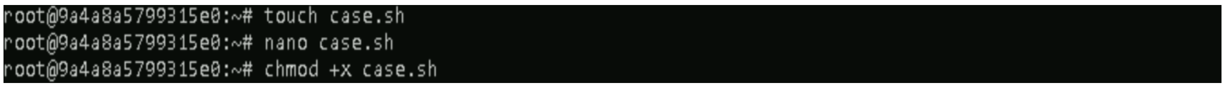
BASH SCRIPTING PROJECT

CASE STATEMENT

1.A Shell Program to define a simple scenario to demonstrate the use of the 'Case Statement'.

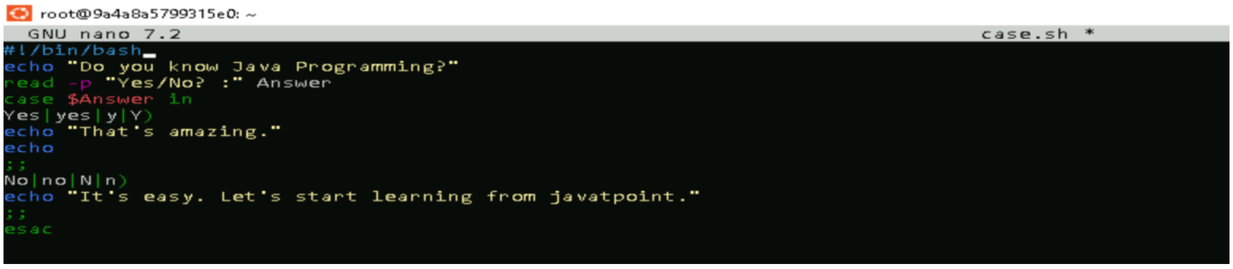
Step 1: Create an "case.sh" script file using touch command

Step 2: Create a nano file to write the code



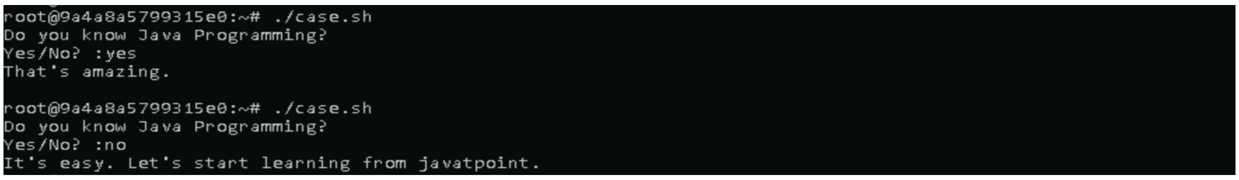
```
root@9a4a8a5799315e0:~# touch case.sh
root@9a4a8a5799315e0:~# nano case.sh
root@9a4a8a5799315e0:~# chmod +x case.sh
```

Step 3: Write the Code in nano case.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                  case.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
```

Step 4: Output



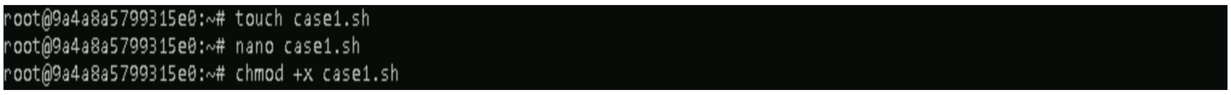
```
root@9a4a8a5799315e0:~# ./case.sh
Do you know Java Programming?
Yes/No? :yes
That's amazing.

root@9a4a8a5799315e0:~# ./case.sh
Do you know Java Programming?
Yes/No? :no
It's easy. Let's start learning from javatpoint.
```

2.A Shell Program to define a combined scenario to demonstrate the use of the 'Case Statement'.

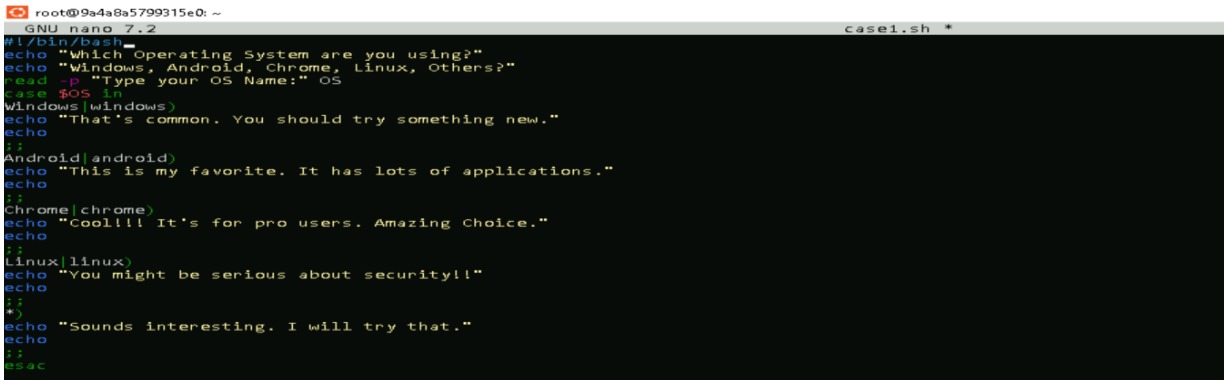
Step 1: Create an "case1.sh" script file using touch command

Step 2: Create a nano file to write the code



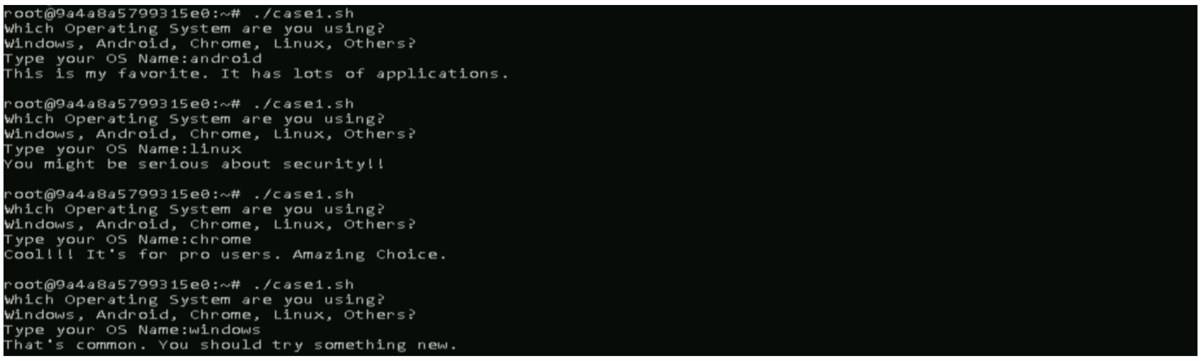
```
root@9a4a8a5799315e0:~# touch case1.sh
root@9a4a8a5799315e0:~# nano case1.sh
root@9a4a8a5799315e0:~# chmod +x case1.sh
```

Step 3: Write the Code in nano case1.sh.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                  case1.sh *
#!/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
```

Step 4: Output

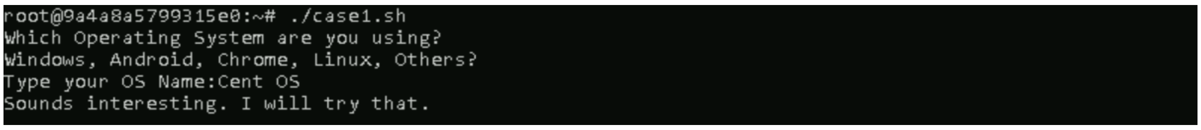


```
root@9a4a8a5799315e0:~# ./case1.sh
Which Operating System are you using?
Windows, Android, Chrome, Linux, Others?
Type your OS Name:android
This is my favorite. It has lots of applications.

root@9a4a8a5799315e0:~# ./case1.sh
Which Operating System are you using?
Windows, Android, Chrome, Linux, Others?
Type your OS Name:linux
You might be serious about security!!

root@9a4a8a5799315e0:~# ./case1.sh
Which Operating System are you using?
Windows, Android, Chrome, Linux, Others?
Type your OS Name:chrome
Cool!!! It's for pro users. Amazing Choice.

root@9a4a8a5799315e0:~# ./case1.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.
```



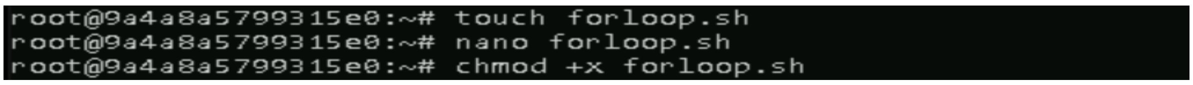
```
root@9a4a8a5799315e0:~# ./case1.sh
Which Operating System are you using?
Windows, Android, Chrome, Linux, Others?
Type your OS Name:Cent OS
Sounds interesting. I will try that.
```

FOR LOOP

3.A Shell Program to demonstrate the use of ‘For Loop’.

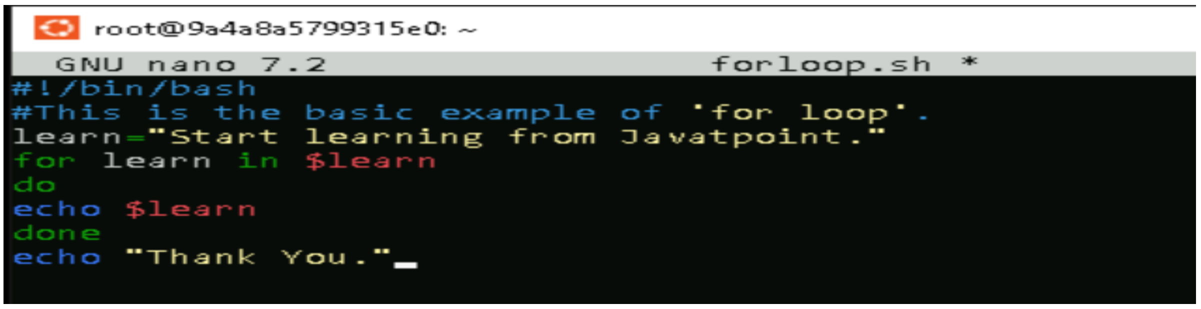
Step 1: Create an “forloop.sh” script file using touch command

