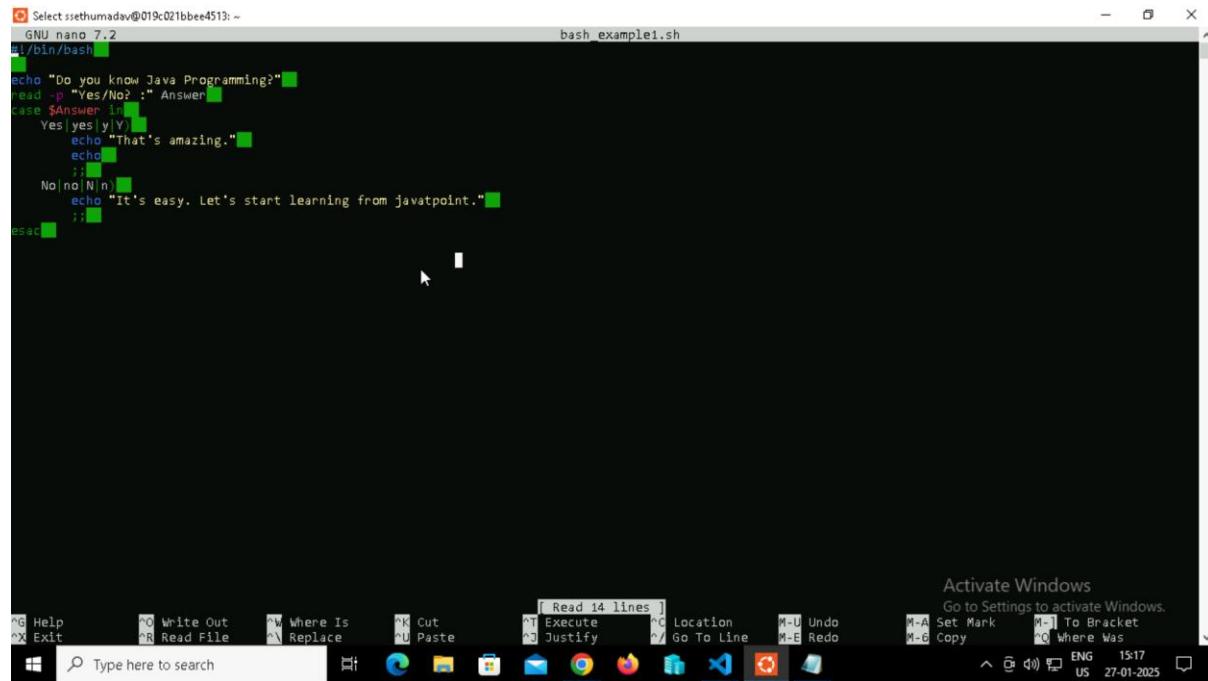


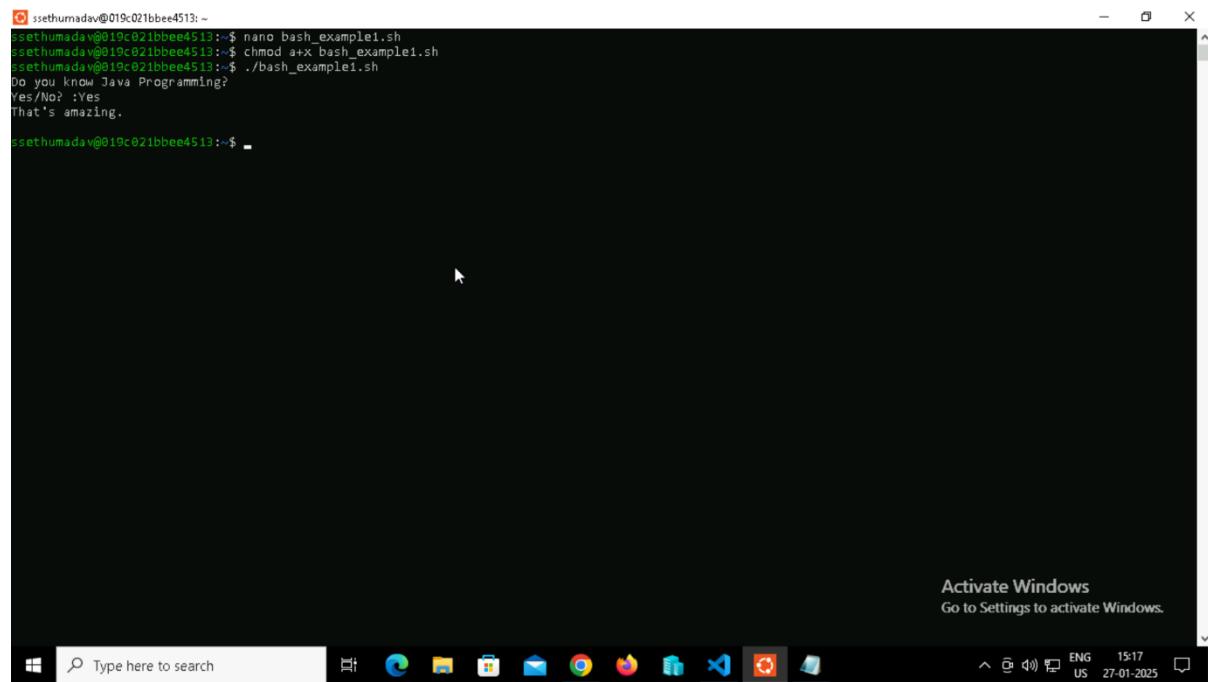
BASH PROJECT:

example 1- (In this example, we have defined a simple scenario to demonstrate the use of the case statement.)



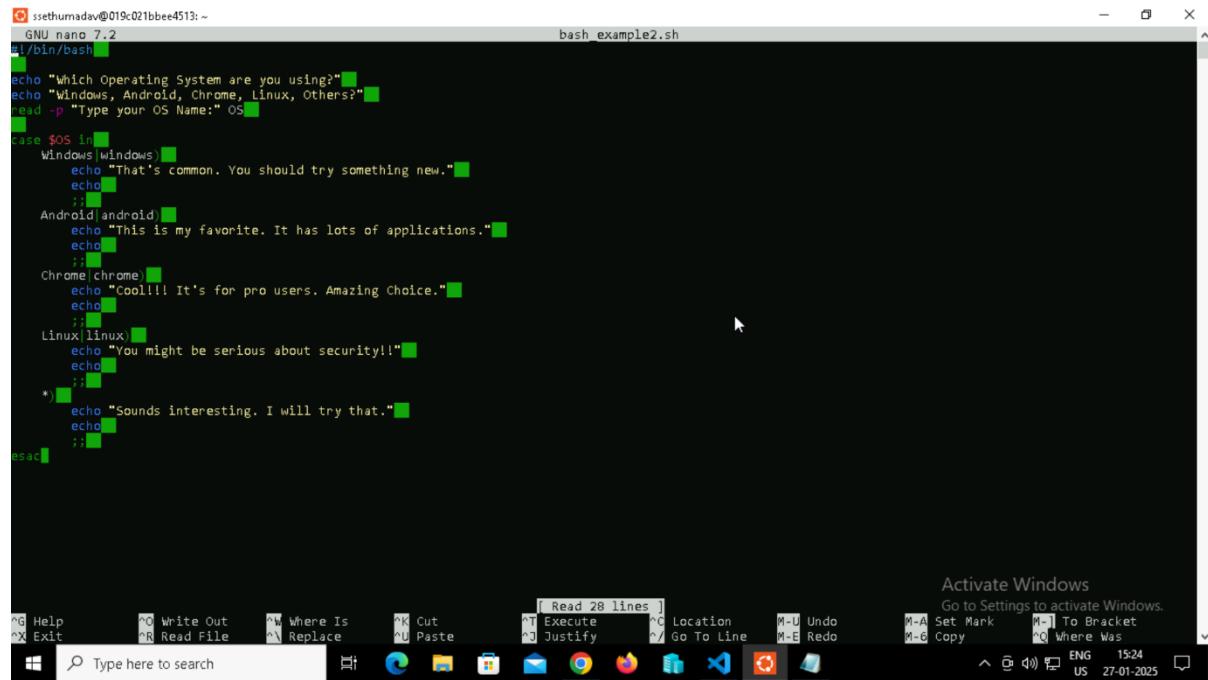
```
GNU nano 7.2                                bash_example1.sh
#!/bin/bash

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
```



```
ssethumadav@019c021bbbe4513:~$ nano bash_example1.sh
ssethumadav@019c021bbbe4513:~$ chmod a+x bash_example1.sh
ssethumadav@019c021bbbe4513:~$ ./bash_example1.sh
Do you know Java Programming?
Yes/No? :yes
That's amazing.
```

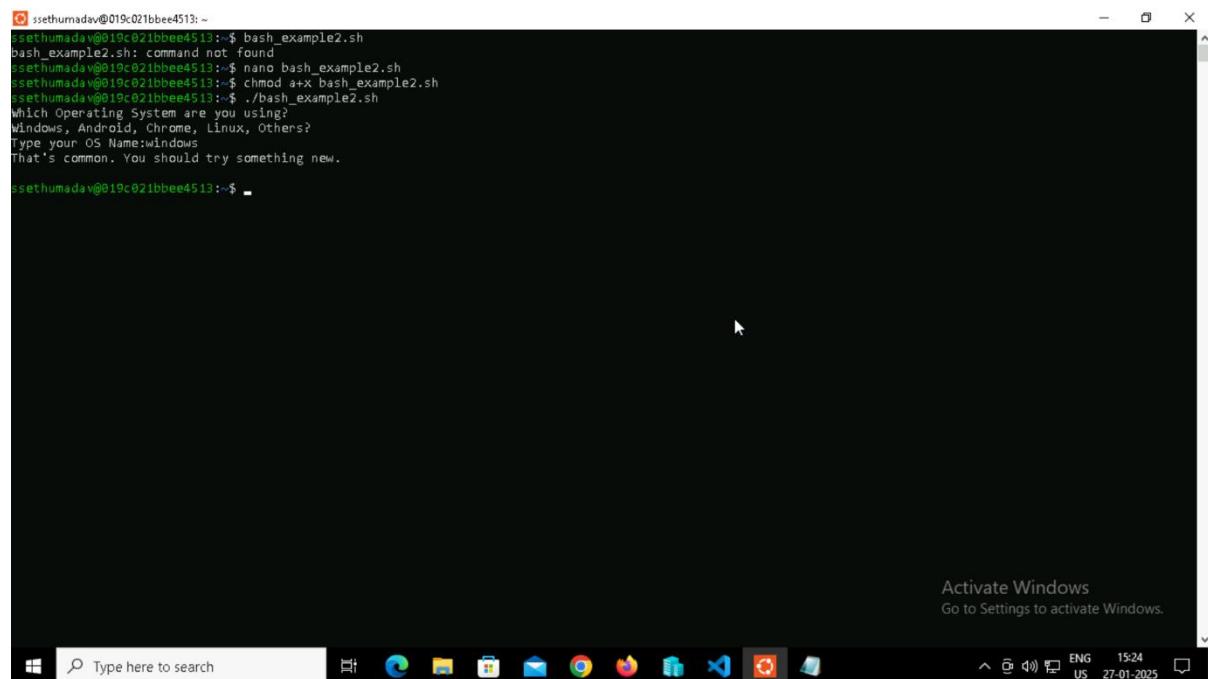
EXAMPLE2: (In this example, we have defined a combined scenario where there is also a default case when no previous matched case is found.)



```
ssethumadav@019c021bbe4513:~$ nano bash_example2.sh
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|Windows)
        echo "That's common. You should try something new."
        echo ;;
    Android|android)
        echo "This is my favorite. It has lots of applications."
        echo ;;
    Chrome|chrome)
        echo "Cool!!! It's for pro users. Amazing Choice."
        echo ;;
    Linux|linux)
        echo "You might be serious about security!!"
        echo ;;
    *)
        echo "Sounds interesting. I will try that."
        echo ;;
esac
```



```
ssethumadav@019c021bbe4513:~$ bash_example2.sh
bash_example2.sh: command not found
ssethumadav@019c021bbe4513:~$ nano bash_example2.sh
ssethumadav@019c021bbe4513:~$ chmod a+x bash_example2.sh
ssethumadav@019c021bbe4513:~$ ./bash_example2.sh
Which Operating System are you using?
Windows, Android, Chrome, Linux, Others?
Type your OS Name:Windows
That's common. You should try something new.
```

BASH FOR LOOP EXAMPLES:

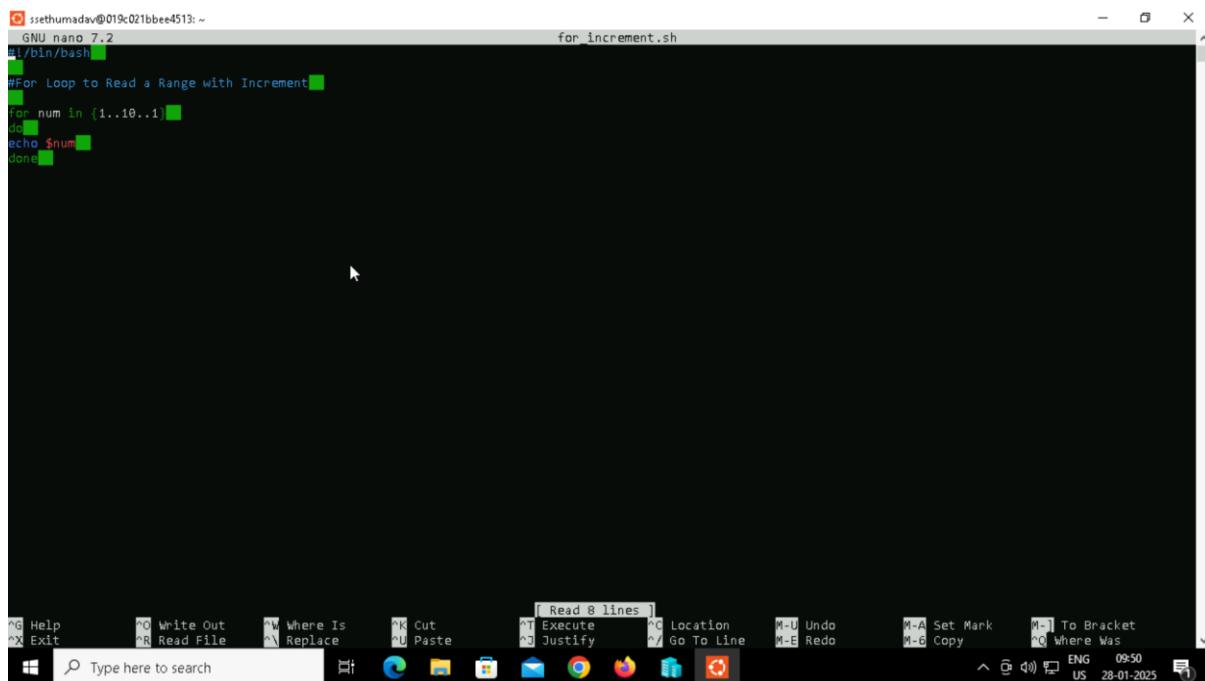
Example1:

The image shows a Windows desktop environment with two terminal windows open. The top terminal window is a standard black terminal window with a white background. It displays the command line and the output of a Bash script named 'bash_ex1.sh'. The bottom terminal window is a nano editor window, also showing the same script code. The desktop taskbar at the bottom includes icons for File Explorer, Task View, Start, Taskbar settings, and several pinned applications like Edge, File Explorer, Mail, Photos, and a browser.

```
ssethumadav@019c021bbe4513:~$ nano bash_ex1.sh
ssethumadav@019c021bbe4513:~$ chmod a+x bash_ex1.sh
ssethumadav@019c021bbe4513:~$ ./bash_ex1.sh
Start
learning
from
Davatpoint.
Thank You.
ssethumadav@019c021bbe4513:~$
```

