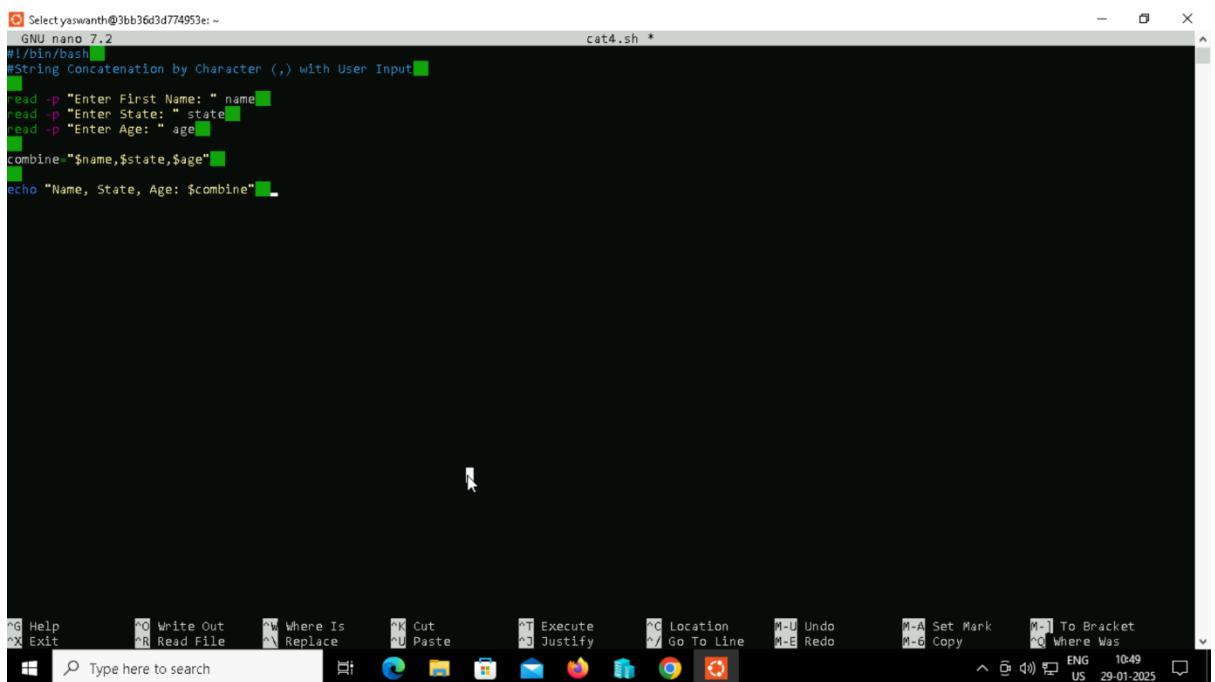
This screenshot shows a Windows terminal window. The command prompt is "Select yaswanth@3bb36d3d774953e: ~". The user runs "nano cat3.sh" to edit a file, then "chmod a+x cat2.sh" and "chmod a+x cat3.sh" to make them executable. Finally, they run "./cat3.sh" which outputs "Welcome to Javatpoint.". The taskbar at the bottom shows various application icons.

```
yaswanth@3bb36d3d774953e: ~
GNU nano 7.2
#!/bin/bash
str="Welcome to"
newstr="${str} Javatpoint."
echo "$newstr"
```

This screenshot shows a Windows terminal window with the nano editor open. The file is named "cat3.sh". The code defines a variable "str" with the value "Welcome to", creates a new string "newstr" by concatenating "str" and " Javatpoint.", and then prints "newstr". The taskbar at the bottom shows various application icons.