Step 2: Create a nano file to write the code



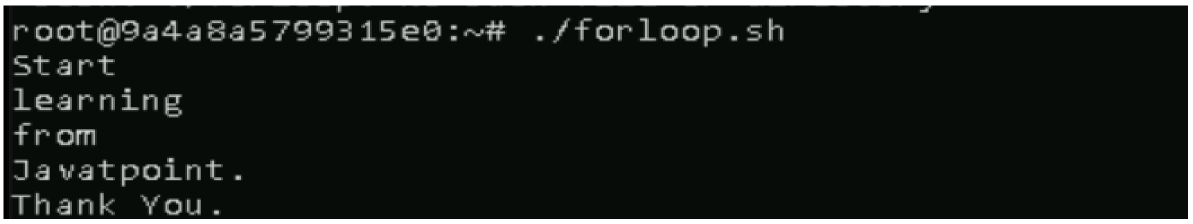
```
root@9a4a8a5799315e0:~# touch forloop.sh
root@9a4a8a5799315e0:~# nano forloop.sh
root@9a4a8a5799315e0:~# chmod +x forloop.sh
```

Step 3: Write the Code in nano forloop.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                    forloop.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."
```

Step 4: Output

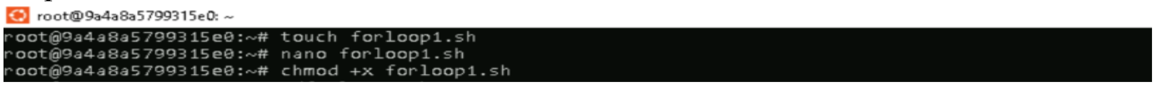


```
root@9a4a8a5799315e0:~# ./forloop.sh
Start
learning
from
Javatpoint.
Thank You.
```

4.A Shell Program to demonstrate the use of ‘For Loop’ to read a range.

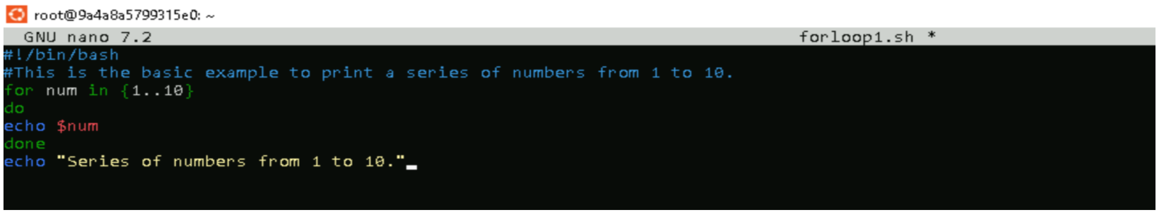
Step 1: Create an “forloop1.sh” script file using touch command

Step 2: Create a nano file to write the code



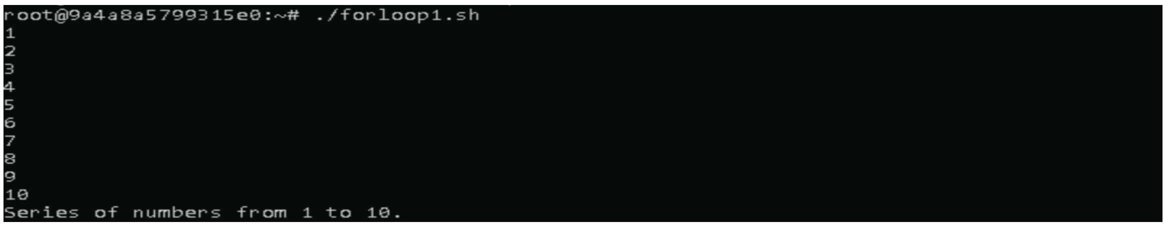
```
root@9a4a8a5799315e0: ~
root@9a4a8a5799315e0:~# touch forloop1.sh
root@9a4a8a5799315e0:~# nano forloop1.sh
root@9a4a8a5799315e0:~# chmod +x forloop1.sh
```

Step 3: Write the Code in nano forloop1.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                          forloop1.sh *
#!/bin/bash
#This is the basic example to print a series of numbers from 1 to 10.
for num in {1..10}
do
echo $num
done
echo "Series of numbers from 1 to 10."
```

Step 4: Output

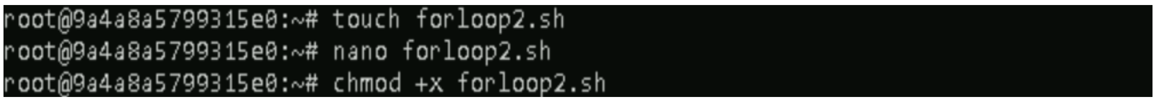


```
root@9a4a8a5799315e0:~# ./forloop1.sh
1
2
3
4
5
6
7
8
9
10
Series of numbers from 1 to 10.
```

5.A Shell Program to demonstrate the use of 'For Loop to read a range with Increment'.

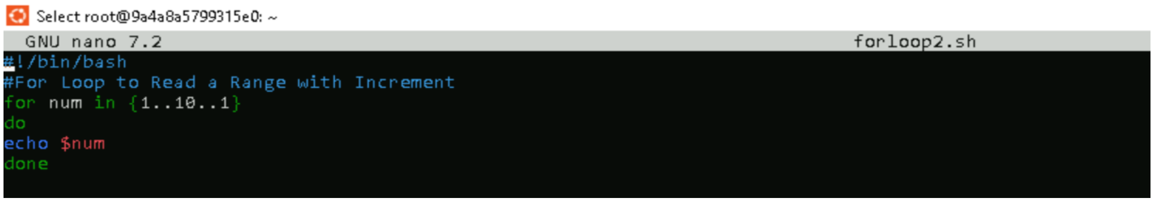
Step 1: Create an "forloop2.sh" script file using touch command

Step 2: Create a nano file to write the code



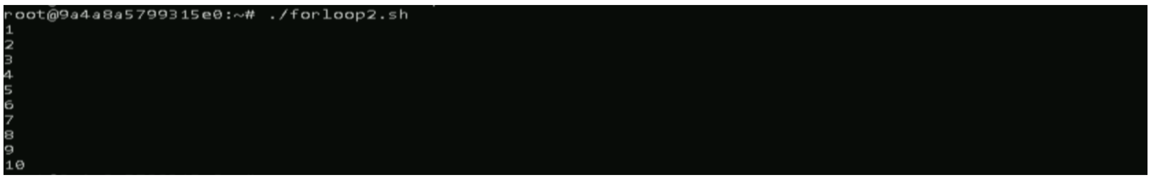
```
root@9a4a8a5799315e0:~# touch forloop2.sh
root@9a4a8a5799315e0:~# nano forloop2.sh
root@9a4a8a5799315e0:~# chmod +x forloop2.sh
```

Step 3: Write the Code in nano forloop2.sh.sh script file



```
Select root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                          forloop2.sh
#!/bin/bash
#For Loop to Read a Range with Increment
for num in {1..10..1}
do
echo $num
done
```

Step 4: Output

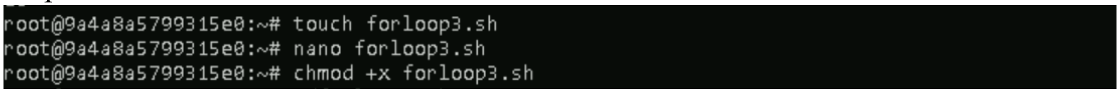


```
root@9a4a8a5799315e0:~# ./forloop2.sh
1
2
3
4
5
6
7
8
9
10
```

6.A Shell Program to demonstrate the use of 'For Loop to read a range with Decrement'.

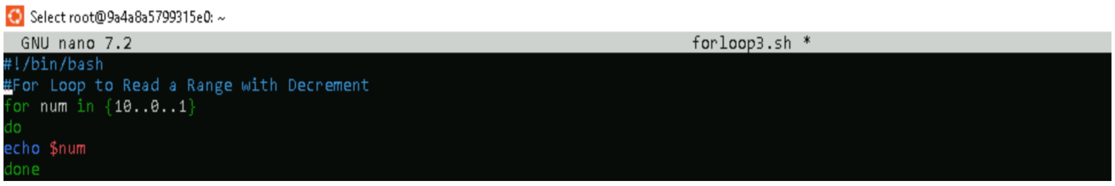
Step 1: Create an "forloop3.sh" script file using touch command

Step 2: Create a nano file to write the code



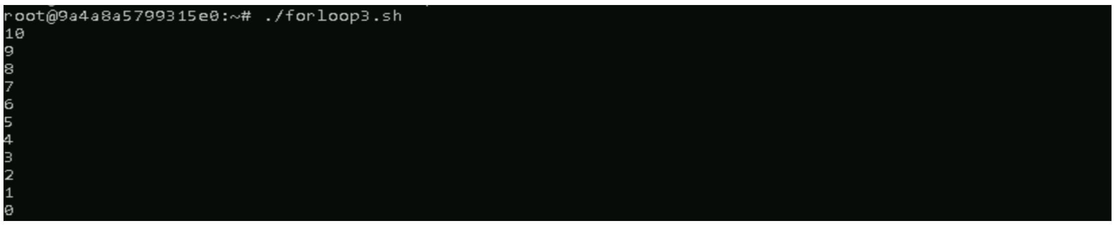
```
root@9a4a8a5799315e0:~# touch forloop3.sh
root@9a4a8a5799315e0:~# nano forloop3.sh
root@9a4a8a5799315e0:~# chmod +x forloop3.sh
```

Step 3: Write the Code in nano forloop3.sh script file



```
Select root@9a4a8a5799315e0: ~
  GNU nano 7.2                                              forloop3.sh *
#!/bin/bash
#For Loop to Read a Range with Decrement
for num in {10..0..1}
do
echo $num
done
```

Step 4: Output

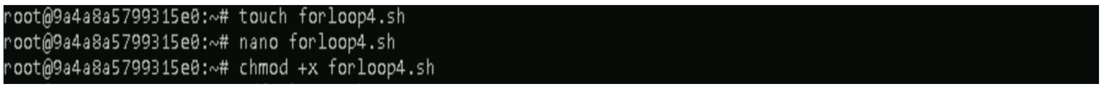


```
root@9a4a8a5799315e0:~# ./forloop3.sh
10
9
8
7
6
5
4
3
2
1
0
```

7.A Shell Program to demonstrate the use of 'For Loop' to iterate over elements of an array.