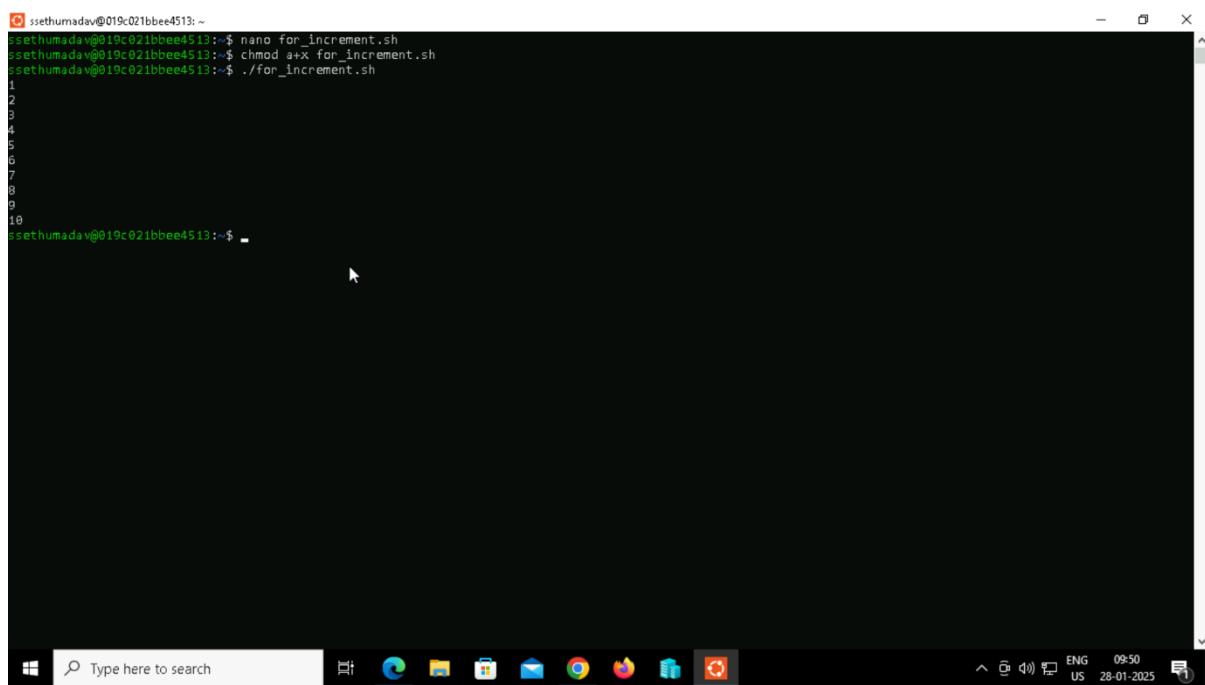
```
ssethumadav@019c021bbe4513:~$ GNU nano 7.2                                bash_ex1.sh
#!/bin/bash
#This is the basic example of 'for loop'.
learn="Start learning from Javatpoint."
For learn in $learn
do
echo $learn
done
echo "Thank You."
```

Example 2:for increment



```
ssethumadav@019c021bbe4513:~  
GNU nano 7.2  
#!/bin/bash  
  
#For Loop to Read a Range with Increment  
  
for num in {1..10..1}  
do  
echo $num  
done
```

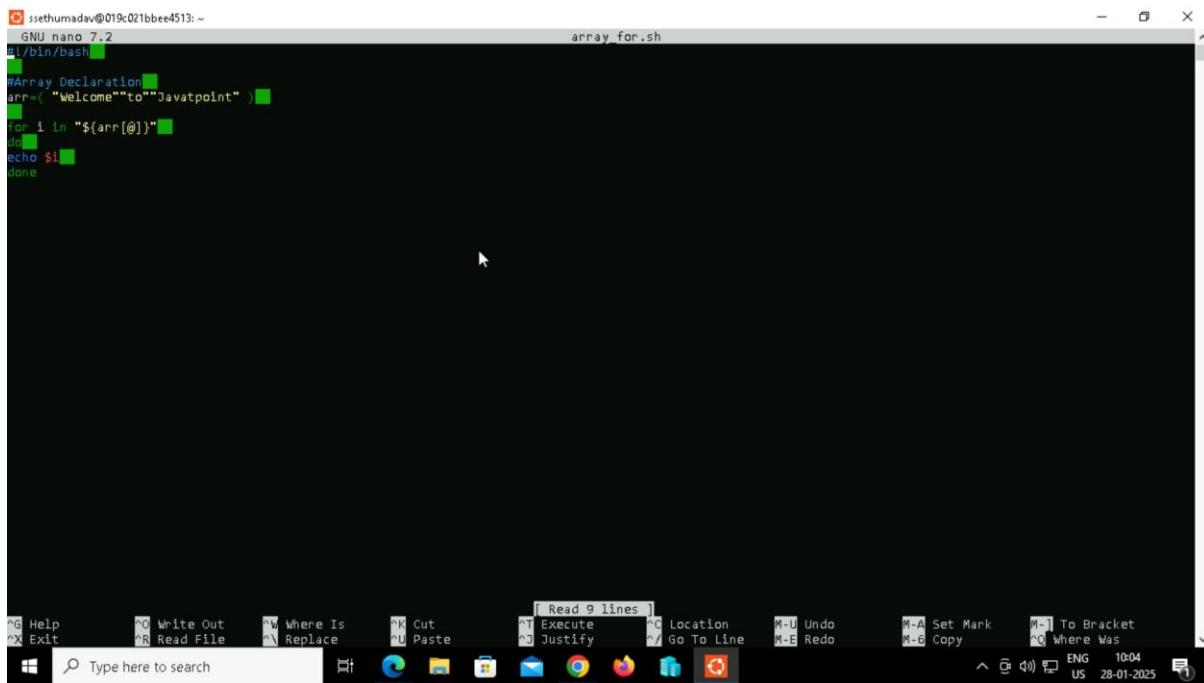
The screenshot shows a terminal window titled "for increment.sh" running in the nano editor. The script contains a for loop that iterates from 1 to 10 with an increment of 1, printing each number on a new line. The terminal window includes a menu bar with options like Help, Exit, Write Out, Read File, Where Is, Replace, Cut, Paste, Execute, Location, Go To Line, Undo, Redo, Set Mark, To Bracket, Copy, and Where Was. The status bar at the bottom shows the path "ssethumadav@019c021bbe4513:~\$", the date and time "28-01-2025 09:50", and the keyboard layout "ENG US".



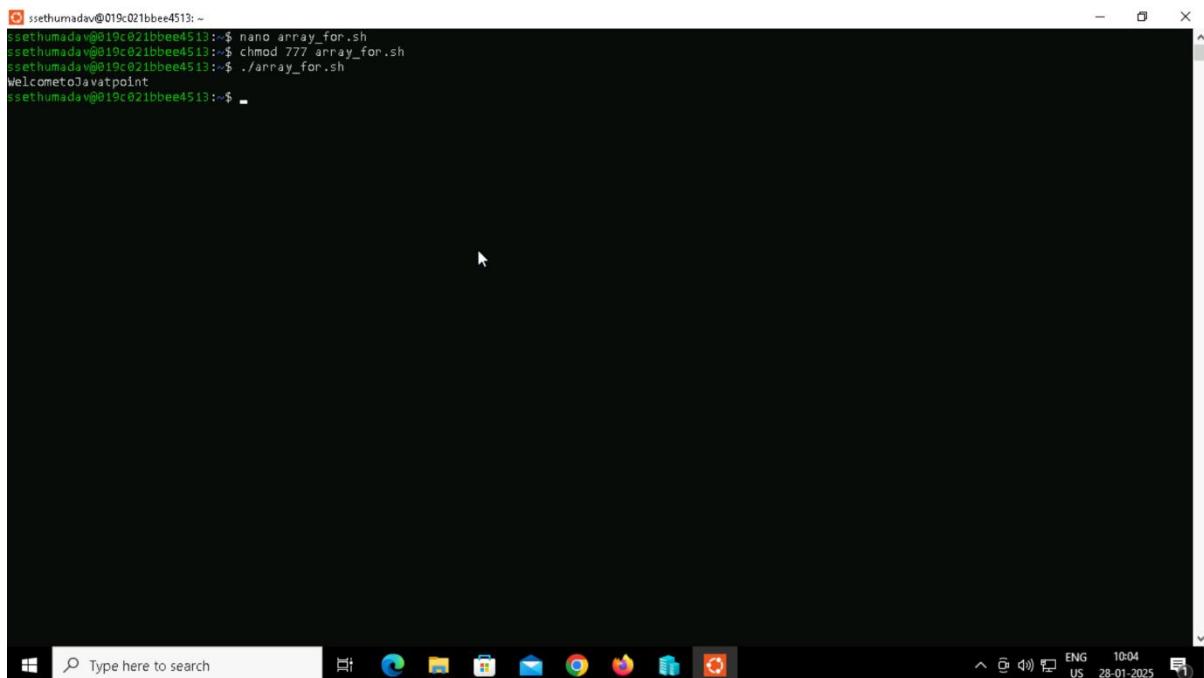
```
ssethumadav@019c021bbe4513:~$ nano for_increment.sh  
ssethumadav@019c021bbe4513:~$ chmod a+x for_increment.sh  
ssethumadav@019c021bbe4513:~$ ./for_increment.sh  
1  
2  
3  
4  
5  
6  
7  
8  
9  
10
```

The screenshot shows a terminal window running a command-line interface. It displays the command to edit the script, change its mode to executable, and then run it. The output shows the numbers 1 through 10 printed sequentially. The terminal window has a similar interface to the one above, with a menu bar, status bar, and taskbar at the bottom.

Example 3:for loop to read array variables using bash:

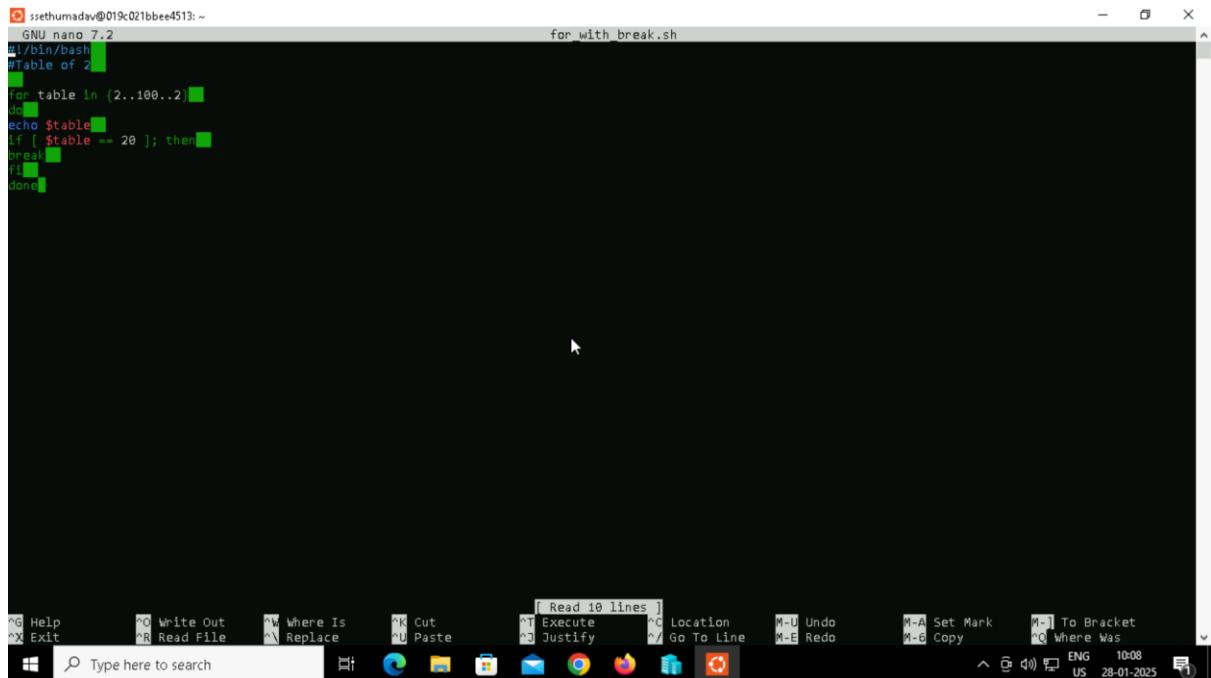


```
ssethumadav@019c021bbbe4513: ~
GNU nano 7.2
#!/bin/bash
#Array Declaration
arr=( "Welcome" "to" "Javatpoint" )
for i in "${arr[@]}"
do
echo $i
done
```

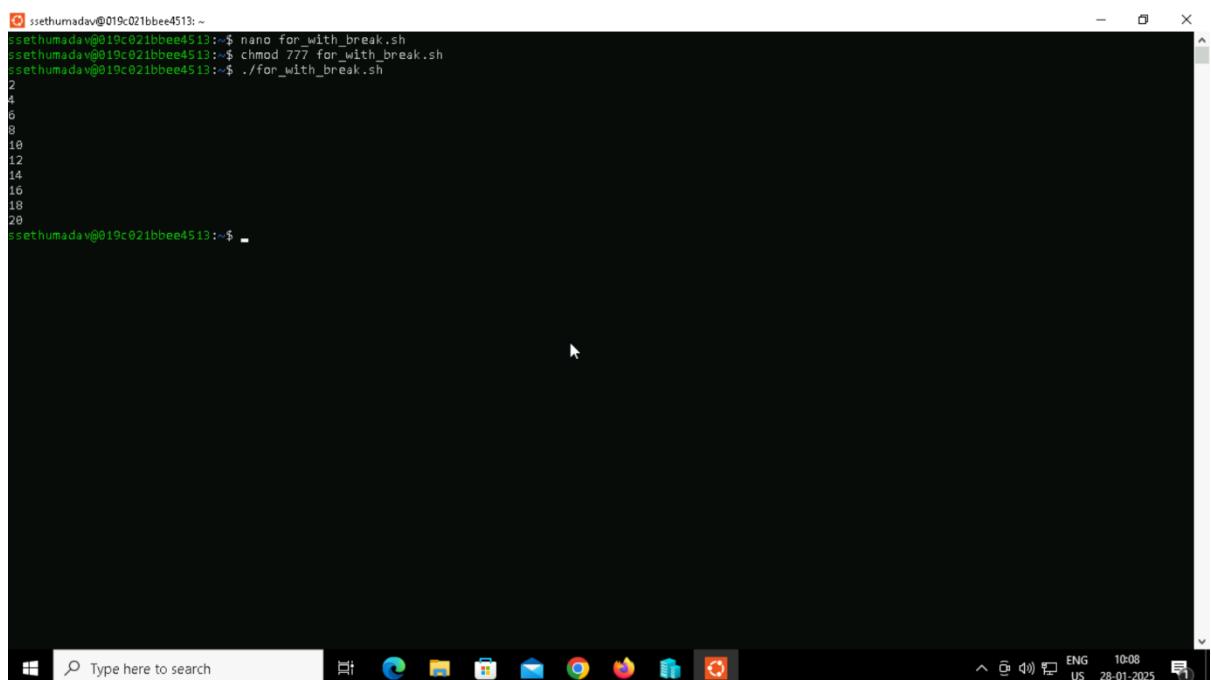


```
ssethumadav@019c021bbbe4513: ~$ nano array_for.sh
ssethumadav@019c021bbbe4513: ~$ chmod 777 array_for.sh
ssethumadav@019c021bbbe4513: ~$ ./array_for.sh
Welcome to Javatpoint
ssethumadav@019c021bbbe4513: ~$
```

Example 4: for loop with break statement:



```
ssethumadav@019c021bbe4513:~  
GNU nano 7.2  
#!/bin/bash  
#Table of 2  
  
for table in {2..100..2}  
do  
echo $table  
if [ $table == 20 ]; then  
break  
fi  
done
```



```
ssethumadav@019c021bbe4513:~$ nano for_with_break.sh  
ssethumadav@019c021bbe4513:~$ chmod 777 for_with_break.sh  
ssethumadav@019c021bbe4513:~$ ./for_with_break.sh  
2  
4  
6  
8  
10  
12  
14  
16  
18  
20  
Break  
ssethumadav@019c021bbe4513:~$
```