31. Using any Character

```
yashwanth@9bb36d3d774953e:~$ nano cat4.sh
yashwanth@9bb36d3d774953e:~$ chmod a+x cat4.sh
yashwanth@9bb36d3d774953e:~$ ./cat4.sh
Enter First Name: yashwanth
Enter State: andhra pradesh
Enter Age: 22
Name, State, Age: yashwanth, andhra pradesh, 22
yashwanth@9bb36d3d774953e:~$
```



The screenshot shows a Windows desktop environment with a terminal window open. The terminal window title is "Selectyashwanth@9bb36d3d774953e: ~". Inside the window, the script content is displayed:

```
GNU nano 7.2
#!/bin/bash
#String Concatenation by Character (,) with User Input
read -p "Enter First Name: " name
read -p "Enter State: " state
read -p "Enter Age: " age
combine="$name,$state,$age"
echo "Name, State, Age: $combine"
```

The terminal window has a standard Windows-style menu bar at the top, including "File", "Edit", "View", "Insert", "Format", "Search", "Tools", and "Help". Below the menu bar is a toolbar with icons for "Help", "Exit", "Write Out", "Read File", "Where Is", "Replace", "Cut", "Paste", "Execute", "Justify", "Location", "Go To Line", "Undo", "Redo", "Set Mark", "Copy", "To Bracket", and "Where Was". At the bottom of the screen is the Windows taskbar, which includes the Start button, a search bar with the placeholder "Type here to search", and icons for various applications like File Explorer, Edge, Mail, and Google Chrome.

32. code that illustrates the procedure on how to pass arguments to functions, and access the arguments inside the function.

The screenshot shows a Windows desktop environment. In the top-left corner, there is a terminal window titled "fun1.sh *". It contains the following code:

```
yaswanth@3bb36d3d774953e:~$ nano fun1.sh
function_arguments()
{
    echo $1
    echo $2
    echo $3
    echo $4
    echo $5
}
function_arguments "We""welcome""you""on""Javatpoint."

```

In the bottom-right corner, there is a file explorer window titled "File Name to Write: fun1.sh". The file content is identical to the terminal code. The taskbar at the bottom shows various pinned icons, and the system tray indicates the date and time as 29-01-2025.

The screenshot shows a Windows desktop environment. In the top-left corner, there is a terminal window titled "fun1.sh". It displays the output of running the script:

```
yaswanth@3bb36d3d774953e:~$ ./fun1.sh
WewelcomeyouonJavatpoint.
```

Below the terminal, another terminal window shows the command to check for the file:

```
yaswanth@3bb36d3d774953e:~$ cat fun1
cat: fun1: No such file or directory
yaswanth@3bb36d3d774953e:~$ cat fun1.sh
function_arguments()
{
    echo $1
    echo $2
    echo $3
    echo $4
    echo $5
}
function_arguments "We""welcome""you""on""Javatpoint."

```

The taskbar at the bottom shows various pinned icons, and the system tray indicates the date and time as 29-01-2025.

33. To understand how variables scope works in Bash Scripting

The screenshot shows a terminal window titled "fun2.sh *". The code inside the terminal is as follows:

```
yaswanth@3bb36d3d774953e:~$ nano 7.2
GNU nano 7.2
v1='A'
v2='B'

my_var () {
local v1='C'
v2='D'
echo "Inside Function"
echo "v1 is $v1."
echo "v2 is $v2."

echo "Before Executing the Function"
echo "v1 is $v1."
echo "v2 is $v2."

my_var
echo "After Executing the Function"
echo "v1 is $v1."
echo "v2 is $v2."
```