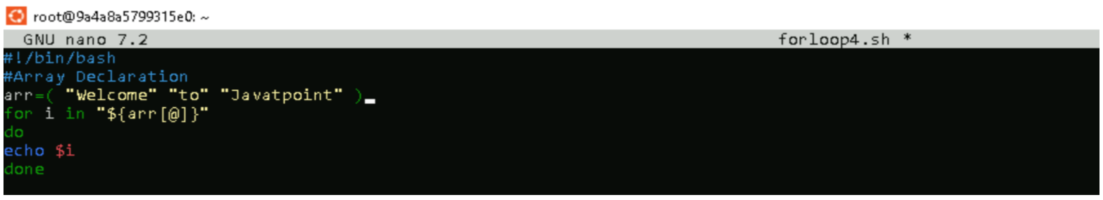
Step 1: Create an "forloop4.sh" script file using touch command

Step 2: Create a nano file to write the code



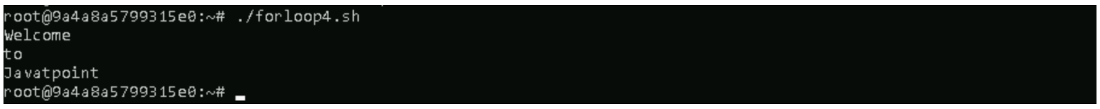
```
root@9a4a8a5799315e0:~# touch forloop4.sh
root@9a4a8a5799315e0:~# nano forloop4.sh
root@9a4a8a5799315e0:~# chmod +x forloop4.sh
```

Step 3: Write the Code in nano forloop4.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                              forloop4.sh *
#!/bin/bash
#Array Declaration
arr=( "Welcome" "to" "Javatpoint" )
for i in "${arr[@]}"
do
echo $i
done
```

Step 4: Output

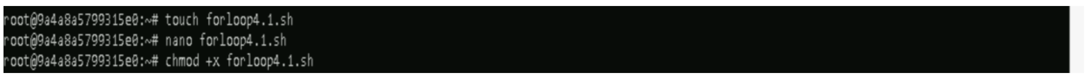


```
root@9a4a8a5799315e0:~# ./forloop4.sh
Welcome
to
Javatpoint
root@9a4a8a5799315e0:~#
```

8.A Shell Program to demonstrate the use of 'For Loop' to read white spaces in string as word separators.

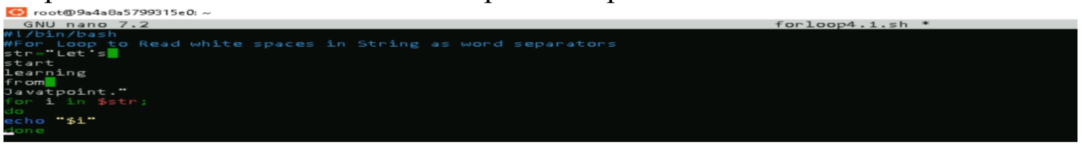
Step 1: Create an "forloop4.1.sh" script file using touch command

Step 2: Create a nano file to write the code



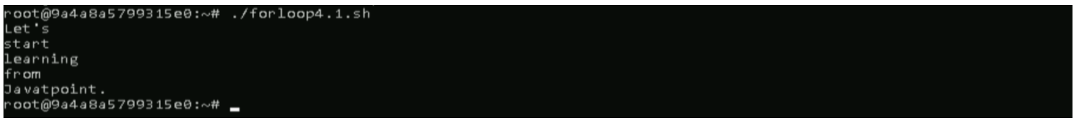
```
root@9a4a8a5799315e0:~# touch forloop4.1.sh
root@9a4a8a5799315e0:~# nano forloop4.1.sh
root@9a4a8a5799315e0:~# chmod +x forloop4.1.sh
```

Step 3: Write the Code in nano forloop4.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                              forloop4.1.sh *
#!/bin/bash
#For Loop to Read white spaces in String as word separators
str="Let's
start
learning
from
Javatpoint."
for i in $str;
do
echo "$i"
done
```

Step 4: Output

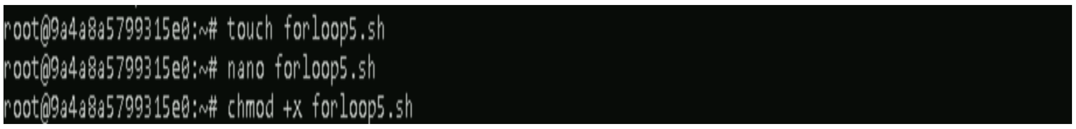


```
root@9a4a8a5799315e0:~# ./forloop4.1.sh
Let's
start
learning
from
Javatpoint.
root@9a4a8a5799315e0:~# _
```

9.A Shell Program to define ‘For Loop’ to read each line in string as a word.

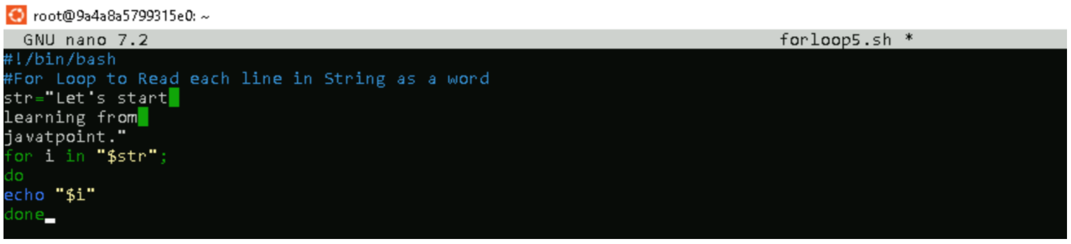
Step 1: Create an “forloop5.sh” script file using touch command

Step 2: Create a nano file to write the code



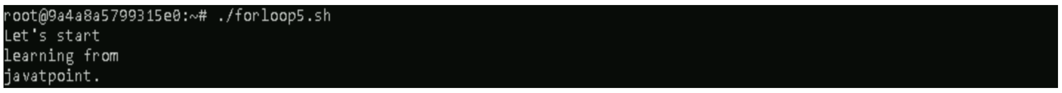
```
root@9a4a8a5799315e0:~# touch forloop5.sh
root@9a4a8a5799315e0:~# nano forloop5.sh
root@9a4a8a5799315e0:~# chmod +x forloop5.sh
```

Step 3: Write the Code in nano forloop5.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                forloop5.sh *
#!/bin/bash
#For Loop to Read each line in String as a word
str="Let's start
learning from
javatpoint."
for i in "$str";
do
echo "$i"
done
```

Step 4: Output

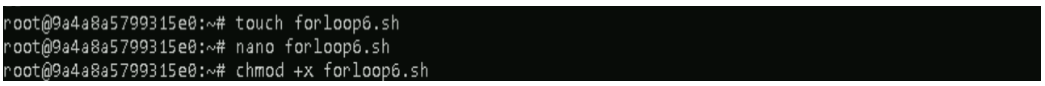


```
root@9a4a8a5799315e0:~# ./forloop5.sh
Let's start
learning from
javatpoint.
```

10.A Shell Program to define ‘For Loop’ to read three-expression.

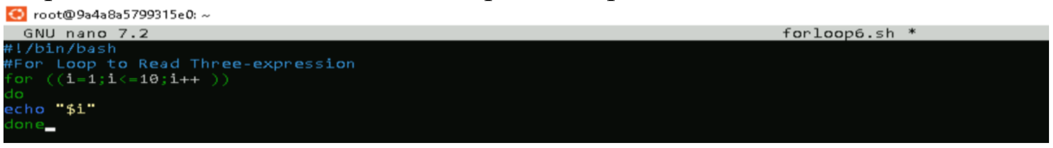
Step 1: Create an “forloop6.sh” script file using touch command

Step 2: Create a nano file to write the code



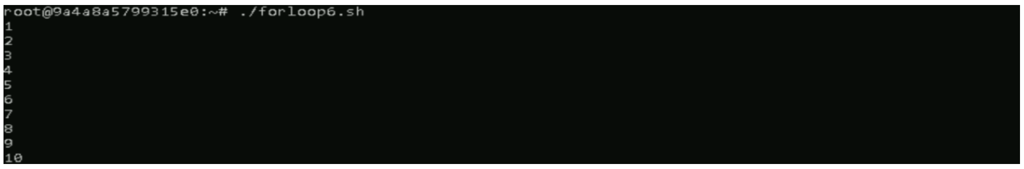
```
root@9a4a8a5799315e0:~# touch forloop6.sh
root@9a4a8a5799315e0:~# nano forloop6.sh
root@9a4a8a5799315e0:~# chmod +x forloop6.sh
```

Step 3: Write the Code in nano forloop6.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                forloop6.sh *
#!/bin/bash
#For Loop to Read Three-expression
for ((i=1;i<=10;i++ ))
do
echo "$i"
done
```

Step 4: Output

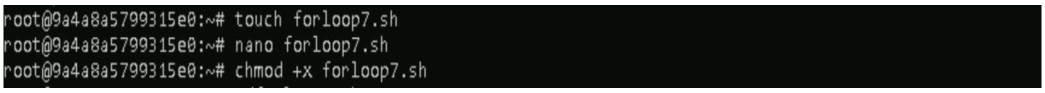


```
root@9a4a8a5799315e0:~# ./forloop6.sh
1
2
3
4
5
6
7
8
9
10
```

11.A Shell Program to define a ‘For Loop with the Break Statement’.

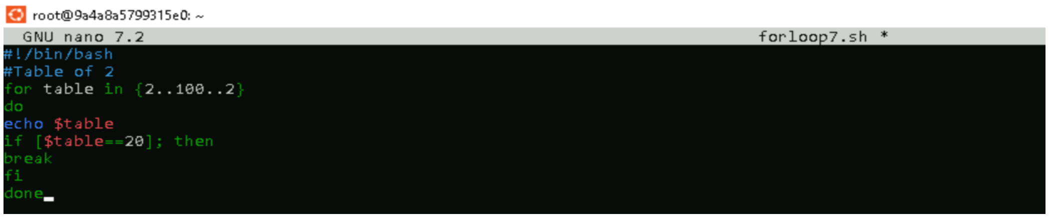
Step 1: Create an “forloop7.sh” script file using touch command

Step 2: Create a nano file to write the code



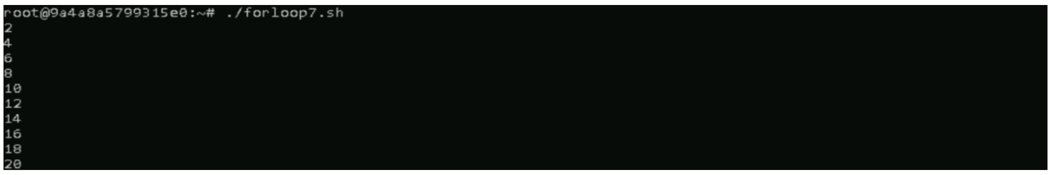
```
root@9a4a8a5799315e0:~# touch forloop7.sh
root@9a4a8a5799315e0:~# nano forloop7.sh
root@9a4a8a5799315e0:~# chmod +x forloop7.sh
```

Step 3: Write the Code in nano forloop7.sh script file



```
root@9a4a8a5799315e0: ~
 GNU nano 7.2                                              forloop7.sh *
#!/bin/bash
#Table of 2
for table in {2..100..2}
do
echo $table
if [$table==20]; then
break
fi
done
```

Step 4: Output

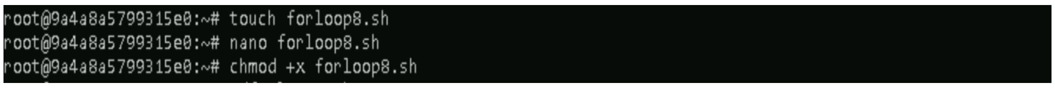


```
root@9a4a8a5799315e0:~# ./forloop7.sh
2
4
6
8
10
12
14
16
18
20
```

12.A Shell Program to define ‘For Loop with a Continue Statement’.