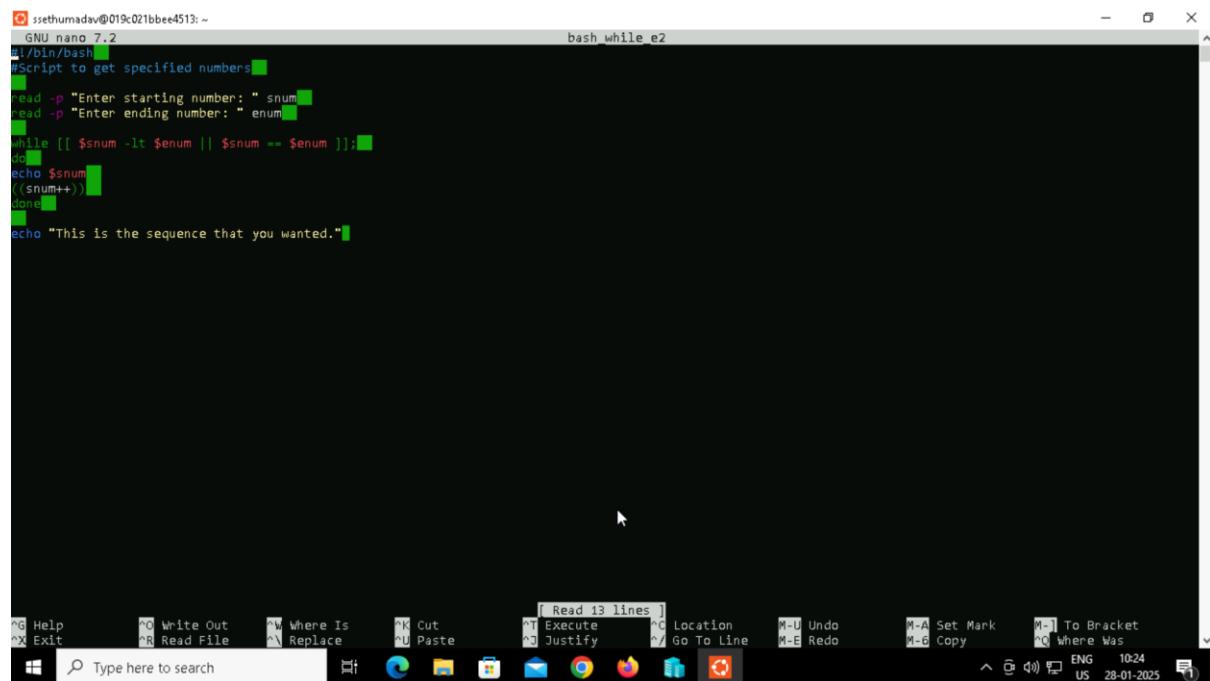
Bash While loops example:

Example 1:while loop with single condition:

```
ssethumadav@019c021bbe4513:~  
GNU nano 7.2  
#!/bin/bash  
#Script to get specified numbers  
  
read -p "Enter starting number: " snum  
read -p "Enter ending number: " enum  
  
while [[ $snum -le $enum ]]; do  
echo $snum  
((snum++))  
done  
  
echo "This is the sequence that you wanted." #!/bin/bash  
#Script to get specified numbers  
  
read -p "Enter starting number: " snum  
read -p "Enter ending number: " enum  
  
while [[ $snum -le $enum ]]; do  
echo $snum  
((snum++))  
done  
  
echo "This is the sequence that you wanted."
```

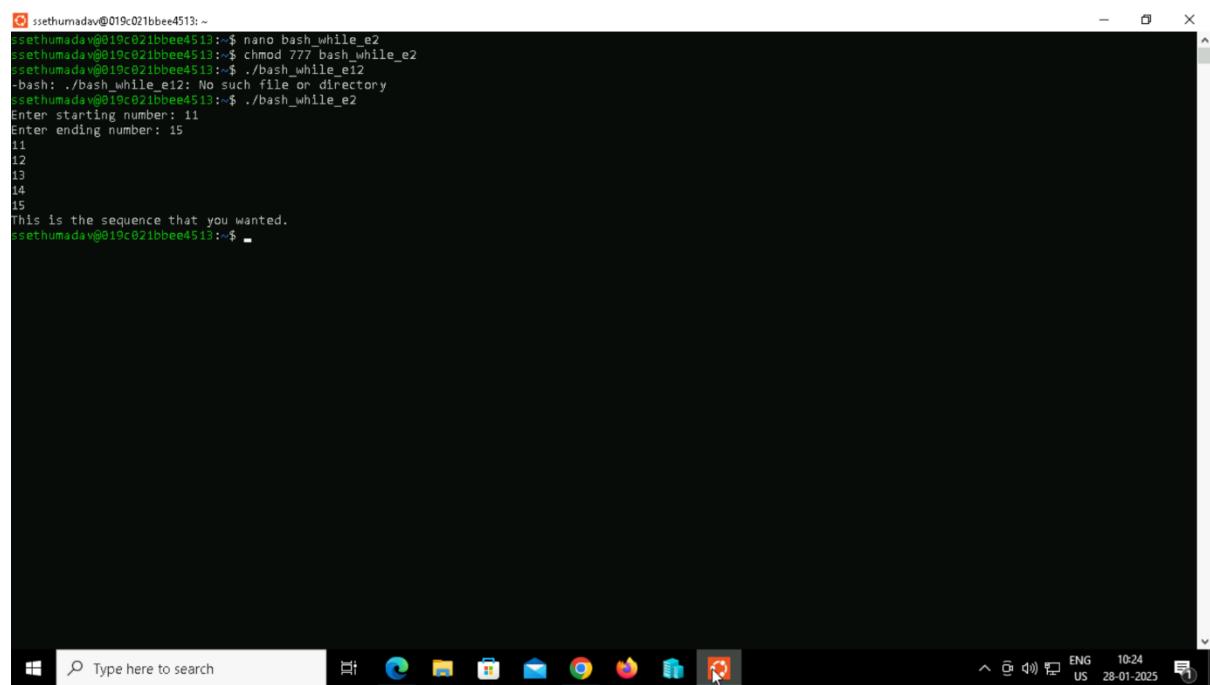
```
ssethumadav@019c021bbe4513:~  
ssethumadav@019c021bbe4513:~$ nano bash_while_e1  
ssethumadav@019c021bbe4513:~$ chmod 777 bash_while_e1  
ssethumadav@019c021bbe4513:~$ ./bash_while_e1  
Enter starting number: 11  
Enter ending number: 23  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
This is the sequence that you wanted.  
Enter starting number: -
```

Example 2:while loop with multiple conditions:



```
ssethumadav@019c021bbe4513:~$ nano bash_while_e2
GNU nano 7.2
#!/bin/bash
#Script to get specified numbers
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."

```



```
ssethumadav@019c021bbe4513:~$ nano bash_while_e2
ssethumadav@019c021bbe4513:~$ chmod 777 bash_while_e2
ssethumadav@019c021bbe4513:~$ ./bash_while_e2
-bash: ./bash_while_e2: No such file or directory
ssethumadav@019c021bbe4513:~$ ./bash_while_e2
Enter starting number: 11
Enter ending number: 15
11
12
13
14
15
This is the sequence that you wanted.
ssethumadav@019c021bbe4513:~$
```

Example 3:infinite while loop

```
ssethumadav@019c021bbe4513: ~
GNU nano 7.2
#!/bin/bash
#An infinite while loop
while :; do echo "Welcome to Javatpoint.>"; done
```

The screenshot shows a terminal window titled 'bash_while_e3'. It displays the command 'nano' being used to edit a file. The file content is an infinite while loop that prints 'Welcome to Javatpoint.' to the screen.

```
ssethumadav@019c021bbe4513: ~
Welcome to Javatpoint.
```

The screenshot shows a terminal window displaying the output of the infinite while loop. The screen is filled with repeated instances of 'Welcome to Javatpoint.' due to the loop's nature.

Example 4: while loop with break:

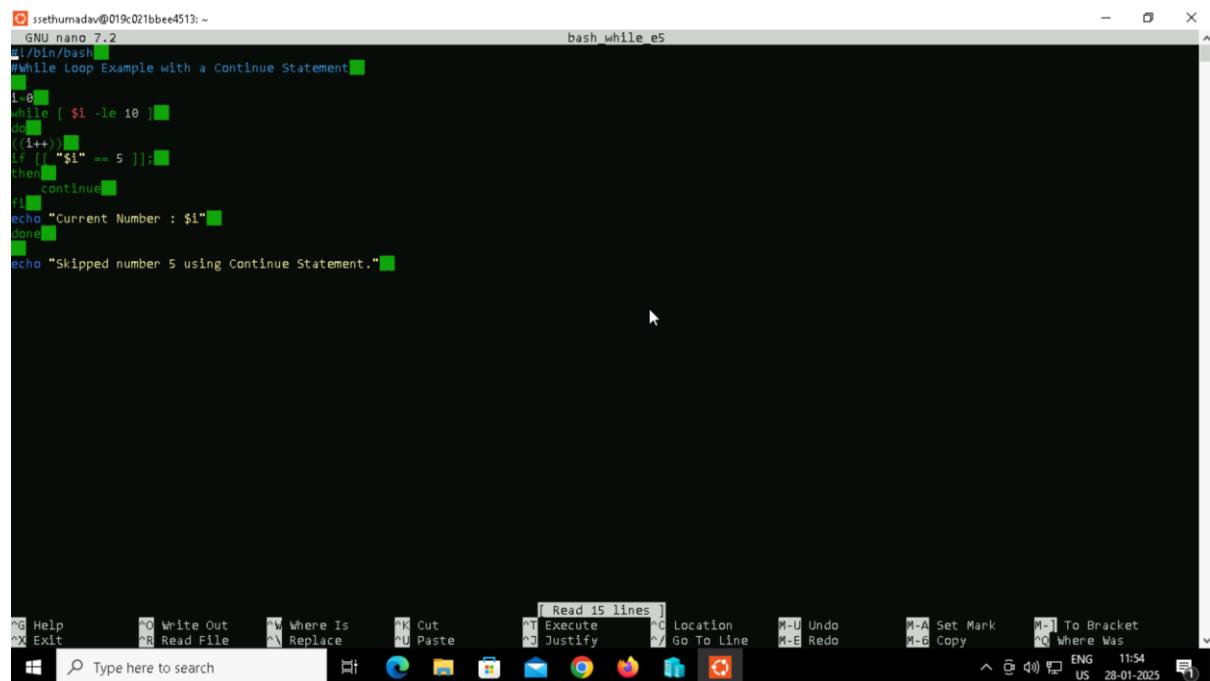
```
ssethumadav@019c021bbeee4513:~$ nano bash_while_e4
GNU nano 7.2
#!/bin/bash
#While Loop Example with a Break Statement
echo "Countdown for Website Launching..."
i=10
while [ $i -ge 1 ]
do
if [ $i == 2 ]
then
echo "Mission Aborted, Some Technical Error Found."
break
fi
echo "$i"
(( i-- ))
done
```

The screenshot shows a terminal window titled "bash_while_e4" containing a Bash script. The script uses a while loop to count down from 10 to 1. If the value of \$i reaches 2, it prints a message and uses the "break" command to exit the loop. The terminal window includes a menu bar with options like Help, Write Out, Where Is, Cut, Paste, Execute, Location, Undo, Set Mark, To Bracket, Read, Replace, and Where Was. Below the menu is a toolbar with icons for Help, Write Out, Where Is, Cut, Paste, Execute, Location, Undo, Set Mark, To Bracket, Read, Replace, and Where Was. At the bottom is a taskbar with various application icons and a search bar.

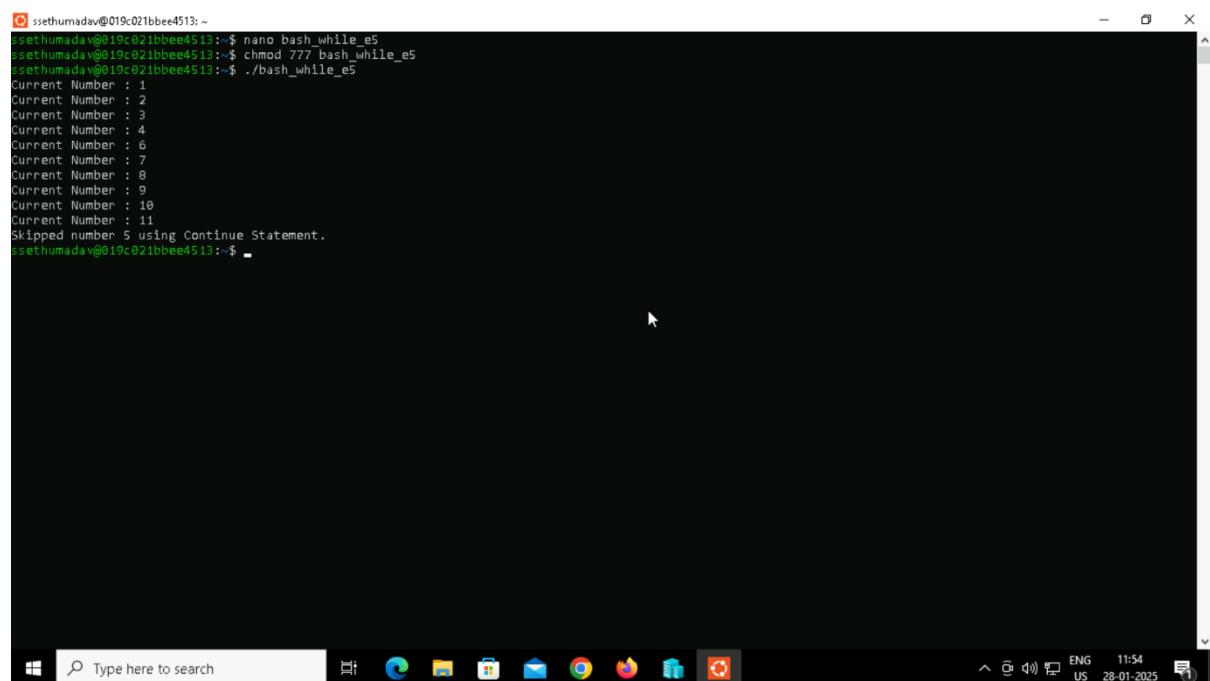
```
ssethumadav@019c021bbeee4513:~$ nano bash_while_e4
ssethumadav@019c021bbeee4513:~$ chmod 777 bash_while_e4
ssethumadav@019c021bbeee4513:~$ ./bash_while_e4
Countdown for Website Launching...
10
9
8
7
6
5
4
3
Mission Aborted, Some Technical Error Found.
ssethumadav@019c021bbeee4513:~$ nano bash_while_e4
ssethumadav@019c021bbeee4513:~$
```

The screenshot shows the same terminal window after executing the script. The output shows the countdown from 10 to 3, followed by the message "Mission Aborted, Some Technical Error Found." and the script exiting. The terminal window interface remains the same as in the previous screenshot.