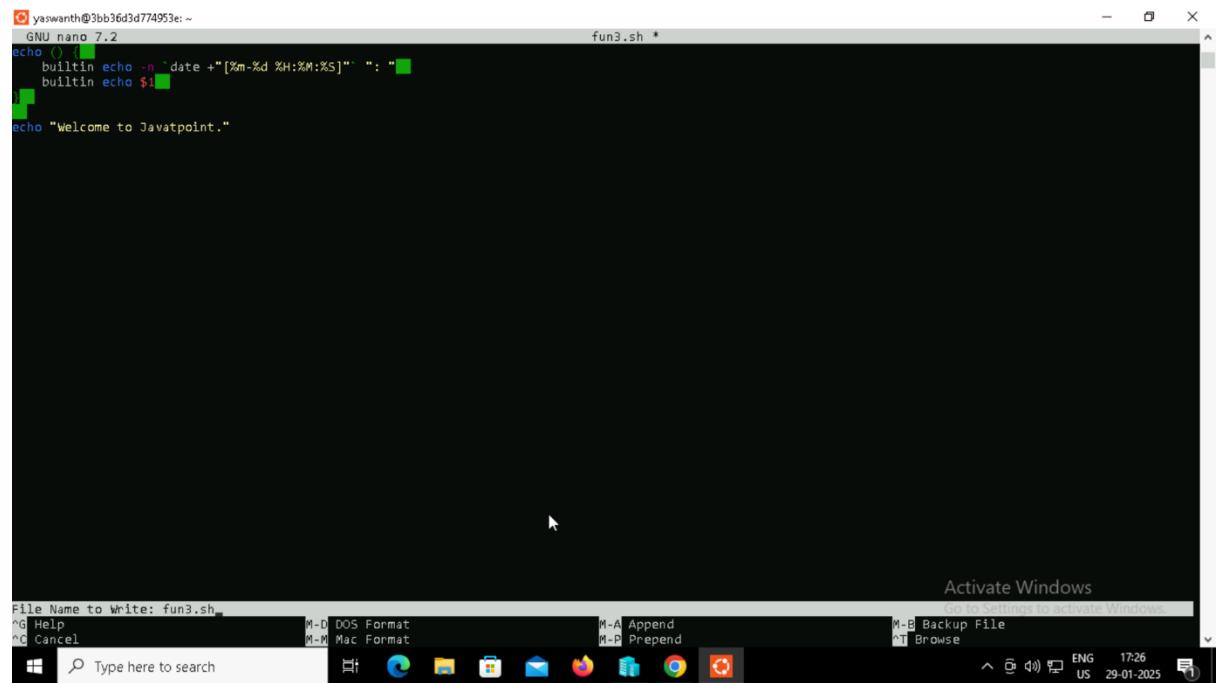
Below the terminal window, a Windows taskbar is visible with various icons and a search bar. A tooltip "Activate Windows" with the text "Go to Settings to activate Windows." is shown.

The screenshot shows a terminal window with the following command and output:

```
yaswanth@3bb36d3d774953e:~$ function arguments "We""welcome""you""on""Javatpoint."
yaswanth@3bb36d3d774953e:~$ nano fun2.sh
yaswanth@3bb36d3d774953e:~$ chmod a+x fun2.sh
yaswanth@3bb36d3d774953e:~$ ./fun2.sh
Before Executing the Function
v1 is A.
v2 is B.
Inside Function
v1 is C.
v2 is D.
After Executing the Function
v1 is A.
v2 is D.
```

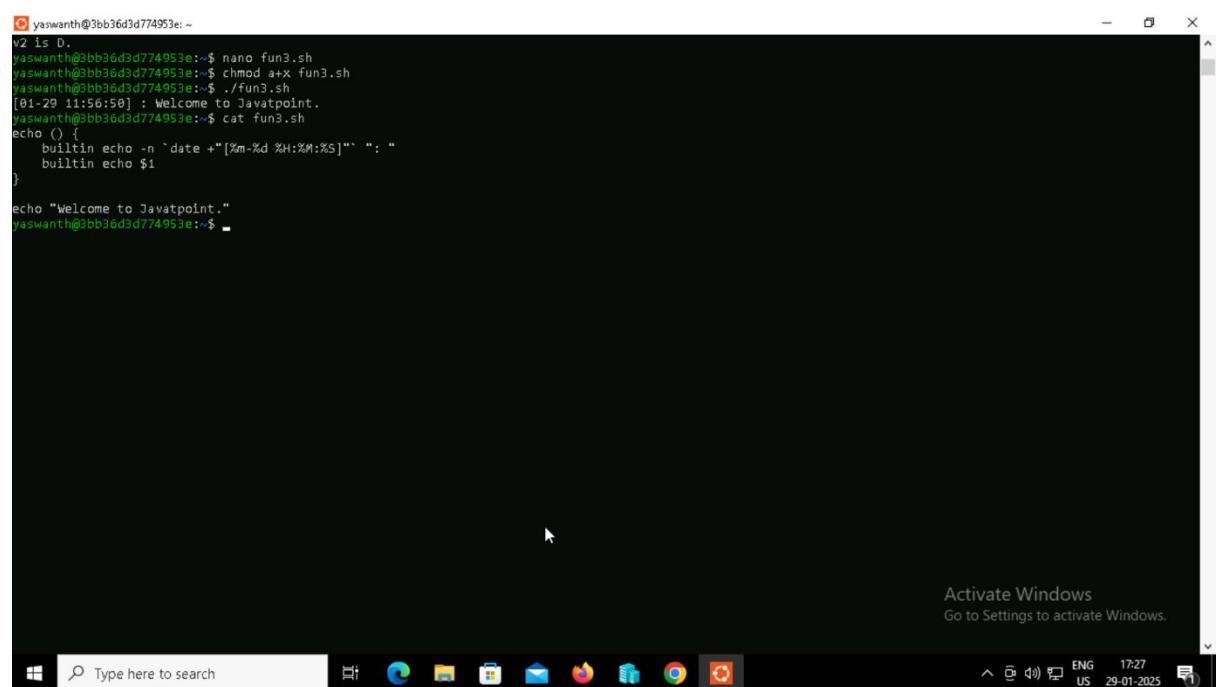
Below the terminal window, a Windows taskbar is visible with various icons and a search bar. A tooltip "Activate Windows" with the text "Go to Settings to activate Windows." is shown.