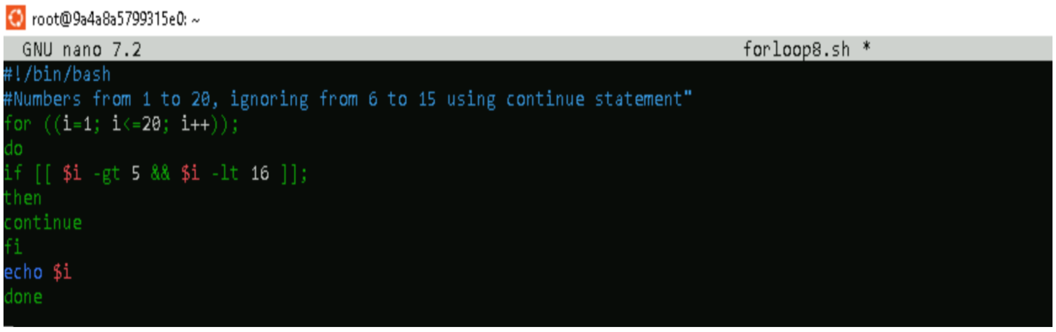
Step 1: Create an “forloop8.sh” script file using touch command

Step 2: Create a nano file to write the code



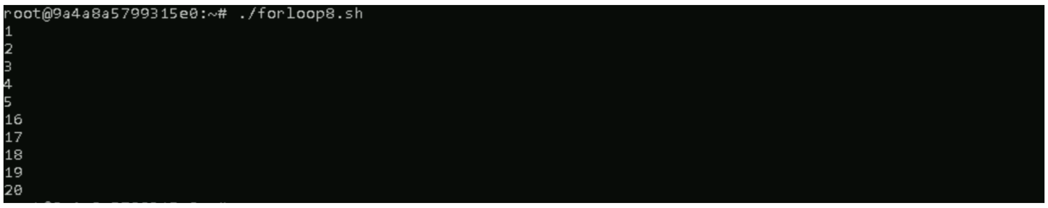
```
root@9a4a8a5799315e0:~# touch forloop8.sh
root@9a4a8a5799315e0:~# nano forloop8.sh
root@9a4a8a5799315e0:~# chmod +x forloop8.sh
```

Step 3: Write the Code in nano forloop8.sh script file



```
root@9a4a8a5799315e0: ~
 GNU nano 7.2                                              forloop8.sh *
#!/bin/bash
#Numbers from 1 to 20, ignoring from 6 to 15 using continue statement"
for ((i=1; i<=20; i++));
do
if [[ $i -gt 5 && $i -lt 16 ]];
then
continue
fi
echo $i
done
```

Step 4: Output

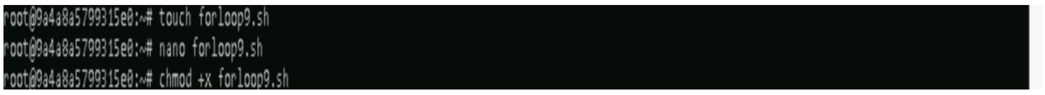


```
root@9a4a8a5799315e0:~# ./forloop8.sh
1
2
3
4
5
16
17
18
19
20
```

13.A Shell Program to define a ‘Infinite bash For Loop’.

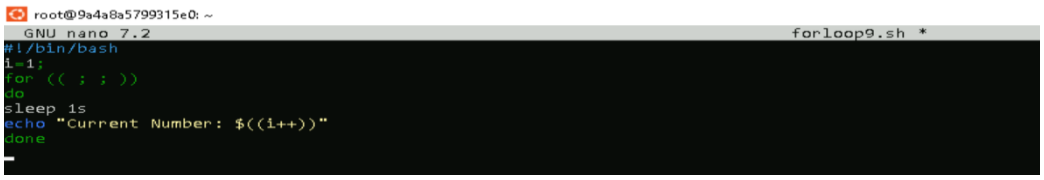
Step 1: Create an “forloop9.sh” script file using touch command

Step 2: Create a nano file to write the code



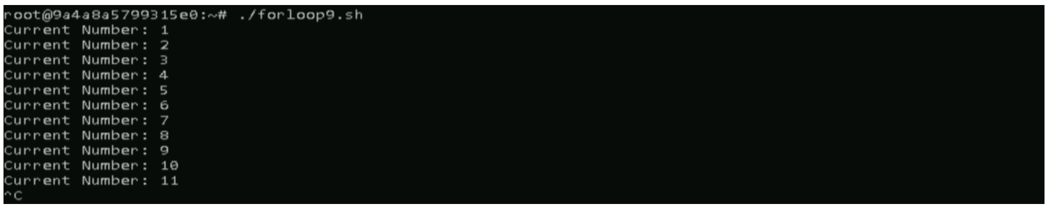
```
root@9a4a8a5799315e0:~# touch forloop9.sh
root@9a4a8a5799315e0:~# nano forloop9.sh
root@9a4a8a5799315e0:~# chmod +x forloop9.sh
```

Step 3: Write the Code in nano forloop9.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                  forloop9.sh *
#!/bin/bash
i=1;
for (( ; ; ))
do
sleep 1s
echo "Current Number: $((i++))"
done
```

Step 4: Output



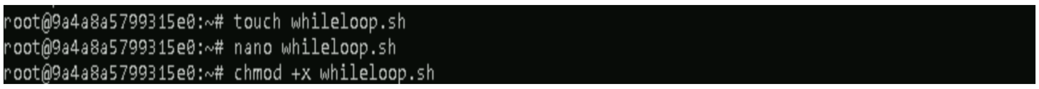
```
root@9a4a8a5799315e0:~# ./forloop9.sh
Current Number: 1
Current Number: 2
Current Number: 3
Current Number: 4
Current Number: 5
Current Number: 6
Current Number: 7
Current Number: 8
Current Number: 9
Current Number: 10
Current Number: 11
^C
```

WHILE LOOP

14.A Shell Program to define 'While Loop' to print series of numbers as per user input.

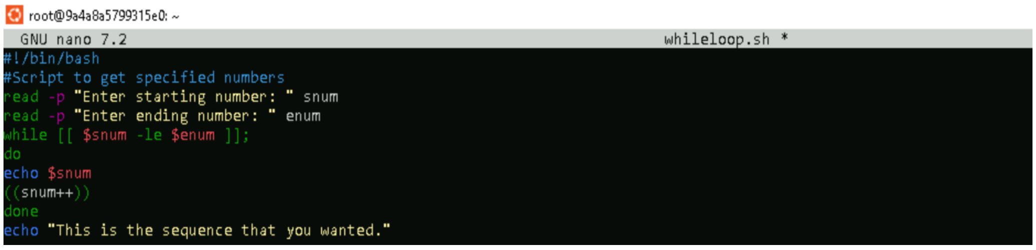
Step 1: Create an "whileloop.sh" script file using touch command

Step 2: Create a nano file to write the code



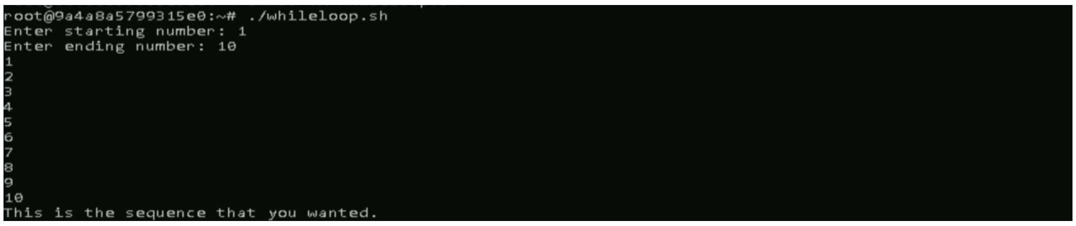
```
root@9a4a8a5799315e0:~# touch whileloop.sh
root@9a4a8a5799315e0:~# nano whileloop.sh
root@9a4a8a5799315e0:~# chmod +x whileloop.sh
```

Step 3: Write the Code in nano whileloop.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                        whileloop.sh *
#!/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."
```

Step 4: Output

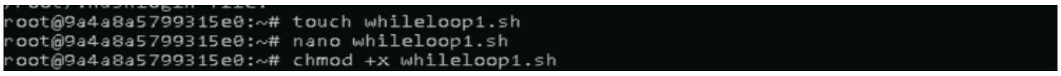


```
root@9a4a8a5799315e0:~# ./whileloop.sh
Enter starting number: 1
Enter ending number: 10
1
2
3
4
5
6
7
8
9
10
This is the sequence that you wanted.
```

15.A Shell Program to define a 'While Loop' with multiple conditions

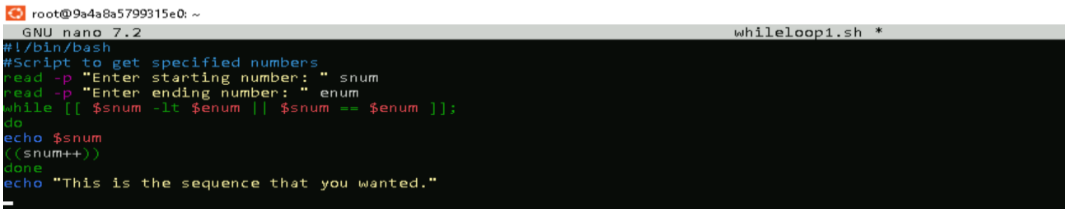
Step 1: Create an "whileloop1.sh" script file using touch command