Example5:while loop with continue:

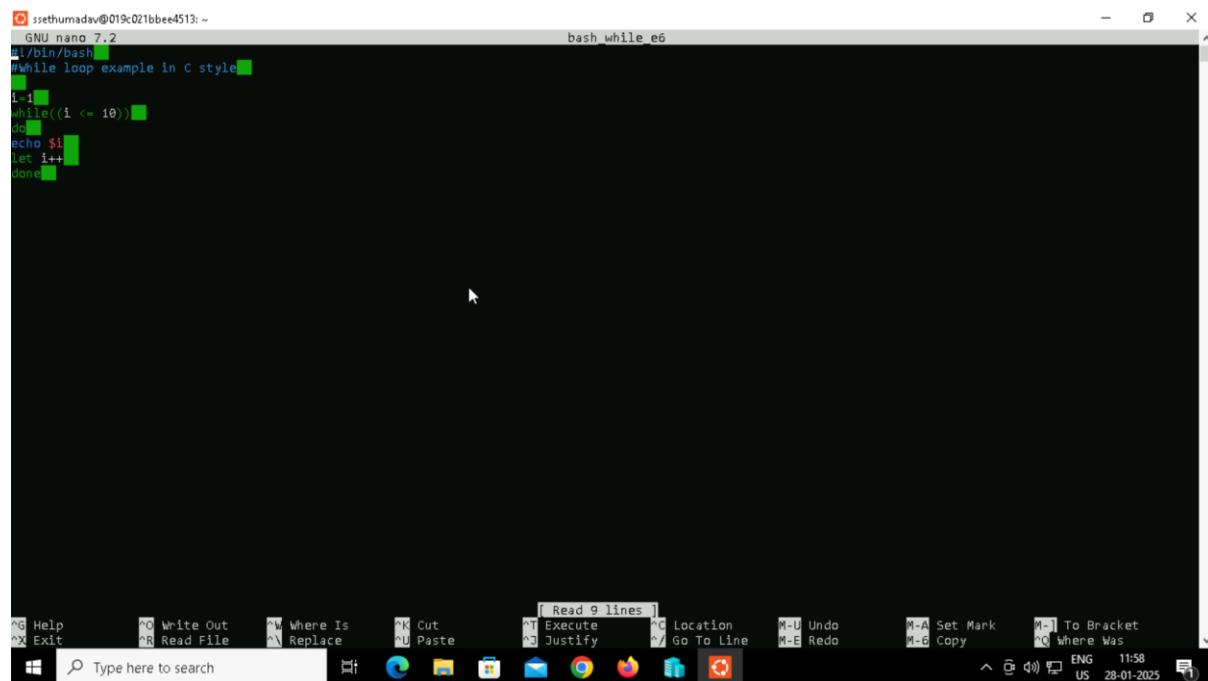


```
ssethumadav@019c021bbe4513:~  
GNU nano 7.2  
#!/bin/bash  
#While Loop Example with a Continue Statement  
  
i=0  
while [ $i -le 10 ]  
do  
((i++))  
if [[ "$i" == 5 ]];  
then  
    continue  
fi  
echo "Current Number : $i"  
done  
  
echo "Skipped number 5 using Continue Statement."
```

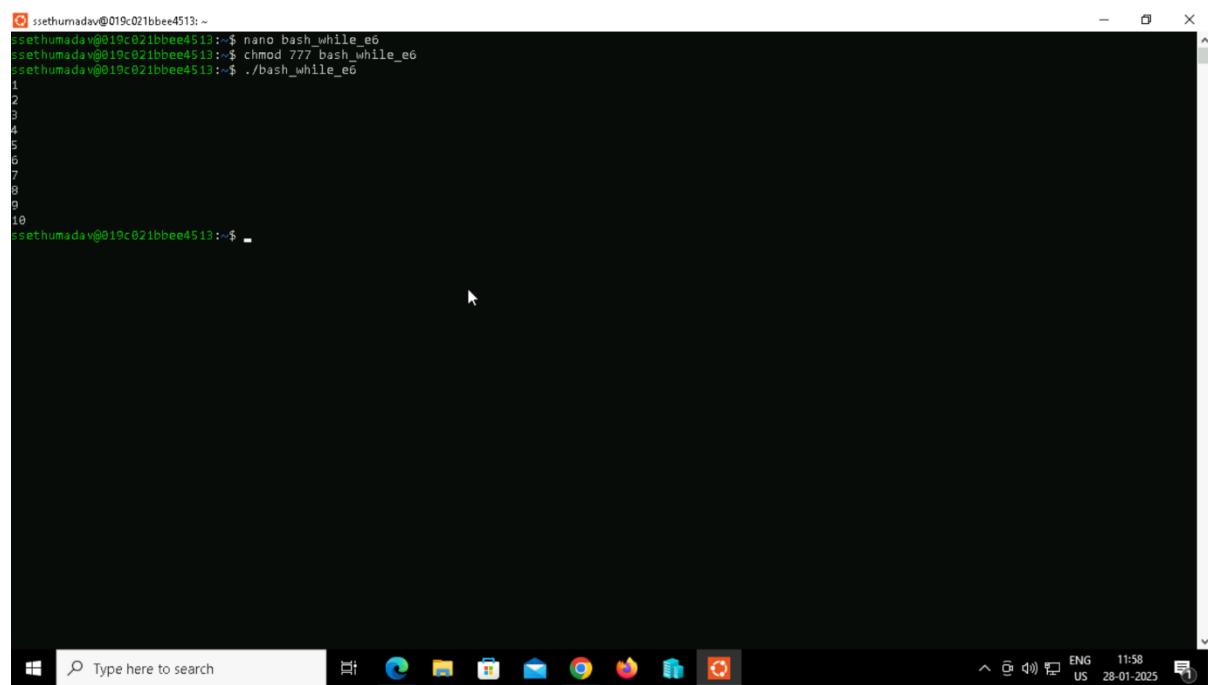


```
ssethumadav@019c021bbe4513:~$ nano bash_while_e5  
ssethumadav@019c021bbe4513:~$ chmod 777 bash_while_e5  
ssethumadav@019c021bbe4513:~$ ./bash_while_e5  
Current Number : 1  
Current Number : 2  
Current Number : 3  
Current Number : 4  
Current Number : 6  
Current Number : 7  
Current Number : 8  
Current Number : 9  
Current Number : 10  
Current Number : 11  
Skipped number 5 using Continue Statement.  
ssethumadav@019c021bbe4513:~$
```

Example 6:while loop in c programming style:



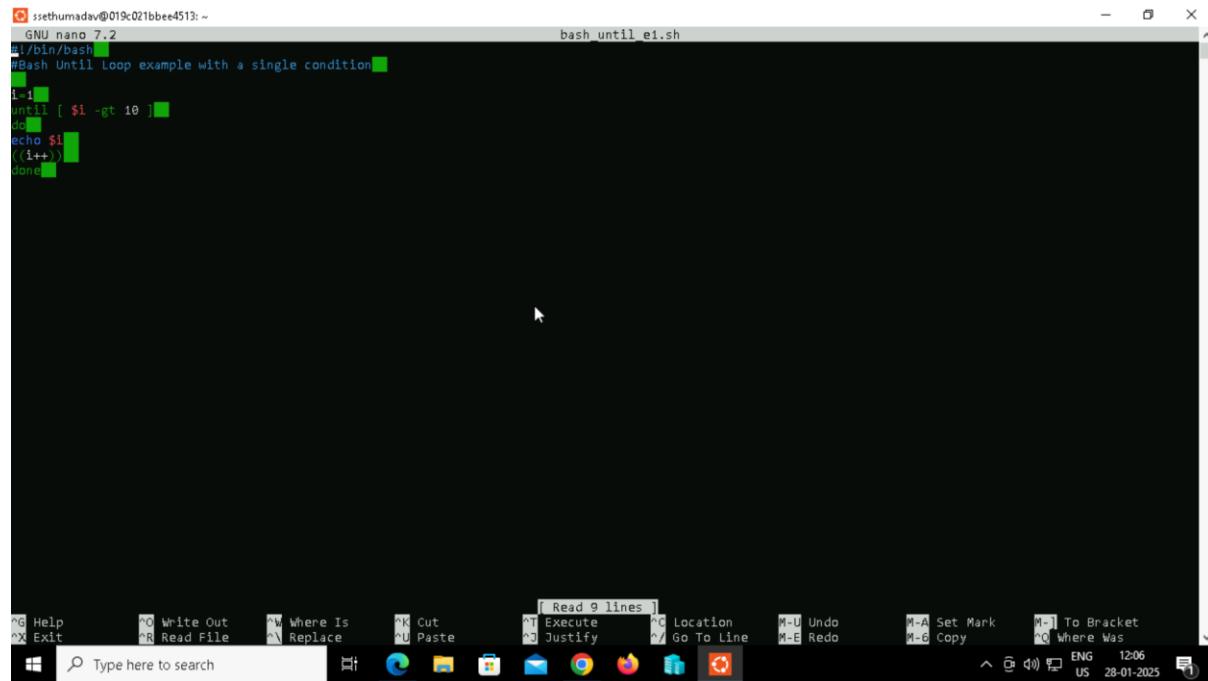
```
ssethumadav@019c021bbe4513:~  
GNU nano 7.2  
#!/bin/bash  
#while loop example in C style  
i=1  
while((i <= 10))  
do  
echo $1  
let i++  
done
```



```
ssethumadav@019c021bbe4513:~$ nano bash_while_e6  
ssethumadav@019c021bbe4513:~$ chmod 777 bash_while_e6  
ssethumadav@019c021bbe4513:~$ ./bash_while_e6  
1  
2  
3  
4  
5  
6  
7  
8  
9  
10
```

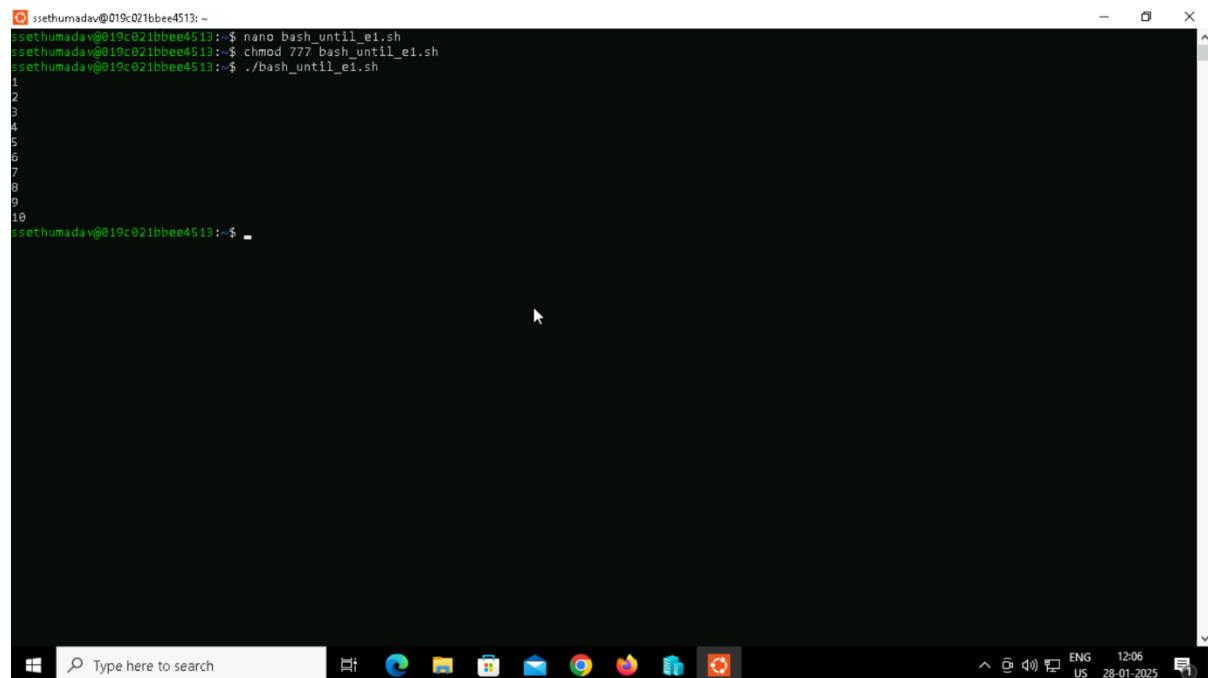
BASH UNTIL LOOP EXAMPLES:

Example 1: until loop with single condition



```
ssethumadav@019c021bbbe4513:~$ nano bash_until_e1.sh
GNU nano 7.2
#!/bin/bash
#Bash Until Loop example with a single condition

i=1
until [ $i -gt 10 ]
do
echo $i
((i++))
done
```



```
ssethumadav@019c021bbbe4513:~$ nano bash_until_e1.sh
ssethumadav@019c021bbbe4513:~$ chmod 777 bash_until_e1.sh
ssethumadav@019c021bbbe4513:~$ ./bash_until_e1.sh
1
2
3
4
5
6
7
8
9
10
```

Example 2:until loop with multiple conditions

A screenshot of a Windows desktop environment. In the center is a terminal window titled "bash until e2.sh". It contains the following code:

```
ssethumadav@019c021bbbe4513:~$ nano bash_until_e2.sh
GNU nano 7.2
#!/bin/bash
#Bash Until Loop example with multiple conditions
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 menu bar at the top with options like Help, Exit, Write Out, Read File, Where Is, Replace, Cut, Paste, Execute, Location, Undo, Redo, Set Mark, To Bracket, and others. Below the menu is a toolbar with icons for search, copy, paste, and others. At the bottom of the terminal window is a status bar showing "Read 13 lines".

At the very bottom of the screen is a Windows taskbar with icons for various applications including File Explorer, Edge, File Manager, Mail, Google Chrome, Firefox, and others. A search bar is also present on the taskbar.

A screenshot of a Windows desktop environment. In the center is a terminal window showing the execution of a script. The output is as follows:

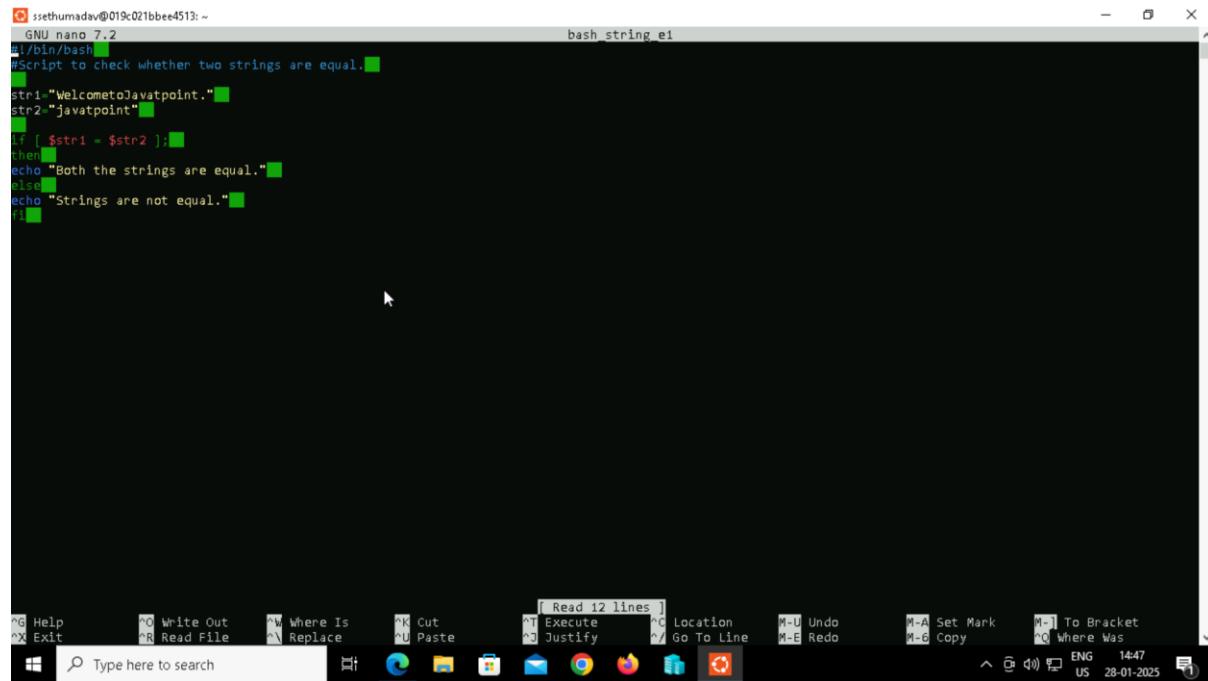
```
ssethumadav@019c021bbbe4513:~$ nano bash_until_e2.sh
ssethumadav@019c021bbbe4513:~$ chmod 777 bash_until_e2.sh
ssethumadav@019c021bbbe4513:~$ ./bash_until_e2.sh
a = 1 & b = 0.
a = 2 & b = 1.
a = 3 & b = 2.
a = 4 & b = 3.
a = 5 & b = 4.
ssethumadav@019c021bbbe4513:~$
```

The terminal window interface is identical to the one in the previous screenshot, with its menu bar, toolbar, and status bar.

The Windows taskbar at the bottom of the screen shows various application icons and a system tray with battery, network, and clock information.

Bash String operations example:

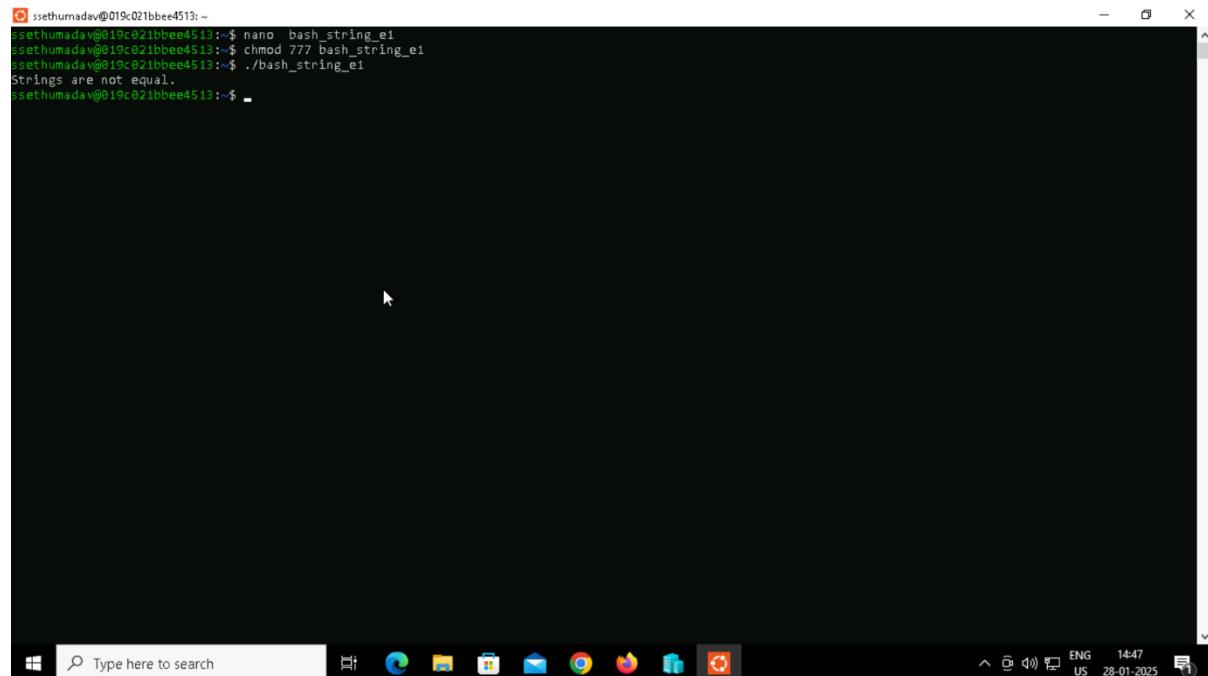
Example 1:to check both strings are equal or not