34. overridden the 'echo' command and added the time stamp in the form of the argument to the 'echo' command



```
yaswanth@3bb36d3d774953e:~$ nano fun3.sh
GNU nano 7.2
echo () {
    builtin echo -n `date +*[%m-%d %H:%M:%S]`": "
    builtin echo $1
}
echo "Welcome to Javatpoint."
yaswanth@3bb36d3d774953e:~$
```

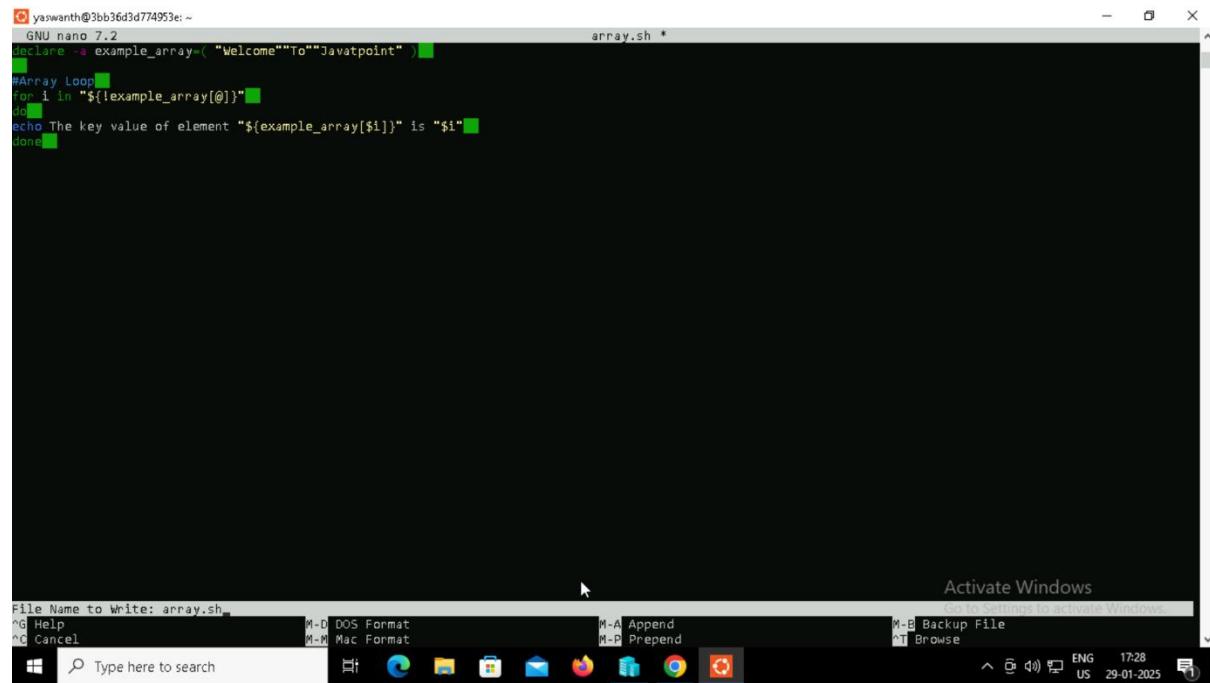
The screenshot shows a Windows terminal window titled "fun3.sh". It contains the code for a shell script named "fun3.sh". The script uses a function definition to print the current date and time followed by a colon and the value of the variable \$1. A welcome message is also present at the end. The terminal window has a standard Windows title bar and taskbar.