Step 2: Create a nano file to write the code



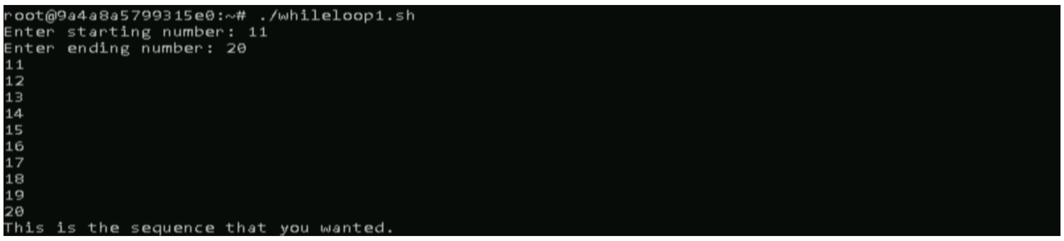
```
root@9a4a8a5799315e0:~# touch whileloop1.sh
root@9a4a8a5799315e0:~# nano whileloop1.sh
root@9a4a8a5799315e0:~# chmod +x whileloop1.sh
```

Step 3: Write the Code in nano whileloop1.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                        whileloop1.sh *
#!/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."
```

Step 4: Output

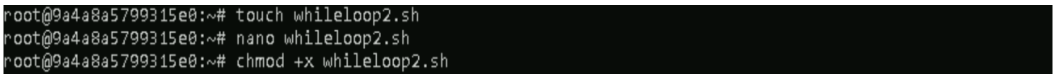


```
root@9a4a8a5799315e0:~# ./whileloop1.sh
Enter starting number: 11
Enter ending number: 20
11
12
13
14
15
16
17
18
19
20
This is the sequence that you wanted.
```

16.A Shell Program to define a ‘Infinite bash While Loop’.

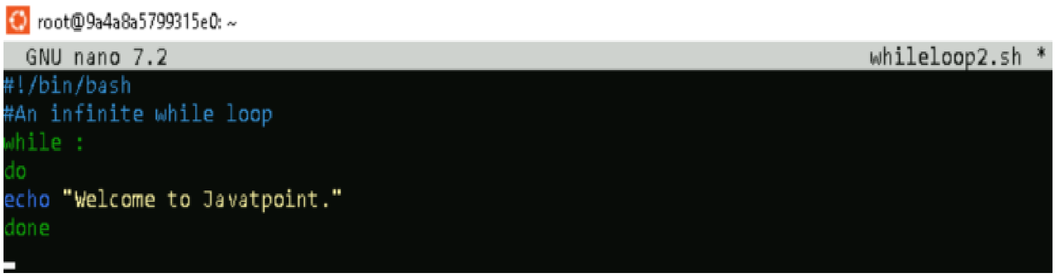
Step 1: Create an “whileloop2.sh” script file using touch command

Step 2: Create a nano file to write the code



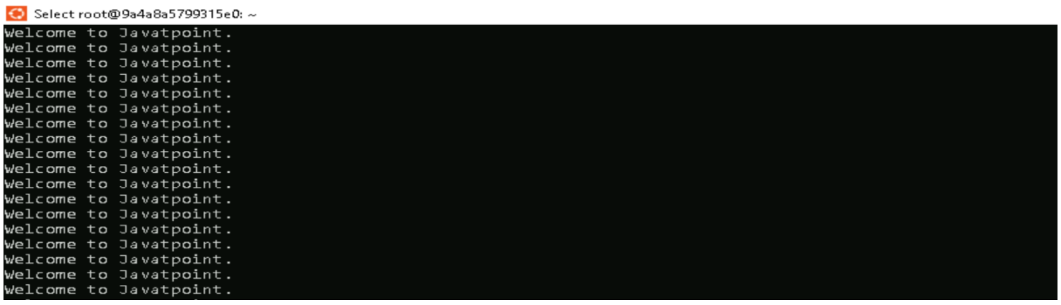
```
root@9a4a8a5799315e0:~# touch whileloop2.sh
root@9a4a8a5799315e0:~# nano whileloop2.sh
root@9a4a8a5799315e0:~# chmod +x whileloop2.sh
```

Step 3: Write the Code in nano whileloop2.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                             whileloop2.sh *
#!/bin/bash
#An infinite while loop
while :
do
echo "Welcome to Javatpoint."
done
```

Step 4: Output

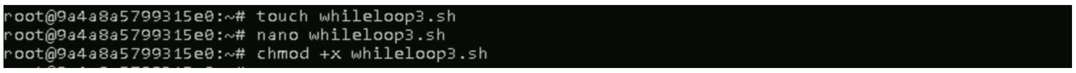


```
Select root@9a4a8a5799315e0: ~
Welcome to Javatpoint.
Welcome to Javatpoint.
Welcome to Javatpoint.
Welcome to Javatpoint.
Welcome to Javatpoint.
Welcome to Javatpoint.
Welcome to Javatpoint.
Welcome to Javatpoint.
Welcome to Javatpoint.
Welcome to Javatpoint.
Welcome to Javatpoint.
Welcome to Javatpoint.
Welcome to Javatpoint.
Welcome to Javatpoint.
Welcome to Javatpoint.
Welcome to Javatpoint.
Welcome to Javatpoint.
Welcome to Javatpoint.
```

17.A Shell Program to define a ‘Infinite bash While Loop’.

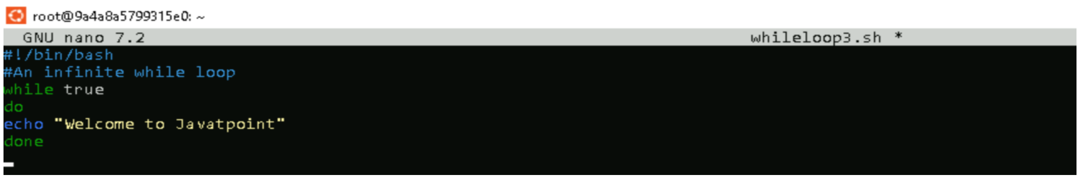
Step 1: Create an “whileloop3.sh” script file using touch command

Step 2: Create a nano file to write the code



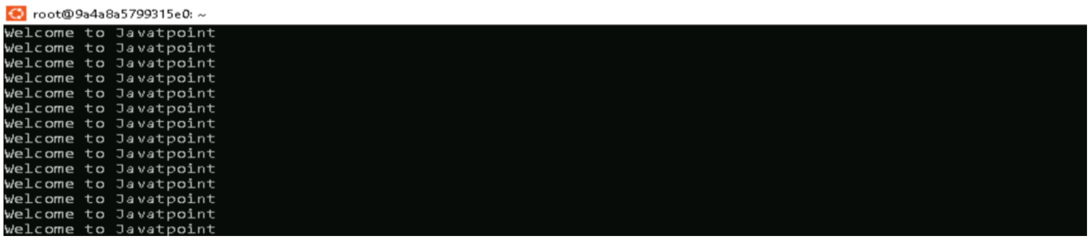
```
root@9a4a8a5799315e0:~# touch whileloop3.sh
root@9a4a8a5799315e0:~# nano whileloop3.sh
root@9a4a8a5799315e0:~# chmod +x whileloop3.sh
```

Step 3: Write the Code in nano whileloop3.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                   whileloop3.sh *
#!/bin/bash
#An infinite while loop
while true
do
echo "Welcome to Javatpoint"
done
```

Step 4: Output

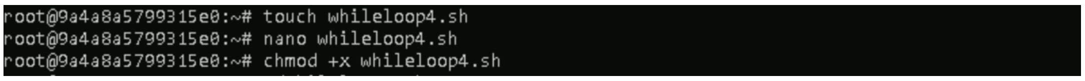


```
root@9a4a8a5799315e0: ~
Welcome to Javatpoint
Welcome to Javatpoint
Welcome to Javatpoint
Welcome to Javatpoint
Welcome to Javatpoint
Welcome to Javatpoint
Welcome to Javatpoint
Welcome to Javatpoint
Welcome to Javatpoint
Welcome to Javatpoint
Welcome to Javatpoint
Welcome to Javatpoint
Welcome to Javatpoint
Welcome to Javatpoint
```

18.A Shell Program to define a ‘While Loop with a Break Statement’.

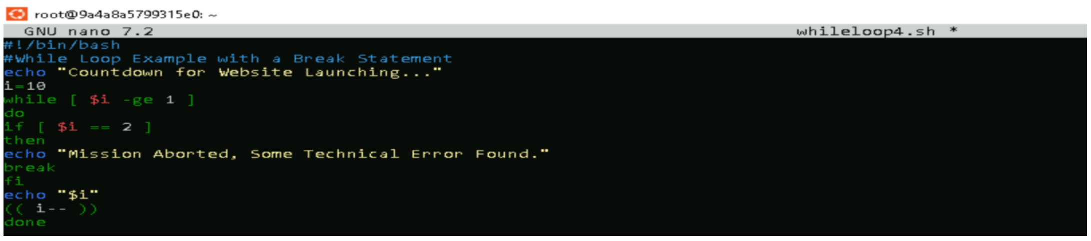
Step 1: Create an “whileloop4.sh” script file using touch command

Step 2: Create a nano file to write the code



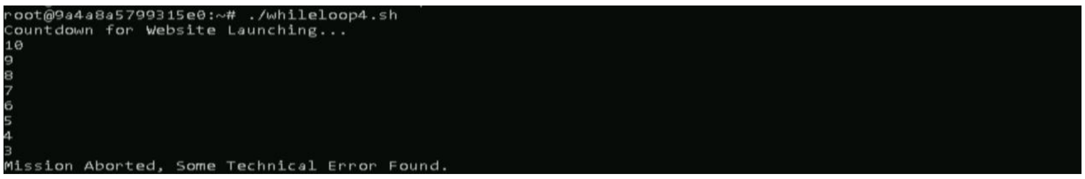
```
root@9a4a8a5799315e0:~# touch whileloop4.sh
root@9a4a8a5799315e0:~# nano whileloop4.sh
root@9a4a8a5799315e0:~# chmod +x whileloop4.sh
```

Step 3: Write the Code in nano whileloop4.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                  whileloop4.sh *
#!/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
```

Step 4: Output

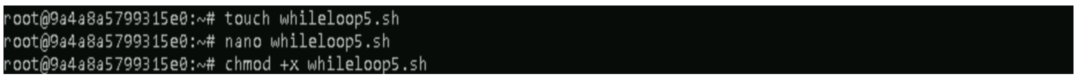


```
root@9a4a8a5799315e0:~# ./whileloop4.sh
Countdown for Website Launching...
10
9
8
7
6
5
4
3
Mission Aborted, Some Technical Error Found.
```

19.A Shell Program to define a ‘While Loop with a Continue Statement’.