```
ssethumadav@019c021bbe4513: ~
GNU nano 7.2                                bash_string_e1
#!/bin/bash
#Script to check whether two strings are equal.

str1 "WelcometoJavatpoint."
str2 "javatpoint"

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



```
ssethumadav@019c021bbe4513: ~
ssethumadav@019c021bbe4513:~$ nano bash_string_e1
ssethumadav@019c021bbe4513:~$ chmod 777 bash_string_e1
ssethumadav@019c021bbe4513:~$ ./bash_string_e1
Strings are not equal.
ssethumadav@019c021bbe4513:~$
```

Example 2:using not operator

ssethumadav@019c021bbbe4513:~

GNU nano 7.2

#!/bin/bash

#Script to check whether two strings are equal.

str1="WelcometoJavaPoint."

str2="javatpoint"

If [[\$str1 != \$str2]]; then

echo "Strings are not equal."

else

echo "Strings are equal."

fi

Activate Windows

Go to Settings to activate Windows.

Help White Out Where Is Cut Paste Execute Location Undo Set Mark To Bracket

Exit Read File Replace Justify Go To Line Redo Copy Where Was

Type here to search

Read 12 lines

ENG 14:58 US 28-01-2025

ssethumadav@019c021bbbe4513:~\$ nano bash_string_e2

ssethumadav@019c021bbbe4513:~\$ nano bash_string.e2

ssethumadav@019c021bbbe4513:~\$ chmod 777 bash_string.e2

ssethumadav@019c021bbbe4513:~\$ nano bash_string_e2

ssethumadav@019c021bbbe4513:~\$./bash_string_e2

Strings are not equal.

ssethumadav@019c021bbbe4513:~\$

Activate Windows

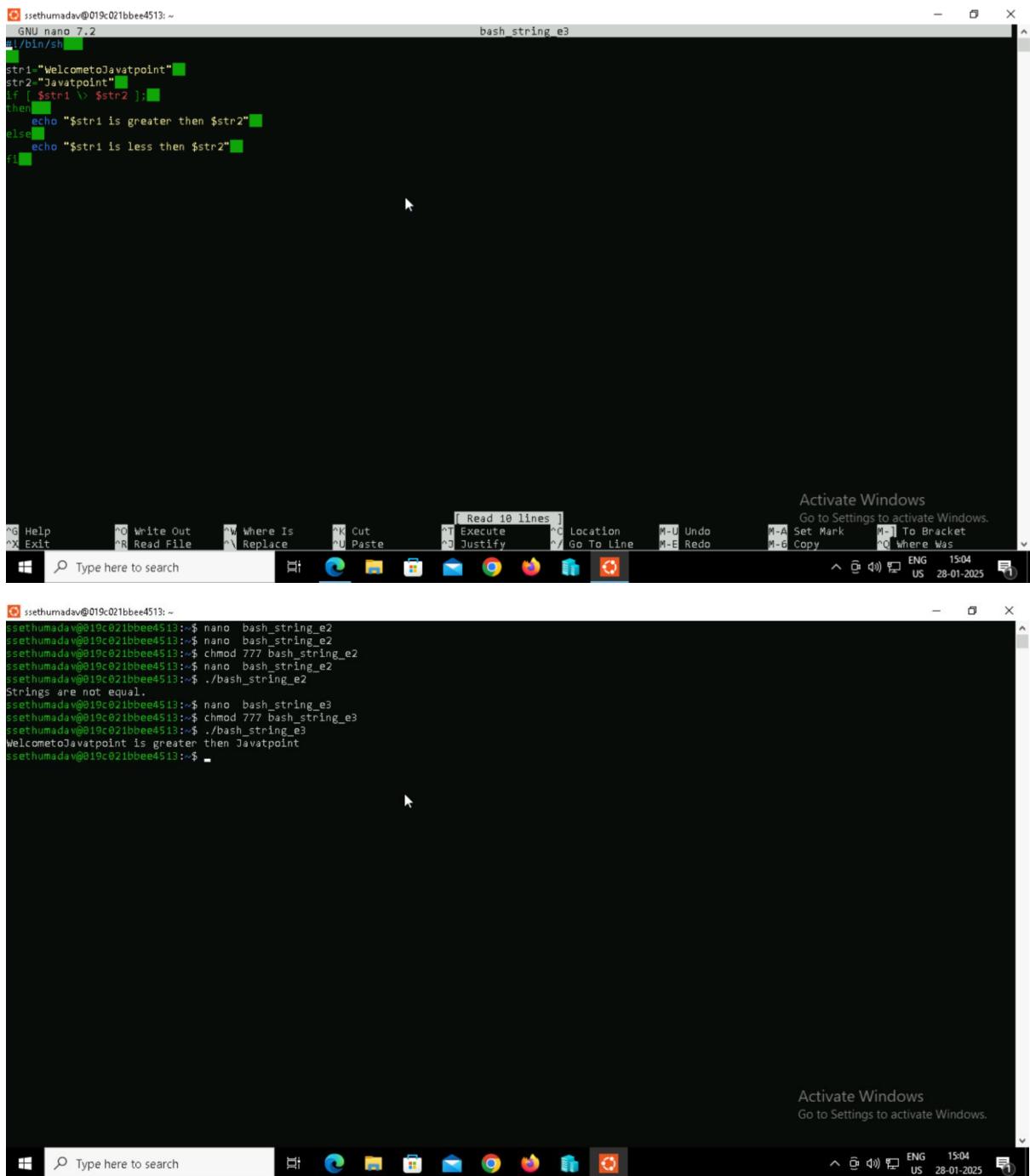
Go to Settings to activate Windows.

Type here to search

Read 12 lines

ENG 15:02 US 28-01-2025

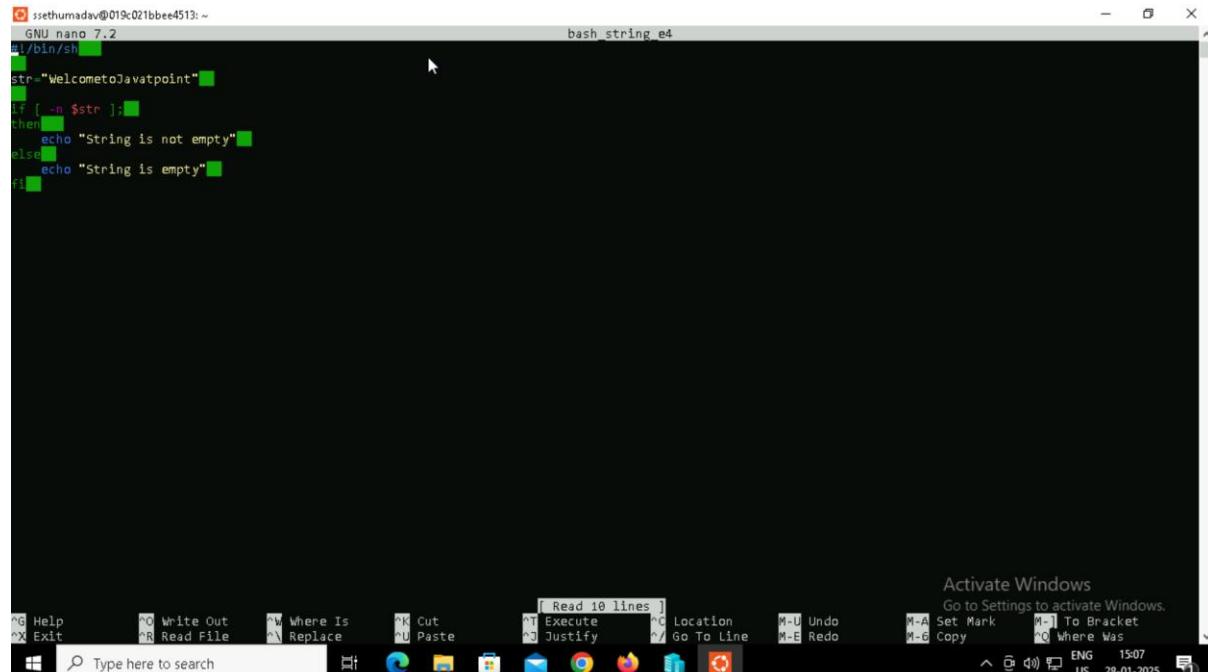
Example 3:



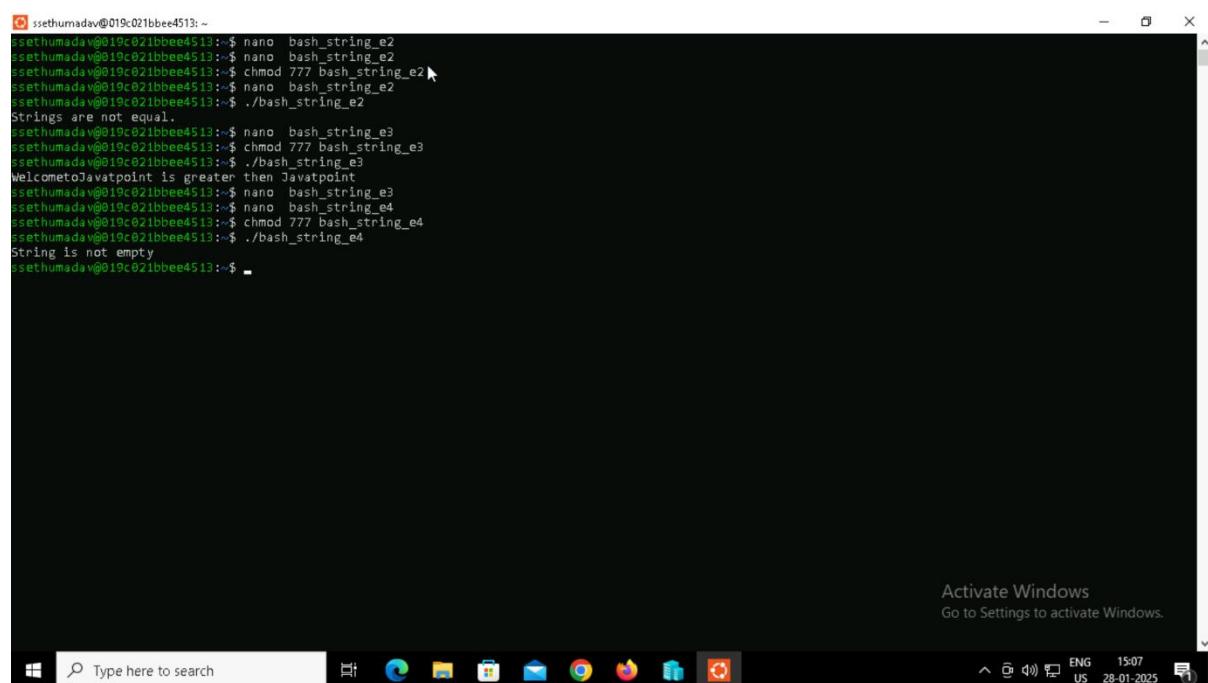
```
ssethumadav@019c021bbbe4513: ~
GNU nano 7.2
#!/bin/sh
str1="WelcometoJavatpoint"
str2="Javatpoint"
if [ $str1 > $str2 ]; then
echo "$str1 is greater than $str2"
else
echo "$str1 is less than $str2"
fi
```

```
ssethumadav@019c021bbbe4513: ~$ nano bash_string_e2
ssethumadav@019c021bbbe4513: ~$ nano bash_string_e2
ssethumadav@019c021bbbe4513: ~$ chmod 777 bash_string_e2
ssethumadav@019c021bbbe4513: ~$ nano bash_string_e2
ssethumadav@019c021bbbe4513: ~$ ./bash_string_e2
Strings are not equal.
ssethumadav@019c021bbbe4513: ~$ nano bash_string_e3
ssethumadav@019c021bbbe4513: ~$ chmod 777 bash_string_e3
ssethumadav@019c021bbbe4513: ~$ ./bash_string_e3
WelcometoJavatpoint is greater than Javatpoint
ssethumadav@019c021bbbe4513: ~$
```

Example4:string empty or not



```
ssethumadav@019c021bbe4513:~$ nano bash_string_e4
GNU nano 7.2
#!/bin/sh
str="WelcometoJavatpoint"
if [ -n $str ]; then
echo "String is not empty"
else
echo "String is empty"
fi
```



```
ssethumadav@019c021bbe4513:~$ nano bash_string_e2
ssethumadav@019c021bbe4513:~$ nano bash_string_e2
ssethumadav@019c021bbe4513:~$ chmod 777 bash_string_e2
ssethumadav@019c021bbe4513:~$ nano bash_string_e2
ssethumadav@019c021bbe4513:~$ ./bash_string_e2
Strings are not equal.
ssethumadav@019c021bbe4513:~$ nano bash_string_e3
ssethumadav@019c021bbe4513:~$ chmod 777 bash_string_e3
ssethumadav@019c021bbe4513:~$ ./bash_string_e3
WelcometoJavatpoint is greater than Javatpoint
ssethumadav@019c021bbe4513:~$ nano bash_string_e4
ssethumadav@019c021bbe4513:~$ nano bash_string_e4
ssethumadav@019c021bbe4513:~$ chmod 777 bash_string_e4
ssethumadav@019c021bbe4513:~$ ./bash_string_e4
String is not empty
ssethumadav@019c021bbe4513:~$
```

Bash split string examples:

Code: example 1 to 5

```
ssethumadav@019c021bbe4513: ~
GNU nano 7.2                                bash split string examples.sh
#!/bin/bash

# Function for Example 1: Split by Space
example_1() {
    echo "Example 1: Split string by space"
    read -p "Enter any string separated by space: " str
    IFS=' ' # setting space as delimiter
    read -ra ADDR << "$str"
    echo "Split string:"
    for i in "${ADDR[@]}"; do
        echo "$i"
    done
}

# Function for Example 2: Split by Symbol (comma)
example_2() {
    echo "Example 2: Split string by comma"
    read -p "Enter Name, State and Age separated by a comma: " entry
    IFS',' # setting comma as delimiter
    read -a strarr << "$entry"
    echo "Name: ${strarr[0]}"
    echo "State: ${strarr[1]}"
    echo "Age: ${strarr[2]}"
}

# Function for Example 3: Split without $IFS using colon as delimiter
example_3() {
    echo "Example 3: Split string without using \$IFS"
    read -p "Enter any string separated by colon(:) " str
    readarray -d : -t strarr << "$str"
    echo "Split string:"
    for ((n=0; n < ${#strarr[*]}; n++)); do
        echo "${strarr[n]}"
    done
}

# Function for Example 4: Split string by another string (Learn)
example_4() {
    echo "Example 4: Split string by 'Learn'"
    str="WeLearnWelcomeLearnYouLearnOnLearnJavaPoint"
    delimiter='Learn'
    s=$str$delimiter
    array=()
    while [[ $s ]]; do
        array+=("${s%%$delimiter*}")
        s=${s#*$delimiter}
    done
    echo "Split string:"
    for element in "${array[@]}"; do
        echo "$element"
    done
}

# Function for Example 5: Split using tr command
example_5() {
    echo "Example 5: Split string using tr command"
    my_str="We;Welcome;you;on;javatpoint."
    my_arr=($echo $my_str | tr ";" "\n")
    echo "Split string:"
    for i in "${my_arr[@]}"; do
        echo $i
    done
}

# Main menu to choose the example
echo "Choose an example to run:"
echo "1. Example 1: Split string by space"
echo "2. Example 2: Split string by comma"
echo "3. Example 3: Split string using colon"
echo "4. Example 4: Split string by 'Learn'"
echo "5. Example 5: Split string using tr command"
read -p "Enter the number of the example you want to run (1-5): " choice
```

```
ssethumadav@019c021bbe4513: ~
GNU nano 7.2                                bash split string examples.sh
Activate Windows
Go to Settings to activate Windows.
M-A Set Mark M-B To Bracket
M-C Copy M-D Where Was
M-U Undo M-E Redo
M-G Copy M-H Where Was
M-F Execute M-J Justify
M-K Location M-L Go To Line
M-O Cut M-P Paste
M-R Read File M-S Replace
M-T Write Out M-V Where Is
M-Z Exit Type here to search ENG 17:46
US 28-01-2025

ssethumadav@019c021bbe4513: ~
GNU nano 7.2                                bash split string examples.sh
Activate Windows
Go to Settings to activate Windows.
M-A Set Mark M-B To Bracket
M-C Copy M-D Where Was
M-U Undo M-E Redo
M-G Copy M-H Where Was
M-F Execute M-J Justify
M-K Location M-L Go To Line
M-O Cut M-P Paste
M-R Read File M-S Replace
M-T Write Out M-V Where Is
M-Z Exit Type here to search ENG 09:27
US 29-01-2025
```

Outputs for example 1 to 5:

1. split string by space:

```
ssethumadav@019c021bbe4513:~$ nano bash_split_string_examples.sh
ssethumadav@019c021bbe4513:~$ chmod 777 bash_split_string_examples.sh
ssethumadav@019c021bbe4513:~$ ./bash_split_string_examples.sh
Choose an example to run:
1. Example 1: Split string by space
2. Example 2: Split string by comma
3. Example 3: Split string using colon
4. Example 4: Split string by 'Learn'
5. Example 5: Split string using tr command
Enter the number of the example you want to run (1-5): 1
Example 1: Split string by space
Enter any string separated by space: sethu madav
Split string:
sethu
madav
ssethumadav@019c021bbe4513:~$
```

2. split string by comma

```
ssethumadav@019c021bbe4513:~$ ./bash_split_string_examples.sh
Choose an example to run:
1. Example 1: Split string by space
2. Example 2: Split string by comma
3. Example 3: Split string using colon
4. Example 4: Split string by 'Learn'
5. Example 5: Split string using tr command
Enter the number of the example you want to run (1-5): 2
Example 2: Split string by comma
Enter Name, State and Age separated by a comma: sethumadav,tamilnadu,22
Name: sethumadav
State: tamilnadu
Age: 22
ssethumadav@019c021bbe4513:~$
```

3. split string using colon

```
ssethumadav@019c021bbbe4513:~$ ./bash_split_string_examples.sh
Choose an example to run:
1. Example 1: Split string by space
2. Example 2: Split string by comma
3. Example 3: Split string using colon
4. Example 4: Split string by 'Learn'
5. Example 5: Split string using tr command
Enter the number of the example you want to run (1-5): 3
Example 3: Split string without using $IFS
Enter any string separated by colon(:): sethu:madav:s
Split string:
sethu
madav
s

ssethumadav@019c021bbbe4513:~$
```

The screenshot shows a Windows terminal window with a black background. The command `./bash_split_string_examples.sh` is run, followed by the selection of Example 3. The user enters the string `sethu:madav:s` and the output shows the string split into three separate lines: `sethu`, `madav`, and `s`. The desktop taskbar at the bottom shows various application icons like File Explorer, Edge, and File Manager.

4. split string by 'learn'

```
ssethumadav@019c021bbbe4513:~$ ./bash_split_string_examples.sh
Choose an example to run:
1. Example 1: Split string by space
2. Example 2: Split string by comma
3. Example 3: Split string using colon
4. Example 4: Split string by 'Learn'
5. Example 5: Split string using tr command
Enter the number of the example you want to run (1-5): 4
Example 4: Split string by 'Learn'
Split string:
We
Welcome
You
On
Javatpoint
ssethumadav@019c021bbbe4513:~$
```

The screenshot shows a Windows terminal window with a black background. The command `./bash_split_string_examples.sh` is run, followed by the selection of Example 4. The user enters the string `We Welcome You On Javatpoint` and the output shows the string split into five lines: `We`, `Welcome`, `You`, `On`, and `Javatpoint`. The desktop taskbar at the bottom shows various application icons like File Explorer, Edge, and File Manager.

5. split string using tr command

```
ssethumadav@019c021bbbe4513:~$ ./bash_split_string_examples.sh
Choose an example to run:
1. Example 1: Split string by space
2. Example 2: Split string by comma
3. Example 3: Split string using colon
4. Example 4: Split string by 'Learn'
5. Example 5: Split string using tr command
Enter the number of the example you want to run (1-5): 5
Example 5: Split string using tr command
Split string:
We
welcome
you
on
javatpoint.
ssethumadav@019c021bbbe4513:~$
```

The screenshot shows a Windows terminal window with a black background. It displays the output of a bash script named `bash_split_string_examples.sh`. The user has selected Example 5, which demonstrates splitting a string using the `tr` command. The terminal window includes a taskbar at the bottom with various icons like File Explorer, Edge, and Task View.

BASH FIND STRING EXAMPLES:

Code for example 1 to 5:

```
GNU nano 7.2                                bash_find_string.sh
#!/bin/bash

# Function to get string length using the '#' symbol
example1() {
    echo "Enter a string:"
    read str
    length=${#str}
    echo "Length of '$str' is $length"
}

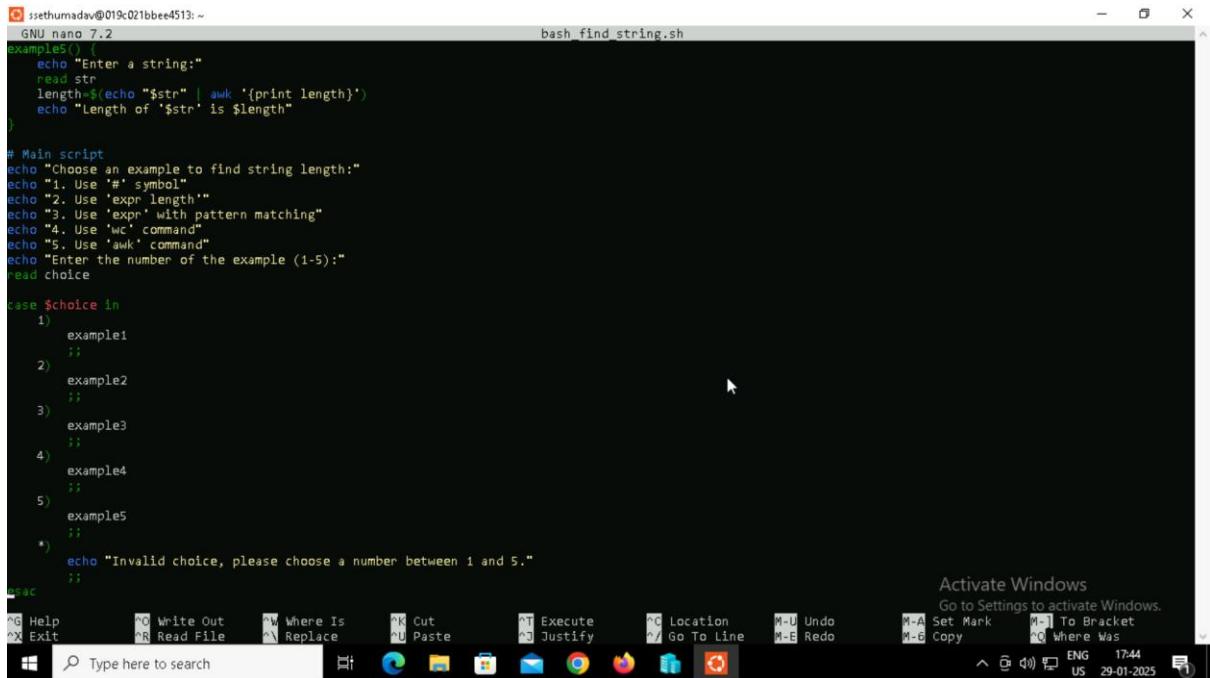
# Function to get string length using 'expr' command
example2() {
    echo "Enter a string:"
    read str
    length=$(expr length "$str")
    echo "Length of '$str' is $length"
}

# Function to get string length using 'expr' with pattern matching
example3() {
    echo "Enter a string:"
    read str
    length=$(expr "$str" : '.')
    echo "Length of '$str' is $length"
}

# Function to get string length using 'wc' command
example4() {
    echo "Enter a string:"
    read str
    length=$(echo "$str" | wc -c)
    echo "Length of '$str' is $length"
}

# Function to get string length using 'awk' command
example5() {
    echo "Enter a string:"
    read str
}
```

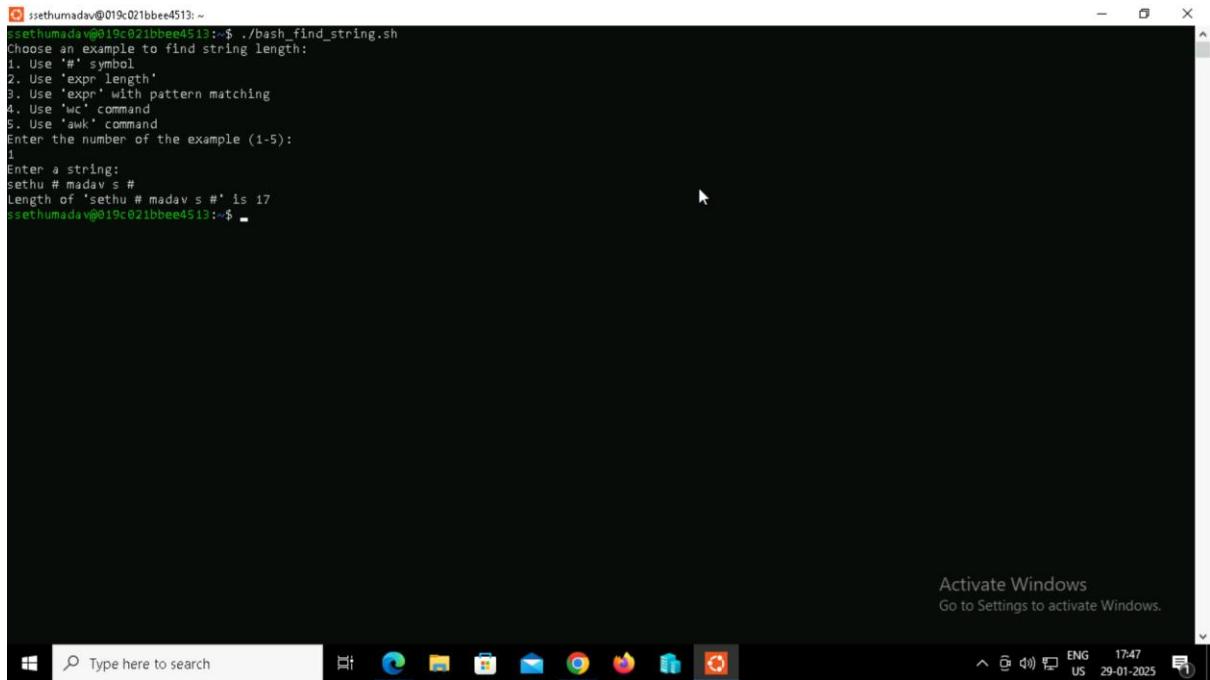
The screenshot shows a Windows terminal window with a black background. It displays a bash script named `bash_find_string.sh` containing five functions: `example1`, `example2`, `example3`, `example4`, and `example5`. Each function prompts the user for a string and prints its length. The terminal window includes a taskbar at the bottom with various icons like File Explorer, Edge, and Task View.



```
ssethumadav@019c021bbbe4513:~  
GNU nano 7.2  
examples() {  
    echo "Enter a string:"  
    read str  
    length=$(echo "$str" | awk '{print length}')  
    echo "Length of '$str' is $length"  
}  
  
# Main script  
echo "Choose an example to find string length:"  
echo "1. Use '#' symbol"  
echo "2. Use 'expr' length"  
echo "3. Use 'expr' with pattern matching"  
echo "4. Use 'wc' command"  
echo "5. Use 'awk' command"  
echo "Enter the number of the example (1-5):"  
read choice  
  
case $choice in  
    1)  
        example1  
        ;;  
    2)  
        example2  
        ;;  
    3)  
        example3  
        ;;  
    4)  
        example4  
        ;;  
    5)  
        example5  
        ;;  
    *)  
        echo "Invalid choice, please choose a number between 1 and 5."  
        ;;  
esac  
^G Help ^O Write Out ^W Where Is ^X Cut ^T Execute ^C Location M-U Undo  
^R Read File ^P Replace ^U Paste ^J Justify ^Y Go To Line M-E Redo  
M-A Set Mark M-L To Bracket M-B Copy M-Q Where Was  
Windows Type here to search ENG 17:44 US 29-01-2025
```

Outputs :

Example 1: using # symbol



```
ssethumadav@019c021bbbe4513:~$ ./bash_find_string.sh  
Choose an example to find string length:  
1. Use '#' symbol  
2. Use 'expr' length  
3. Use 'expr' with pattern matching  
4. Use 'wc' command  
5. Use 'awk' command  
Enter the number of the example (1-5):  
1  
Enter a string:  
sethu # madav s #  
Length of 'sethu # madav s #' is 17  
ssethumadav@019c021bbbe4513:~$
```

Activate Windows
Go to Settings to activate Windows.

Windows Type here to search ENG 17:47 US 29-01-2025

Example 2: using expr length

```
ssethumadav@019c021bbe4513:~$ ./bash_find_string.sh
Choose an example to find string length:
1. Use '#' symbol
2. Use 'expr length'
3. Use 'expr' with pattern matching
4. Use 'wc' command
5. Use 'awk' command
Enter the number of the example (1-5):
2
Enter a string:
happy man in happy world
Length of 'happy man in happy world' is 24
ssethumadav@019c021bbe4513:~$
```

The screenshot shows a Windows desktop environment. A terminal window is open in the center, displaying a shell script execution. The script asks for an example to find string length, and the user selects option 2 (using 'expr length'). The script then prompts for a string, which is entered as 'happy man in happy world'. The output shows the length of the string is 24. The desktop taskbar at the bottom includes icons for File Explorer, Edge browser, File Manager, Mail, and others. The system tray shows network, battery, and clock information.