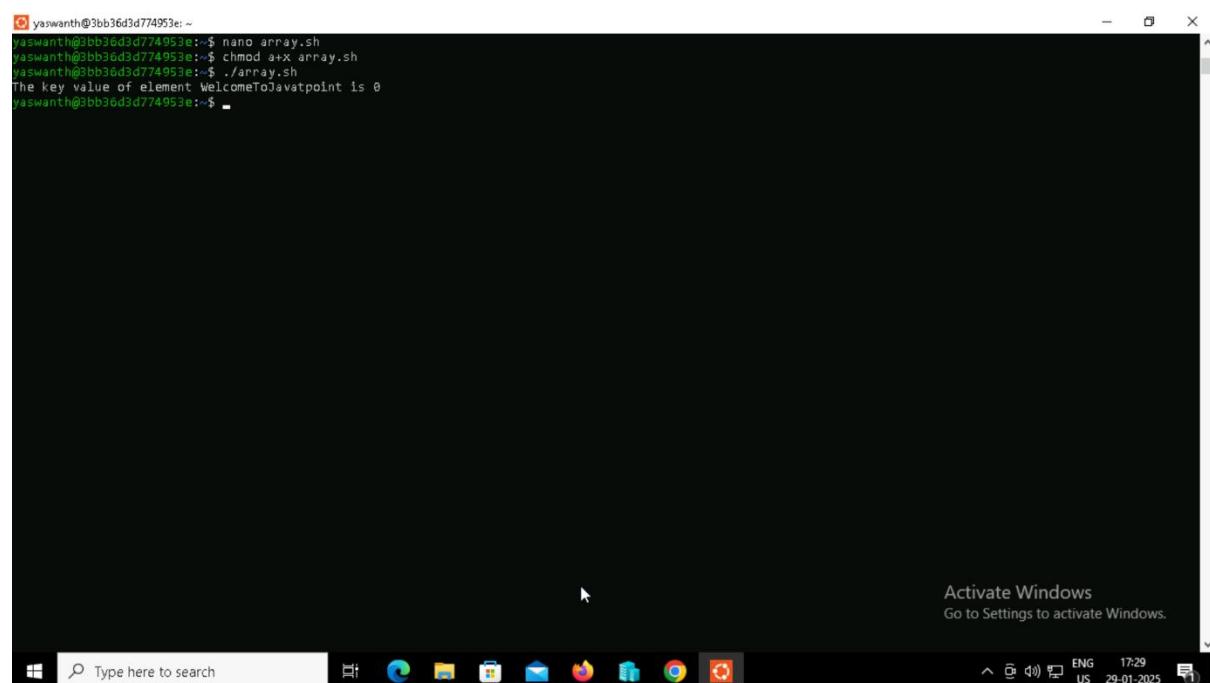
```
File Name to Write: fun3.sh
^G Help ^M-D DOS Format ^M-A Append
^Q Cancel ^M-M Mac Format ^M-P Prepend
^M-B Backup File
^M-T Browse
Type here to search  Activate Windows
Go to Settings to activate Windows.
ENG 17:26 29-01-2025
v2 is D.
yaswanth@3bb36d3d774953e:~$ nano fun3.sh
yaswanth@3bb36d3d774953e:~$ chmod a+x fun3.sh
yaswanth@3bb36d3d774953e:~$ ./fun3.sh
[01-29 11:56:50] : Welcome to Javatpoint.
yaswanth@3bb36d3d774953e:~$ cat fun3.sh
echo () {
    builtin echo -n `date +*[%m-%d %H:%M:%S]`": "
    builtin echo $1
}
echo "Welcome to Javatpoint."
yaswanth@3bb36d3d774953e:~$
```

The screenshot shows a Windows terminal window with a dark theme. It displays the output of running the "fun3.sh" script. The script prints the current date and time followed by a colon and the string "Welcome to Javatpoint.". The terminal window includes a file menu, keyboard shortcuts, and system status indicators like language and date.