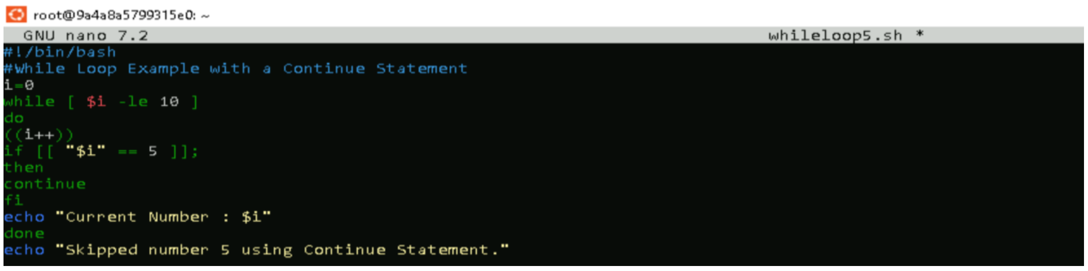
Step 1: Create an “whileloop5.sh” script file using touch command

Step 2: Create a nano file to write the code



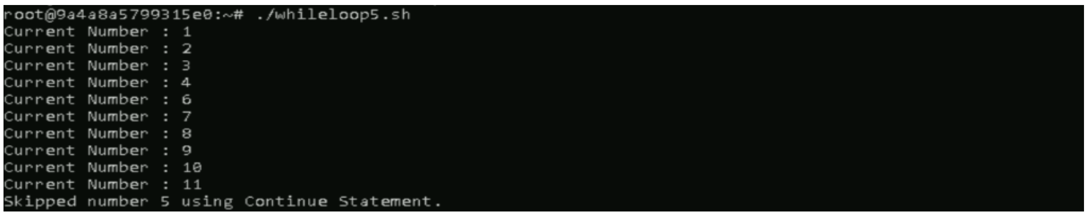
```
root@9a4a8a5799315e0:~# touch whileloop5.sh
root@9a4a8a5799315e0:~# nano whileloop5.sh
root@9a4a8a5799315e0:~# chmod +x whileloop5.sh
```

Step 3: Write the Code in nano whileloop5.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                    whileloop5.sh *
#!/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."
```

Step 4: Output

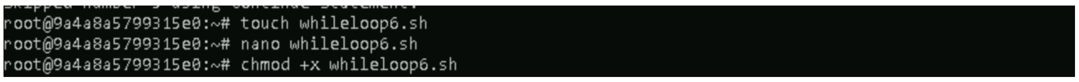


```
root@9a4a8a5799315e0:~# ./whileloop5.sh
Current Number : 1
Current Number : 2
Current Number : 3
Current Number : 4
Current Number : 6
Current Number : 7
Current Number : 8
Current Number : 9
Current Number : 10
Current Number : 11
Skipped number 5 using Continue Statement.
```

20.A Shell Program to define a while loop in bash script as similar as a 'While Loop in C programming language'.

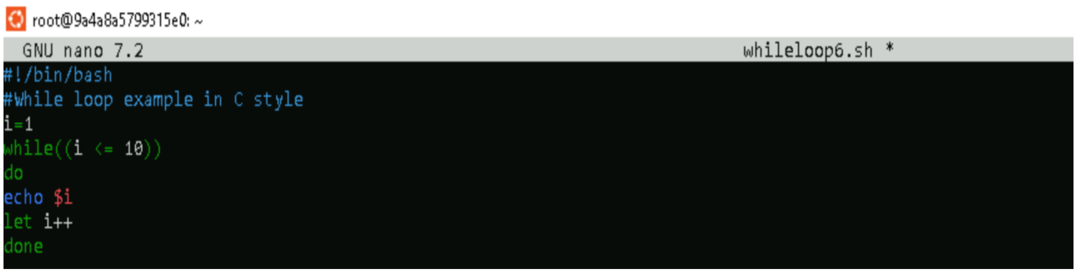
Step 1: Create an "whileloop6.sh" script file using touch command

Step 2: Create a nano file to write the code



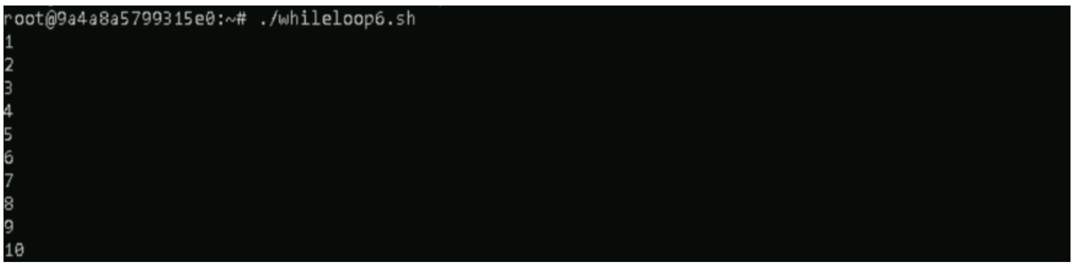
```
root@9a4a8a5799315e0:~# touch whileloop6.sh
root@9a4a8a5799315e0:~# nano whileloop6.sh
root@9a4a8a5799315e0:~# chmod +x whileloop6.sh
```

Step 3: Write the Code in nano whileloop6.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                 whileloop6.sh *
#!/bin/bash
#While loop example in C style
i=1
while((i <= 10))
do
echo $i
let i++
done
```

Step 4: Output



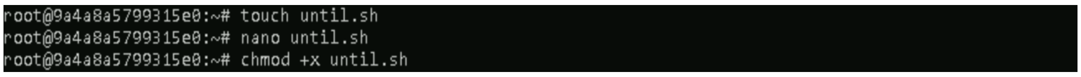
```
root@9a4a8a5799315e0:~# ./whileloop6.sh
1
2
3
4
5
6
7
8
9
10
```

UNTIL LOOP

21.A basic example of 'Until Loop' which will print series of numbers from 1 to 10.

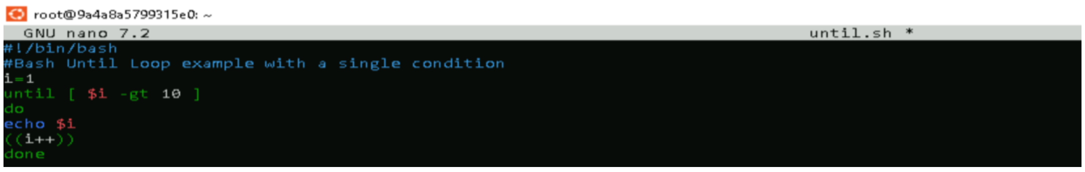
Step 1: Create an "until.sh" script file using touch command

Step 2: Create a nano file to write the code



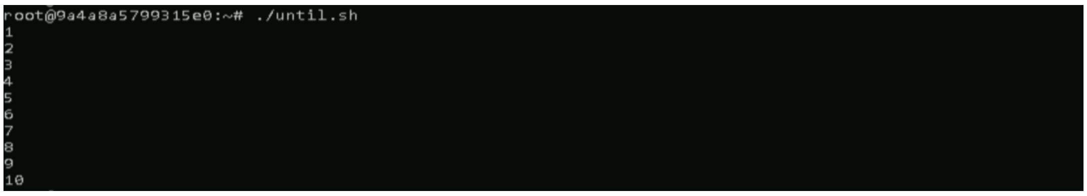
```
root@9a4a8a5799315e0:~# touch until.sh
root@9a4a8a5799315e0:~# nano until.sh
root@9a4a8a5799315e0:~# chmod +x until.sh
```

Step 3: Write the Code in nano until.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                  until.sh *
#!/bin/bash
#Bash Until Loop example with a single condition
i=1
until [ $i -gt 10 ]
do
echo $i
((i++))
done
```

Step 4: Output

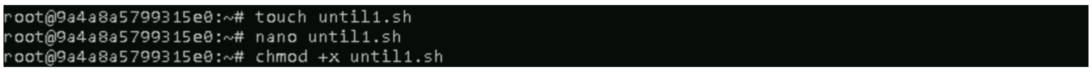


```
root@9a4a8a5799315e0:~# ./until.sh
1
2
3
4
5
6
7
8
9
10
```

22.A Shell Program to define 'Until Loop' with multiple conditions.

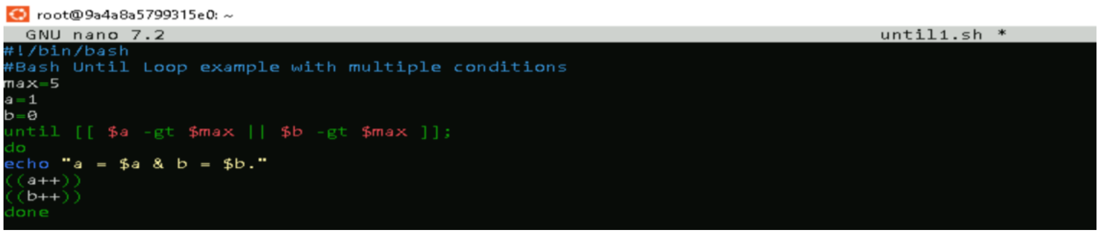
Step 1: Create an "until1.sh" script file using touch command

Step 2: Create a nano file to write the code



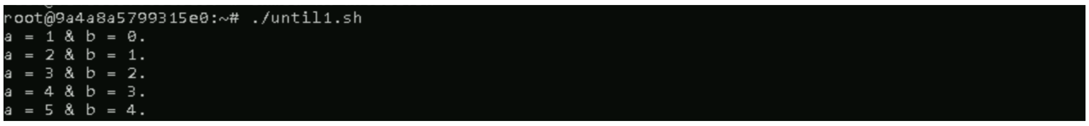
```
root@9a4a8a5799315e0:~# touch until1.sh
root@9a4a8a5799315e0:~# nano until1.sh
root@9a4a8a5799315e0:~# chmod +x until1.sh
```

Step 3: Write the Code in nano until1.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                  until1.sh *
#!/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
```

Step 4: Output



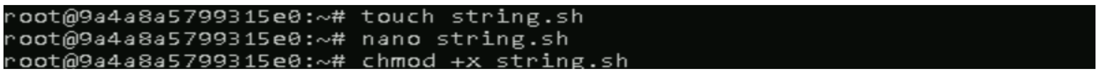
```
root@9a4a8a5799315e0:~# ./until1.sh
a = 1 & b = 0.
a = 2 & b = 1.
a = 3 & b = 2.
a = 4 & b = 3.
a = 5 & b = 4.
```

STRING

23.A Shell Program to check whether two strings are equal or not.