Example 3:using expr with pattern

```
ssethumadav@019c021bbe4513:~$ ./bash_find_string.sh
Choose an example to find string length:
1. Use '#' symbol
2. Use 'expr length'
3. Use 'expr' with pattern matching
4. Use 'wc' command
5. Use 'awk' command
Enter the number of the example (1-5):
3
Enter a string:
hi hello how are you?happy?^[[D^[[D^[[D
Length of 'hi hello how are you?happy?^[[D^[[D^[[D' is 36
ssethumadav@019c021bbe4513:~$
```

This screenshot is identical to the previous one, showing the same terminal session and desktop environment. The user has selected option 3 (using 'expr' with pattern matching). The input string contains a pattern with control codes (including ^[[D) and question marks, and the output shows its length as 36. The desktop interface remains the same with the taskbar and system tray visible.

Example 4: using wc command

```
sethumadav@019c021bbe4513:~$ ./bash_find_string.sh
Choose an example to find string length:
1. Use '#' symbol
2. Use 'expr' length
3. Use 'expr' with pattern matching
4. Use 'wc' command
5. Use 'awk' command
Enter the number of the example (1-5):
4
Enter a string:
sethu madav s
Length of 'sethu madav s' is 14
sethumadav@019c021bbe4513:~$
```

The screenshot shows a Windows terminal window with a black background and white text. It displays the output of a shell script that asks for an example to find string length and then prompts for a string. The user enters 'sethu madav s' and the script outputs its length as 14. The terminal window has a title bar, a scroll bar on the right, and a taskbar at the bottom with various icons.

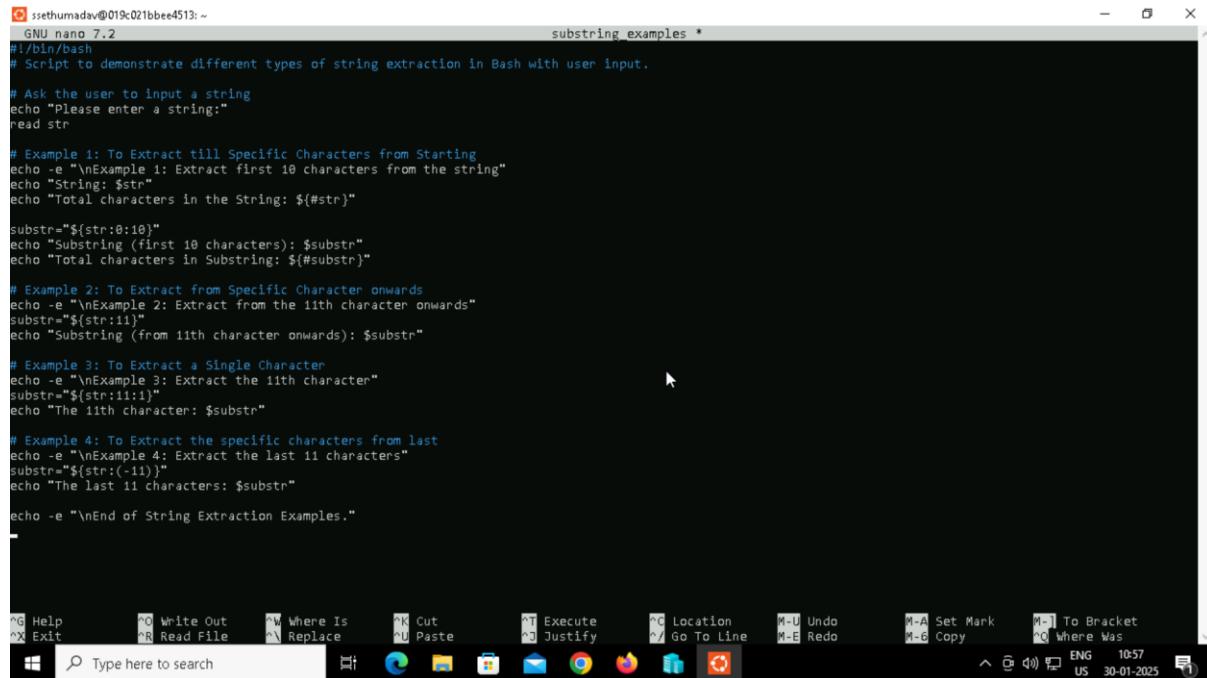
Example 5: using awk command

```
sethumadav@019c021bbe4513:~$ ./bash_find_string.sh
Choose an example to find string length:
1. Use '#' symbol
2. Use 'expr' length
3. Use 'expr' with pattern matching
4. Use 'wc' command
5. Use 'awk' command
Enter the number of the example (1-5):
5
Enter a string:
hi happy,nice to meet you.
Length of 'hi happy,nice to meet you.' is 26
sethumadav@019c021bbe4513:~$
```

The screenshot shows a Windows terminal window with a black background and white text. It displays the output of a shell script that asks for an example to find string length and then prompts for a string. The user enters 'hi happy,nice to meet you.' and the script outputs its length as 26. The terminal window has a title bar, a scroll bar on the right, and a taskbar at the bottom with various icons.

Bash Substring examples:

code for examples 1 to 4:



```
ssethumadav@019c021bbe4513:~          substring_examples *
GNU nano 7.2
#!/bin/bash
# Script to demonstrate different types of string extraction in Bash with user input.

# Ask the user to input a string
echo "Please enter a string:"
read str

# Example 1: To Extract till Specific Characters from Starting
echo -e "\nExample 1: Extract first 10 characters from the string"
echo "String: $str"
echo "Total characters in the String: ${#str}"

substr="${str:0:10}"
echo "Substring (first 10 characters): $substr"
echo "Total characters in Substring: ${#substr}"

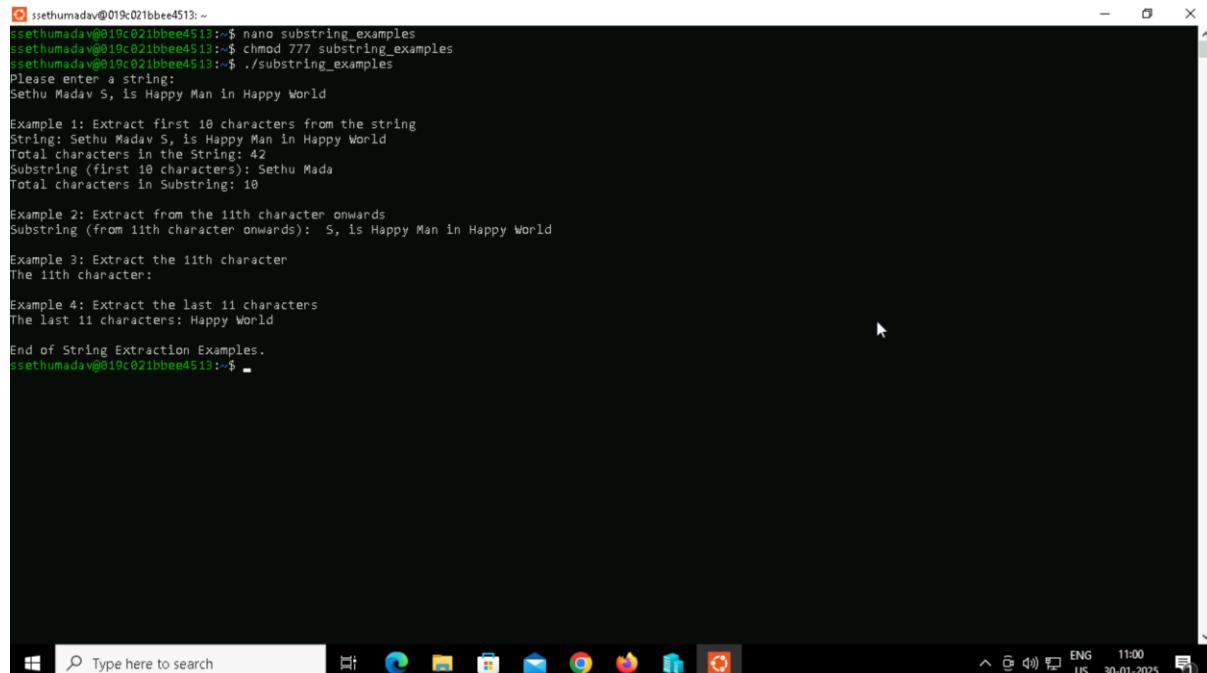
# Example 2: To Extract from Specific Character onwards
echo -e "\nExample 2: Extract from the 11th character onwards"
substr="${str:11}"
echo "Substring (from 11th character onwards): $substr"

# Example 3: To Extract a Single Character
echo -e "\nExample 3: Extract the 11th character"
substr="${str:11:1}"
echo "The 11th character: $substr"

# Example 4: To Extract the specific characters from last
echo -e "\nExample 4: Extract the last 11 characters"
substr="${str:(-11)}"
echo "The last 11 characters: $substr"

echo -e "\nEnd of String Extraction Examples."
```

Output:



```
ssethumadav@019c021bbe4513:~          substring_examples *
ssethumadav@019c021bbe4513:~$ nano substring_examples
ssethumadav@019c021bbe4513:~$ chmod 777 substring_examples
ssethumadav@019c021bbe4513:~$ ./substring_examples
Please enter a string:
Sethu Madav S, is Happy Man in Happy World

Example 1: Extract first 10 characters from the string
String: Sethu Madav S, is Happy Man in Happy World
Total characters in the String: 42
Substring (first 10 characters): Sethu Mada
Total characters in Substring: 10

Example 2: Extract from the 11th character onwards
Substring (from 11th character onwards): S, is Happy Man in Happy World

Example 3: Extract the 11th character
The 11th character: S

Example 4: Extract the last 11 characters
The last 11 characters: Happy World

End of String Extraction Examples.
ssethumadav@019c021bbe4513:~
```

Bash Concatenate String:

Example 1: Write Variables Side by Side

Example 2: Using Double Quotes

Example 3: Using Append Operator with Loop

Example 4: Using the Printf Function

Example 5: Using Literal Strings

Example 6: Using Underscore

Example 7: Using any Character

Code:for example 1 to 7:

```
ssethurnadav@019c021bb6e4513: ~
GNU nano 7.2
concatenate string
#!/bin/bash
# Script to demonstrate different types of string concatenation in Bash with user input.

# Example 1: Concatenate strings side by side
echo "Example 1: Concatenate strings side by side"
echo "Enter first string:"
read str1
echo "Enter second string:"
read str2

# Combine the two strings
str3="$str1$str2"
echo "Combined string: $str3"

# Example 2: Using double quotes for string concatenation
echo -e "\nExample 2: Using double quotes for string concatenation"
echo "Enter a string:"
read str
echo "$str on Javatpoint."

# Example 3: Using append operator with a loop
echo -e "\nExample 3: Using append operator with a loop"
echo "Enter a list of programming languages (e.g., java python C C++):"
read -a lang_list # This will read the input as an array

# Initialize an empty string
lang=""

# Loop through the list and append to the string
for value in "${lang_list[@]}"; do
    lang+="$value "
done

echo "Programming languages: $lang"

# Example 4: Using printf function
echo -e "\nExample 4: Using printf for string concatenation"
[ Read 71 lines ]
[ Execute ] [ Cut ] [ Paste ] [ Replace ] [ Justify ] [ Go To Line ] [ Undo ] [ Redo ] [ Set Mark ] [ To Bracket ]
[ Read File ] [ Write Out ] [ Where Is ] [ Help ] [ Exit ] [ Copy ] [ Where Was ]
Type here to search ENG 12-27
10:00 AM 30-01-2025
```

```
# Example 4: Using printf function
echo -e "\nExample 4: Using printf for string concatenation"
echo "Enter the first part of the string:"
read str
printf -v new_str "$str to Javatpoint."
echo "Concatenated string: $new_str"

# Example 5: Using literal strings
echo -e "\nExample 5: Using literal strings for concatenation"
echo "Enter the first string:"
read str
newstr="${str} Javatpoint."
echo "Concatenated string: $newstr"

# Example 6: Using underscore for concatenation
echo -e "\nExample 6: Using underscore for concatenation"
echo "Enter first string:"
read str1
echo "Enter second string:"
read str2
echo "${str1}_${str2}"

# Example 7: Using any character for concatenation with user input
echo -e "\nExample 7: Using any character for concatenation"
echo "Enter your first name:"
read name
echo "Enter your state:"
read state
echo "Enter your age:"
read age
combine="$name,$state,$age"
echo "Name, State, Age: $combine"
echo -e "\nEnd of String Concatenation Examples."
```

Output:

```
ssethumadav@019c021bbe4513:~$ nano concatenate_string
ssethumadav@019c021bbe4513:~$ chmod 777 concatenate_string
ssethumadav@019c021bbe4513:~$ ./concatenate_string
Please enter a string:
Sethumadav is HappymaninHappyWorld!!!!
Example 1: Extract first 10 characters from the string
String: Sethumadav is HappymaninHappyWorld!!!!
Total characters in the String: 38
Substring (first 10 characters): Sethumadav
Total characters in Substring: 10
Example 2: Extract from the 11th character onwards
Substring (from 11th character onwards): is HappymaninHappyWorld!!!!
Example 3: Extract the 11th character
The 11th character: i
Example 4: Extract the last 11 characters
The last 11 characters: pyWorld!!!!
End of String Extraction Examples.
ssethumadav@019c021bbe4513:~$ ./concatenate_string
Please enter a string:
sethu
Example 1: Extract first 10 characters from the string
String: sethu
Total characters in the String: 5
Substring (first 10 characters): sethu
Total characters in Substring: 5
Example 2: Extract from the 11th character onwards
Substring (from 11th character onwards):
Example 3: Extract the 11th character
The 11th character:
Example 4: Extract the last 11 characters
The last 11 characters:
```

```
ssethumadav@019c021bbe4513:~  
Example 2: Using double quotes for string concatenation  
Enter a string:  
sethumadav  
sethumadav on Javatpoint.  
  
Example 3: Using append operator with a loop  
Enter a list of programming languages (e.g., java python c C++):  
java python c c++ ruby  
Programming languages: java python c c++ ruby  
  
Example 4: Using printf for string concatenation  
Enter the first part of the string:  
sethu  
Concatenated string: sethu to Javatpoint.  
  
Example 5: Using literal strings for concatenation  
Enter the first string:  
sethu  
Concatenated string: sethu Javatpoint.  
  
Example 6: Using underscore for concatenation  
Enter first string:  
hiI hello  
Enter second string:  
how r u  
hiI hello_ how r u  
  
Example 7: Using any character for concatenation  
Enter your first name:  
s  
Enter your state:  
tn  
Enter your age:  
22  
Name, State, Age: s,tn,22  
  
End of String Concatenation Examples.  
ssethumadav@019c021bbe4513:~$
```



Bash_Functions_examples:

Code (example 1 to 7):

```
ssethumadav@019c021bbe4513: ~                               functions_inbash_example
GNU nano 7.2
#!/bin/bash
# Script to demonstrate functions, argument passing, variable scope, return values, and command overriding.

# Method 1: Defining a function and calling it
echo "Example 1: Method 1"
JTP () {
    echo 'Welcome to Javatpoint.'
}

JTP

# Method 2: Defining a function using the 'function' keyword
echo -e "\nExample 2: Method 2"
function JTP () {
    echo 'Welcome to Javatpoint.'
}

JTP

# Passing Arguments to Functions
echo -e "\nExample 3: Passing Arguments to Functions"
function function_arguments() {
    echo "Argument 1: $1"
    echo "Argument 2: $2"
    echo "Argument 3: $3"
    echo "Argument 4: $4"
    echo "Argument 5: $5"
}

# Get user input for function arguments
echo "Enter 5 Words separated by space (e.g., We welcome you on Javatpoint):"
read word1 word2 word3 word4 word5

# call the function with user-provided arguments
function_arguments "$word1" "$word2" "$word3" "$word4" "$word5"

# Variable Scope (Global and Local)
ssethumadav@019c021bbe4513: ~                               functions_inbash_example
GNU nano 7.2
# Variable Scope (Global and Local)
echo -e "\nExample 4: Variable Scope"
echo "Enter a value for global variable v1:"
read v1_input
echo "Enter a value for global variable v2:"
read v2_input

v1="$v1_input"
v2="$v2_input"

my_var () {
    local v1='C'
    v2='D'
    echo "Inside Function:"
    echo "v1 is $v1."
    echo "v2 is $2."
}

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.

# Return Values from Functions
echo -e "\nExample 5: Return Values from Functions"
print_it () {
    echo "Hello $1"
    return 5
}

echo "Enter a name for the function to greet:"
read name

print_it $name
```

```
ssethumadav@019c021bbe4513: ~                               functions_inbash_example
GNU nano 7.2
# Return Values from Functions
echo -e "\nExample 5: Return Values from Functions"
print_it () {
    echo "Hello $1"
    return 5
}

echo "Enter a name for the function to greet:"
read name

print_it $name
```

```
ssethumadav@019c021bbe4513:~  
GNUnano 7.2  
my_var  
  
echo "After Executing the Function"  
echo "v1 is $v1."  
echo "v2 is $v2."  
  
# Return Values from Functions  
echo -e "\nExample 5: Return Values from Functions"  
print_it () {  
    echo "Hello $1"  
    return 5  
}  
  
echo "Enter a name for the function to greet:"  
read name  
print_it "$name"  
echo "The previous function returned a value of $?"  
  
# Returning a value using echo  
echo -e "\nExample 6: Returning Value using Echo"  
print_it () {  
    local my_greet="Welcome to Javatpoint."  
    echo "$my_greet"  
}  
  
my_greet=$(print_it)  
echo "$my_greet"  
  
# Overriding Commands  
echo -e "\nExample 7: Overriding Commands"  
echo "Enter a message to override the echo command with timestamp:"  
read user_message  
  
echo "$user_message"  
  
-  
^G Help ^O Write Out ^W Where Is ^X Exit ^R Read File ^U Paste ^T Execute ^J Justify ^C Location ^Y Go To Line M-U Undo M-E Redo M-A Set Mark M-B Copy M-Q To Bracket M-Q Where Was  
Type here to search ENG 12:42 US 30-01-2025
```