35. The general method to iterate over each item in an array is by using the 'for loop'.



```
yaswanth@3bb36d3d774953e: ~
GNU nano 7.2
array.sh *
declare -a example_array=( "Welcome""To""Javatpoint" )
#Array Loop
for i in "${example_array[@]}"
do
echo The key value of element "${example_array[$i]}" is "$i"
done
```



```
File Name to Write: array.sh
File Name to Write: array.sh
^D Help M-D DOS Format M-A Append
^Q Cancel M-M Mac Format M-P Prepend
M-B Backup File
^T Browse
Type here to search
Activate Windows
Go to Settings to activate Windows.
ENG 17:28
US 29-01-2025
```

```
yaswanth@3bb36d3d774953e: ~$ nano array.sh
yaswanth@3bb36d3d774953e: ~$ chmod a+x array.sh
yaswanth@3bb36d3d774953e: ~$ ./array.sh
The key value of element WelcomeToJavatpoint is 0
yaswanth@3bb36d3d774953e: ~$
```

```
Activate Windows
Go to Settings to activate Windows.
ENG 17:29
US 29-01-2025
```

36 . Adding Elements to an Array

```
yaswanth@3bb36d3d774953e: ~
GNU nano 7.2                                         array1.sh *
declare -a example_array=( "Java""Python""PHP""HTML" )
example_array[4]="JavaScript"
echo "${example_array[@]}"
```

The screenshot shows a terminal window titled "array1.sh *". It contains a shell script that declares an array "example_array" with elements "Java", "Python", "PHP", "HTML", and adds "JavaScript" at index 4. The script then prints the entire array using the \${example_array[@]} syntax.

```
File Name to Write: array1.sh_
^D Help           M-D DOS Format      M-A Append
^Q Cancel        M-M Mac Format      M-P Prepend
M-B Backup File
^T Browse
Type here to search
```

Activate Windows
Go to Settings to activate Windows.

17:30 ENG US 29-01-2025

```
yaswanth@3bb36d3d774953e: ~
chmod: cannot access 'array1.sh': No such file or directory
yaswanth@3bb36d3d774953e: ~$ nano array1.sh
yaswanth@3bb36d3d774953e: ~$ chmod a+x array1.sh
yaswanth@3bb36d3d774953e: ~$ ./array1.sh
JavaPythonPHPHTML JavaScript
yaswanth@3bb36d3d774953e: ~$ cat array1.sh
declare -a example_array=( "Java""Python""PHP""HTML" )
#Adding new element
example_array[4]="JavaScript"
#Printing all the elements
echo "${example_array[@]}"
yaswanth@3bb36d3d774953e: ~ -
```

The screenshot shows a terminal window with a file dialog open, prompting for a file name to write. The user has typed "array1.sh_". Below the dialog, the terminal shows the execution of the script. It first attempts to run "array1.sh" but fails because it doesn't exist. Then it runs "nano array1.sh" to edit the script. After saving and exiting nano, it runs "chmod a+x array1.sh" to make the script executable. Finally, it runs the script itself, which prints the elements of the array followed by "JavaScript".

37 . Deleting the Entire Array

Save modified buffer? Yes No Cancel

Activate Windows
Go to Settings to activate Windows

Windows Start button Type here to search Taskbar icons: File Explorer, Edge, File Manager, Mail, Firefox, Chrome, Task View, Power User, Network, System, Date/Time, Language, Battery

```
yaswanth@3bb36d3d774953e:~$ nano array2.sh
#!/bin/bash
#Script to delete the entire Array

declare -a example_array=( "Java""Python""HTML""CSS""JavaScript" )
unset example_array
echo ${!example_array[@]}
echo ${!example_array[@]}

yaswanth@3bb36d3d774953e:~$ chmod a+x array2.sh
yaswanth@3bb36d3d774953e:~$ ./array2.sh
-bash: ./array2.sh: Permission denied

yaswanth@3bb36d3d774953e:~$ ./array2.sh

yaswanth@3bb36d3d774953e:~$ cat array2.sh
#!/bin/bash
#Script to delete the entire Array

declare -a example_array=( "Java""Python""HTML""CSS""JavaScript" )
unset example_array
echo ${!example_array[@]}
echo ${!example_array[@]}

yaswanth@3bb36d3d774953e:~$
```