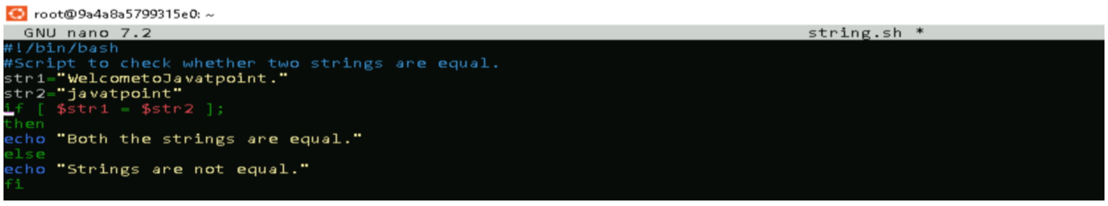
Step 1: Create an "string.sh" script file using touch command

Step 2: Create a nano file to write the code



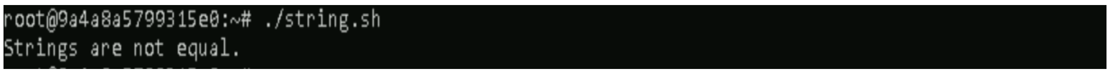
```
root@9a4a8a5799315e0:~# touch string.sh
root@9a4a8a5799315e0:~# nano string.sh
root@9a4a8a5799315e0:~# chmod +x string.sh
```

Step 3: Write the Code in nano string.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                             string.sh *
#!/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
```

Step 4: Output

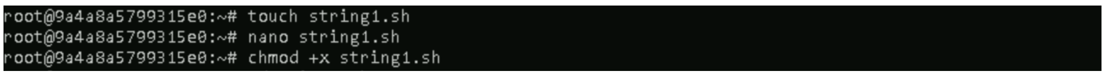


```
root@9a4a8a5799315e0:~# ./string.sh
Strings are not equal.
```

24.A Shell Program to check whether two strings are equal or not..

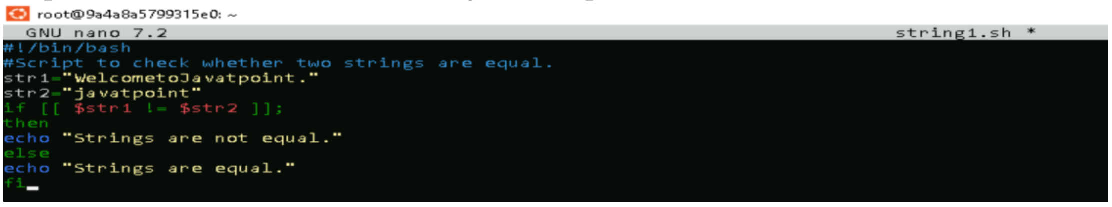
Step 1: Create an “string1.sh” script file using touch command

Step 2: Create a nano file to write the code



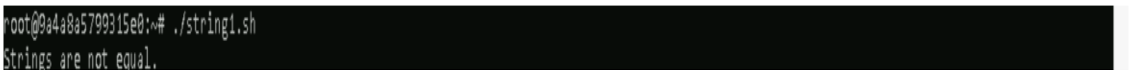
```
root@9a4a8a5799315e0:~# touch string1.sh
root@9a4a8a5799315e0:~# nano string1.sh
root@9a4a8a5799315e0:~# chmod +x string1.sh
```

Step 3: Write the Code in nano string1.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                          string1.sh *
#!/bin/bash
#Script to check whether two strings are equal.
str1="WelcometoJavatpoint."
str2="javatpoint"
if [[ $str1 != $str2 ]];
then
echo "Strings are not equal."
else
echo "Strings are equal."
fi
```

Step 4: Output

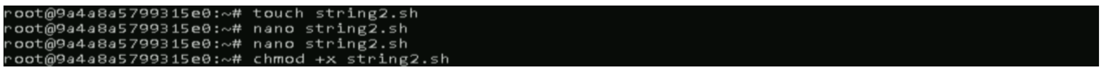


```
root@9a4a8a5799315e0:~# ./string1.sh
Strings are not equal.
```

25.A Shell Program to define a conditional operator which is used to check if string1 is less than string2.

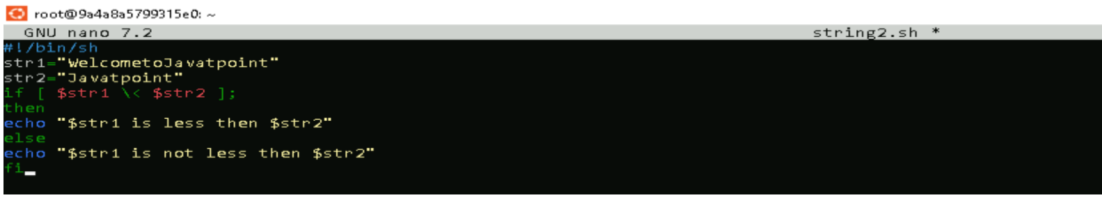
Step 1: Create an “string2.sh” script file using touch command

Step 2: Create a nano file to write the code



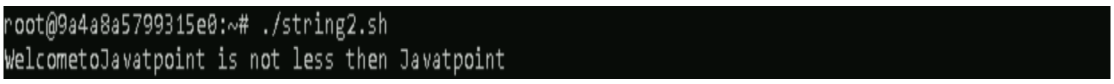
```
root@9a4a8a5799315e0:~# touch string2.sh
root@9a4a8a5799315e0:~# nano string2.sh
root@9a4a8a5799315e0:~# nano string2.sh
root@9a4a8a5799315e0:~# chmod +x string2.sh
```

Step 3: Write the Code in nano string2.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                          string2.sh *
#!/bin/sh
str1="WelcometoJavatpoint"
str2="Javatpoint"
if [ $str1 \< $str2 ];
then
echo "$str1 is less then $str2"
else
echo "$str1 is not less then $str2"
fi
```

Step 4: Output

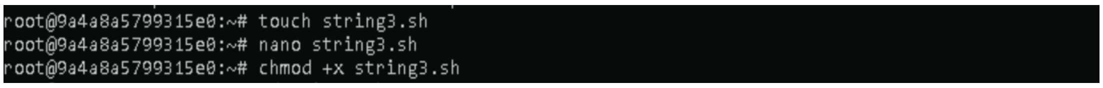


```
root@9a4a8a5799315e0:~# ./string2.sh
WelcometoJavatpoint is not less then Javatpoint
```

26.A Shell Program to define a conditional operator which is used to check if string1 is greater than string2.

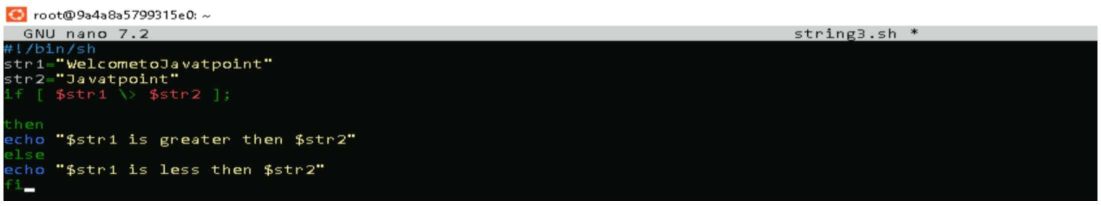
Step 1: Create an “string3.sh” script file using touch command

Step 2: Create a nano file to write the code



```
root@9a4a8a5799315e0:~# touch string3.sh
root@9a4a8a5799315e0:~# nano string3.sh
root@9a4a8a5799315e0:~# chmod +x string3.sh
```

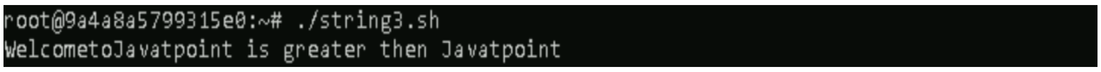
Step 3: Write the Code in nano string.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                          string3.sh *
#!/bin/sh
str1="WelcometoJavatpoint"
str2="Javatpoint"
if [ $str1 \> $str2 ];

then
echo "$str1 is greater then $str2"
else
echo "$str1 is less then $str2"
fi
```

Step 4: Output

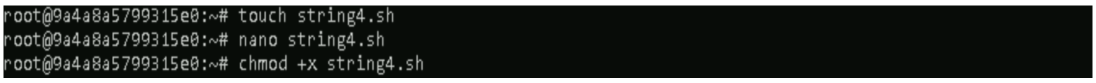


```
root@9a4a8a5799315e0:~# ./string3.sh
WelcometoJavatpoint is greater then Javatpoint
```

27.A Shell Program to check if the string length is greater than zero.