Outputs:

```
ssethumadav@019c021bbe4513:~  
ssethumadav@019c021bbe4513:~$ nano functions_inbash_example  
ssethumadav@019c021bbe4513:~$ chmod 777 functions_inbash_example  
ssethumadav@019c021bbe4513:~$ ./functions_inbash_example  
Example 1: Method 1  
Welcome to Javatpoint.  
  
Example 2: Method 2  
Welcome to Javatpoint.  
  
Example 3: Passing Arguments to Functions  
Enter 5 words separated by space (e.g., We welcome you on Javatpoint):  
sethu is so happy world  
Argument 1: sethu  
Argument 2: is  
Argument 3: so  
Argument 4: happy  
Argument 5: world  
  
Example 4: Variable Scope  
Enter a value for global variable v1:  
11  
Enter a value for global variable v2:  
66  
Before Executing the Function  
v1 is 11.  
v2 is 66.  
Inside Function:  
v1 is C.  
v2 is D.  
After Executing the Function  
v1 is 11.  
v2 is D.  
  
Example 5: Return Values from Functions  
Enter a name for the function to greet:  
welcome  
Hello welcome  
The previous function returned a value of 5  
  
Example 6: Returning Value using Echo  
Welcome to Javatpoint.  
-  
Type here to search ENG 12:39 US 30-01-2025
```

```
○ Select ssethumadav@019c021bbe4513: ~
v1 is C.
v2 is D.
After Executing the Function
v1 is 11.
v2 is D.

Example 5: Return Values from Functions
Enter a name for the function to greet:
welcome
Hello welcome
The previous function returned a value of 5

Example 6: Returning Value using Echo
Welcome to Javatpoint.

Example 7: Overriding Commands
Enter a message to override the echo command with timestamp:
dont do
dont do
ssethumadav@019c021bbe4513:~$
```

Windows Taskbar: Type here to search, Start button, File Explorer, Mail, Photos, OneDrive, Edge, Google Chrome, Firefox, Task View, Refresh, ENG US 12:39 30-01-2025

Array Operations: Examples

1. Print an Element by Index
2. Print All Elements
3. Print Keys (Indexes) of the Array
4. Find the Length of the Array
5. Loop Through the Array Using @ Expansion
6. Loop Through the Array Using a C-style for Loop
7. Add a New Element to the Array
8. Update an Existing Element
9. Delete an Element from the Array
10. Delete the Entire Array
11. Slice the Array

Code:

The screenshot shows a Windows terminal window titled "array_operations_example". The terminal is running a bash script that demonstrates various array operations. The script includes functions for printing elements by index, printing all elements, printing keys, finding the length, looping through arrays using @ expansion and a C-style for loop, adding new elements, updating existing elements, deleting elements, and deleting the entire array. The terminal window has a standard Windows title bar and taskbar at the bottom.

```
ssethumadav@019c021bbe4513: ~
GNU nano 7.2
#!/bin/bash
#Script to demonstrate Array operations with user input

# Function to print an element of the array with a specific index
print_element_by_index() {
    echo -e "\nEnter the index of the element you want to print:"
    read index
    echo "The element at index $index is: ${example_array[$index]}"
}

# Function to print all elements of the array
print_all_elements() {
    echo -e "\nAll elements in the array: ${example_array[@]}"
}

# Function to print keys (indexes) of the array
print_keys() {
    echo -e "\nThe keys (indexes) of the array are: ${!example_array[@]}"
}

# Function to find the length of the array
array_length() {
    echo -e "\nThe array contains ${#example_array[@]} elements."
}

# Function to loop through array with a for loop
loop_through_array() {
    echo -e "\nLooping through the array with @ expansion:"
    for i in "${example_array[@]}"; do
        echo "$i"
    done
}

# Function to loop through array with C-style loop
loop_through_array_c_style() {
    echo -e "\nLooping through the array with a C-style for loop:"
    length=${#example_array[@]}
}
```

```
ssethumadav@019c021bbef4513: ~
GNU nano 7.2                                     array_operations_example

length ${example_array[@]}
for (( i=0; i < $length; i++ )); do
    echo "$i: ${example_array[$i]}"
done
}

# Function to add a new element to the array
add_element() {
    echo -e "\nEnter a new element to add to the array:"
    read new_element
    example_array+=("$new_element")
    echo "Updated array: ${example_array[@]}"
}

# Function to update an element in the array
update_element() {
    echo -e "\nEnter the index of the element to update:"
    read index
    echo -e "\nEnter the new value for element at index $index:"
    read new_value
    example_array[$index]="$new_value"
    echo "Updated array: ${example_array[@]}"
}

# Function to delete an element from the array
delete_element() {
    echo -e "\nEnter the index of the element to delete:"
    read index
    unset example_array[$index]
    echo "Updated array: ${example_array[@]}"
}

# Function to delete the entire array
delete_entire_array() {
    unset example_array
    echo -e "\nThe entire array has been deleted."
}

# Help menu
Help
Exit

# File operations
Write Out
Read File
Where Is
Replace

# Edit operations
Cut
Paste
Execute
Justify

# Navigation
Location
Go To Line

# Undo/Redo
Undo
Redo

# Mark
Set Mark
Copy

# Windows
To Bracket
Where Was

# System
Activate Windows
Go to Settings to activate Windows.

# Bottom bar
Type here to search
File Explorer
Recycle Bin
Task View
Start
Search
System
Battery
Network
Volume
Language
14:33
US
30-01-2025
```

```
ssethunadav@019c021bbe4513:~  
GNU nano 7.2  
# Show the initial array  
echo -e "\nInitial Array: ${example_array[@]}"  
  
# Interactive menu for user input  
while true; do  
    echo -e "\nChoose an option:  
    echo "1. Print an element by index"  
    echo "2. Print all elements"  
    echo "3. Print keys (indexes)"  
    echo "4. Find the length of the array"  
    echo "5. Loop through the array"  
    echo "6. Loop through the array (C-style)"  
    echo "7. Add a new element"  
    echo "8. Update an existing element"  
    echo "9. Delete an element"  
    echo "10. Delete the entire array"  
    echo "11. Slice the array"  
    echo "12. Exit"  
  
    read -p "Enter your choice (1-12): " choice  
  
    case $choice in  
        1) print_element_by_index ;;  
        2) print_all_elements ;;  
        3) print_keys ;;  
        4) array_length ;;  
        5) loop_through_array ;;  
        6) loop_through_array_c_style ;;  
        7) add_element ;;  
        8) update_element ;;  
        9) delete_element ;;  
        10) delete_entire_array ;;  
        11) slice_array ;;  
        12) echo "Exiting..."; exit 0 ;;  
        *) echo "Invalid choice, please try again.";;  
    esac  
done  
  
^G Help      ^O Write Out    ^W Where Is     ^K Cut          ^X Execute      ^L Location     ^U Undo  
^Q Exit      ^R Read File    ^N Replace     ^P Paste         ^J Justify      ^C Go To Line   ^M-E Redo  
Activate Windows  
Go to Settings to activate Windows.  
Set Mark      ^W To Bracket  
M-G Copy      ^O Where Was  
ENG 1443  
US 30-01-2025
```

Outputs:

```
ssethumadav@019c021bbe4513:~$ nano array_operations_example
ssethumadav@019c021bbe4513:~$ nano array_operations_example
ssethumadav@019c021bbe4513:~$ ./array_operations_example
Array Operations Script

Initial Array: Java Python HTML CSS JavaScript

Choose an option:
1. Print an element by index
2. Print all elements
3. Print keys (indexes)
4. Find the length of the array
5. Loop through the array
6. Loop through the array (C-style)
7. Add a new element
8. Update an existing element
9. Delete an element
10. Delete the entire array
11. Slice the array
12. Exit
Enter your choice (1-12): 1

Enter the index of the element you want to print:
2
The element at index 2 is: HTML

Choose an option:
1. Print an element by index
2. Print all elements
3. Print keys (indexes)
4. Find the length of the array
5. Loop through the array
6. Loop through the array (C-style)
7. Add a new element
8. Update an existing element
9. Delete an element
10. Delete the entire array
11. Slice the array
12. Exit
Enter your choice (1-12): 2

Windows Taskbar: Type here to search, File, Internet Explorer, File Explorer, Mail, Google Chrome, Mozilla Firefox, Edge, Task View, Start button, Date/Time: 14:51, Language: ENG US, Date: 30-01-2025
```

```
ssethumadav@019c021bbe4513:~$ ./array_operations_example
11. Slice the array
12. Exit
Enter your choice (1-12): 2

All elements in the array: Java Python HTML CSS JavaScript

Choose an option:
1. Print an element by index
2. Print all elements
3. Print keys (indexes)
4. Find the length of the array
5. Loop through the array
6. Loop through the array (C-style)
7. Add a new element
8. Update an existing element
9. Delete an element
10. Delete the entire array
11. Slice the array
12. Exit
Enter your choice (1-12): 3

The keys (indexes) of the array are: 0 1 2 3 4

Choose an option:
1. Print an element by index
2. Print all elements
3. Print keys (indexes)
4. Find the length of the array
5. Loop through the array
6. Loop through the array (C-style)
7. Add a new element
8. Update an existing element
9. Delete an element
10. Delete the entire array
11. Slice the array
12. Exit
Enter your choice (1-12): 4

The array contains 5 elements.

Windows Taskbar: Type here to search, File, Internet Explorer, File Explorer, Mail, Google Chrome, Mozilla Firefox, Edge, Task View, Start button, Date/Time: 14:53, Language: ENG US, Date: 30-01-2025
```

```
ssethumadav@019c021bbe4513: ~
10. Delete the entire array
11. Slice the array
12. Exit
Enter your choice (1-12): 4
The array contains 5 elements.

Choose an option:
1. Print an element by index
2. Print all elements
3. Print keys (indexes)
4. Find the length of the array
5. Loop through the array
6. Loop through the array (C-style)
7. Add a new element
8. Update an existing element
9. Delete an element
10. Delete the entire array
11. Slice the array
12. Exit
Enter your choice (1-12): 5

Looping through the array with @ expansion:
Java
Python
HTML
CSS
JavaScript

Choose an option:
1. Print an element by index
2. Print all elements
3. Print keys (indexes)
4. Find the length of the array
5. Loop through the array
6. Loop through the array (C-style)
7. Add a new element
8. Update an existing element
9. Delete an element
10. Delete the entire array
11. Slice the array
12. Exit
Activate Windows
Go to Settings to activate Windows.

Windows taskbar: Type here to search, File, Start, Task View, Edge, File Explorer, Mail, Photos, Google Chrome, Firefox, File Explorer, Refresh, Date/Time: 14:53, Language: ENG US, Date: 30-01-2025
```

```
ssethumadav@019c021bbe4513: ~
Enter your choice (1-12): 6

Looping through the array with a C-style for loop:
0: Java
1: Python
2: HTML
3: CSS
4: JavaScript

Choose an option:
1. Print an element by index
2. Print all elements
3. Print keys (indexes)
4. Find the length of the array
5. Loop through the array
6. Loop through the array (C-style)
7. Add a new element
8. Update an existing element
9. Delete an element
10. Delete the entire array
11. Slice the array
12. Exit
Enter your choice (1-12): 7

Enter a new element to add to the array:
sethu
Updated array: Java Python HTML CSS JavaScript sethu

Choose an option:
1. Print an element by index
2. Print all elements
3. Print keys (indexes)
4. Find the length of the array
5. Loop through the array
6. Loop through the array (C-style)
7. Add a new element
8. Update an existing element
9. Delete an element
10. Delete the entire array
11. Slice the array
12. Exit
Activate Windows
Go to Settings to activate Windows.

Windows taskbar: Type here to search, File, Start, Task View, Edge, File Explorer, Mail, Photos, Google Chrome, Firefox, File Explorer, Refresh, Date/Time: 14:54, Language: ENG US, Date: 30-01-2025
```

```
ssethumadav@019c021bbe4513: ~
11. Slice the array
12. Exit
Enter your choice (1-12): 8

Enter the index of the element to update:
2
Enter the new value for element at index 2:
hi
Updated array: Java Python hi CSS JavaScript sethu

Choose an option:
1. Print an element by index
2. Print all elements
3. Print keys (indexes)
4. Find the length of the array
5. Loop through the array
6. Loop through the array (C-style)
7. Add a new element
8. Update an existing element
9. Delete an element
10. Delete the entire array
11. Slice the array
12. Exit
Enter your choice (1-12): 9

Enter the index of the element to delete:
3
Updated array: Java Python hi JavaScript sethu

Choose an option:
1. Print an element by index
2. Print all elements
3. Print keys (indexes)
4. Find the length of the array
5. Loop through the array
6. Loop through the array (C-style)
7. Add a new element
8. Update an existing element
9. Delete an element
10. Delete the entire array
11. Slice the array
Activate Windows
Go to Settings to activate Windows.

Windows Type here to search ENG 14:55 US 30-01-2025
```

```
ssethumadav@019c021bbe4513: ~
Enter your choice (1-12): 10

The entire array has been deleted.

Choose an option:
1. Print an element by index
2. Print all elements
3. Print keys (indexes)
4. Find the length of the array
5. Loop through the array
6. Loop through the array (C-style)
7. Add a new element
8. Update an existing element
9. Delete an element
10. Delete the entire array
11. Slice the array
12. Exit
Enter your choice (1-12): 11

Enter the starting index for slicing:
2
Enter the number of elements to slice:
4
Sliced array:

Choose an option:
1. Print an element by index
2. Print all elements
3. Print keys (indexes)
4. Find the length of the array
5. Loop through the array
6. Loop through the array (C-style)
7. Add a new element
8. Update an existing element
9. Delete an element
10. Delete the entire array
11. Slice the array
12. Exit
Enter your choice (1-12): 11

Enter the starting index for slicing:
Windows Type here to search ENG 14:55 US 30-01-2025
```

```
ssethumadav@019c021bbbe4513:~  
9. Update an existing element  
9. Delete an element  
10. Delete the entire array  
11. Slice the array  
12. Exit  
Enter your choice (1-12): 11  
Enter the starting index for slicing:  
1  
Enter the number of elements to slice:  
3  
Sliced array:  
Choose an option:  
1. Print an element by index  
2. Print all elements  
3. Print keys (indexes)  
4. Find the length of the array  
5. Loop through the array  
6. Loop through the array (C-style)  
7. Add a new element  
8. Update an existing element  
9. Delete an element  
10. Delete the entire array  
11. Slice the array  
12. Exit  
Enter your choice (1-12): 12  
Exiting...  
ssethumadav@019c021bbbe4513:~$
```

Activate Windows
Go to Settings to activate Windows.