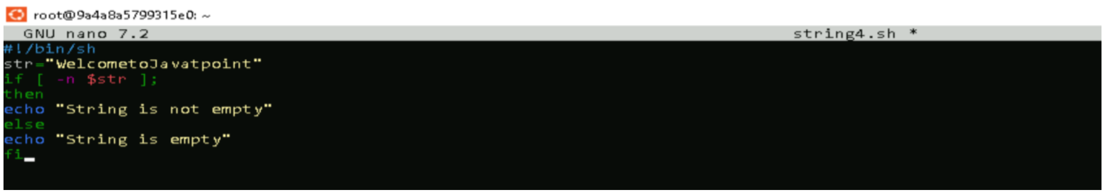
Step 1: Create an “string4.sh” script file using touch command

Step 2: Create a nano file to write the code



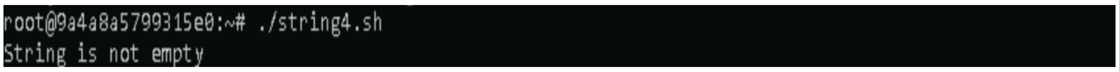
```
root@9a4a8a5799315e0:~# touch string4.sh
root@9a4a8a5799315e0:~# nano string4.sh
root@9a4a8a5799315e0:~# chmod +x string4.sh
```

Step 3: Write the Code in nano string4.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                          string4.sh *
#!/bin/sh
str="WelcometoJavatpoint"
if [ -n $str ];
then
echo "String is not empty"
else
echo "String is empty"
fi
```

Step 4: Output

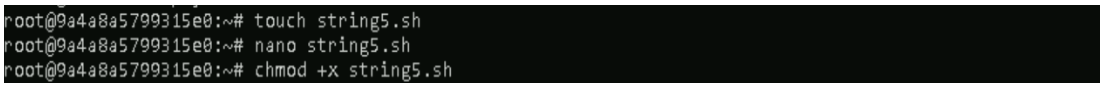


```
root@9a4a8a5799315e0:~# ./string4.sh
String is not empty
```

28.A Shell Program to check if the string length is equal to zero.

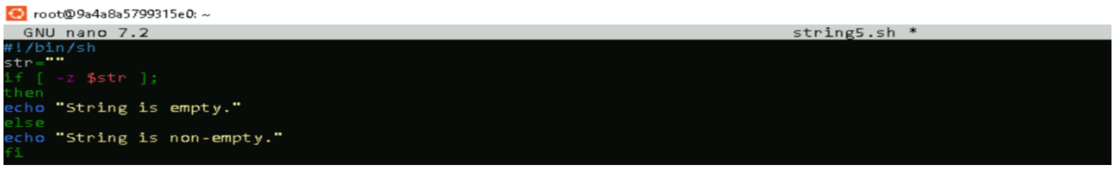
Step 1: Create an “string5.sh” script file using touch command

Step 2: Create a nano file to write the code



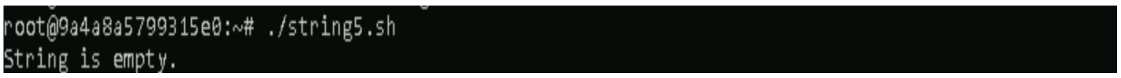
```
root@9a4a8a5799315e0:~# touch string5.sh
root@9a4a8a5799315e0:~# nano string5.sh
root@9a4a8a5799315e0:~# chmod +x string5.sh
```

Step 3: Write the Code in nano string5.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                    string5.sh *
#!/bin/sh
str=""
if [ -z $str ];
then
echo "String is empty."
else
echo "String is non-empty."
fi
```

Step 4: Output



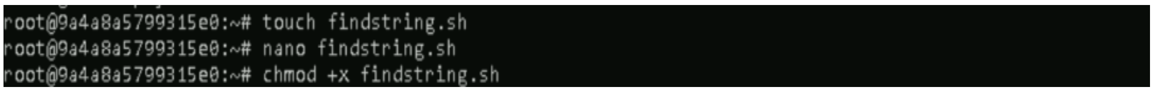
```
root@9a4a8a5799315e0:~# ./string5.sh
String is empty.
```

FIND STRING

29.A Shell Program to calculate the length of a string using '#' symbol.

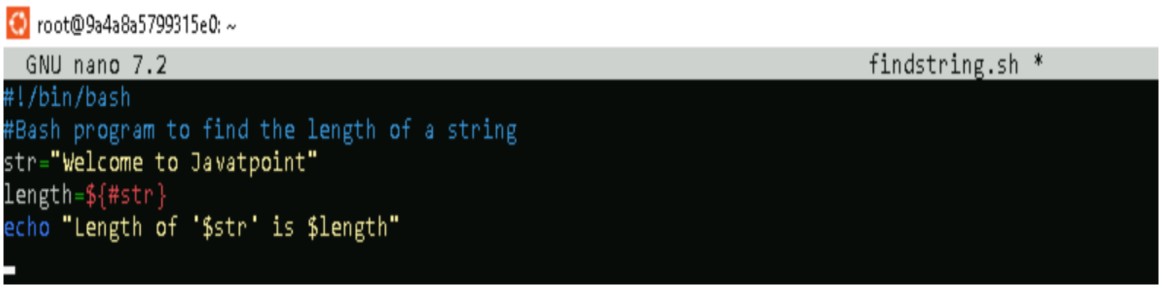
Step 1: Create an "findstring.sh" script file using touch command

Step 2: Create a nano file to write the code



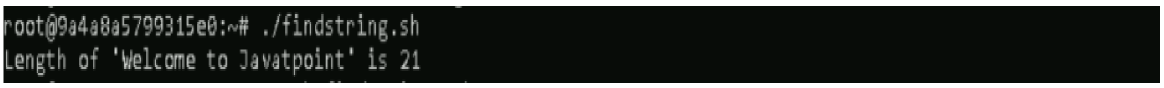
```
root@9a4a8a5799315e0:~# touch findstring.sh
root@9a4a8a5799315e0:~# nano findstring.sh
root@9a4a8a5799315e0:~# chmod +x findstring.sh
```

Step 3: Write the Code in nano findstring.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                        findstring.sh *
#!/bin/bash
#Bash program to find the length of a string
str="Welcome to Javatpoint"
length=${#str}
echo "Length of '$str' is $length"
```

Step 4: Output

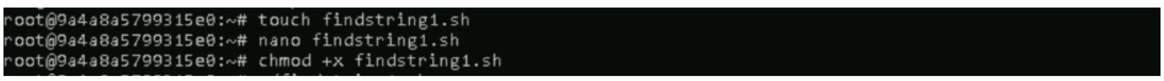


```
root@9a4a8a5799315e0:~# ./findstring.sh
Length of 'Welcome to Javatpoint' is 21
```

30.A Shell Program to calculate the length of a string using 'expr' command with the 'length' keyword.

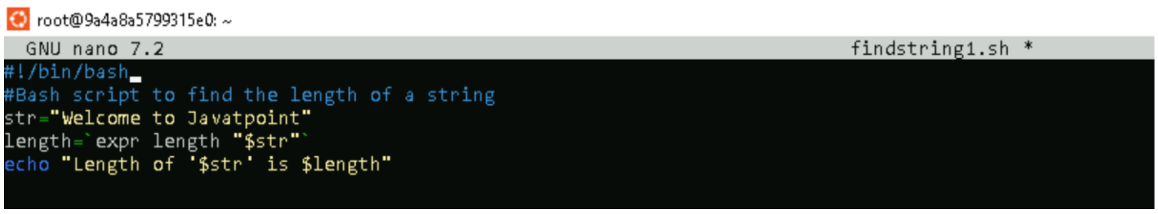
Step 1: Create an "findstring1.sh" script file using touch command

Step 2: Create a nano file to write the code



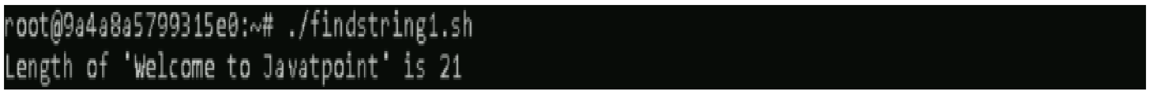
```
root@9a4a8a5799315e0:~# touch findstring1.sh
root@9a4a8a5799315e0:~# nano findstring1.sh
root@9a4a8a5799315e0:~# chmod +x findstring1.sh
```

Step 3: Write the Code in nano findstring1.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                          findstring1.sh *
#!/bin/bash
#Bash script to find the length of a string
str="Welcome to Javatpoint"
length=`expr length "$str"`
echo "Length of '$str' is $length"
```

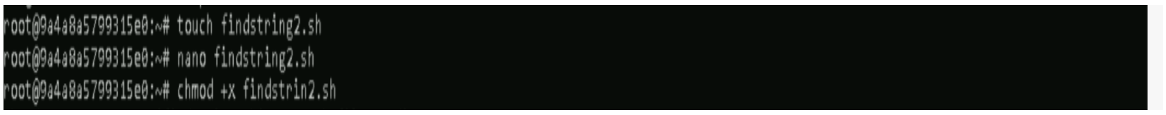
Step 4: Output


```
root@9a4a8a5799315e0:~# ./findstring1.sh
Length of 'Welcome to Javatpoint' is 21
```

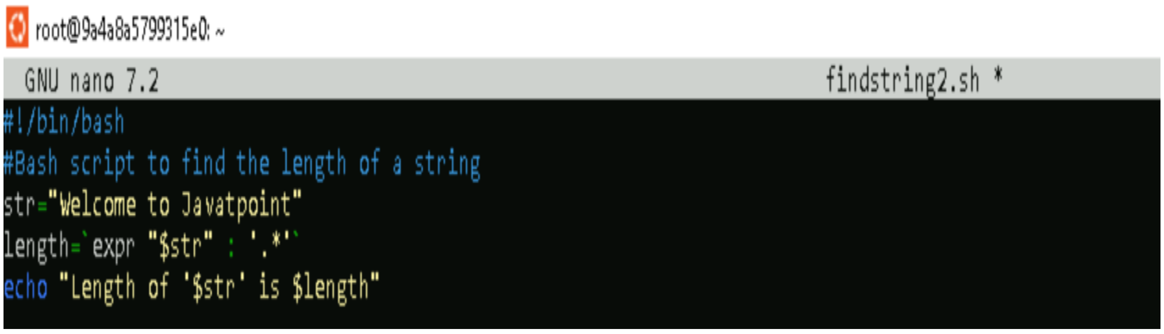
31.A Shell Program to calculate the length of a string using ‘expr “$str”:’.

Step 1: Create an “findstring2.sh” script file using touch command

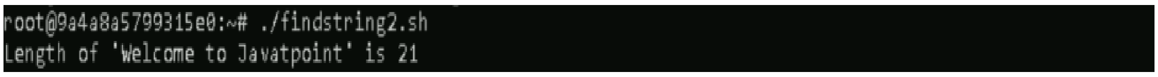
Step 2: Create a nano file to write the code


```
root@9a4a8a5799315e0:~# touch findstring2.sh
root@9a4a8a5799315e0:~# nano findstring2.sh
root@9a4a8a5799315e0:~# chmod +x findstrin2.sh
```

Step 3: Write the Code in nano findstring2.sh script file


```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                    findstring2.sh *
#!/bin/bash
#Bash script to find the length of a string
str="Welcome to Javatpoint"
length=`expr "$str" : '.*'`
echo "Length of '$str' is $length"
```

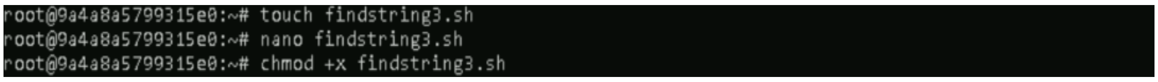
Step 4: Output


```
root@9a4a8a5799315e0:~# ./findstring2.sh
Length of 'Welcome to Javatpoint' is 21
```

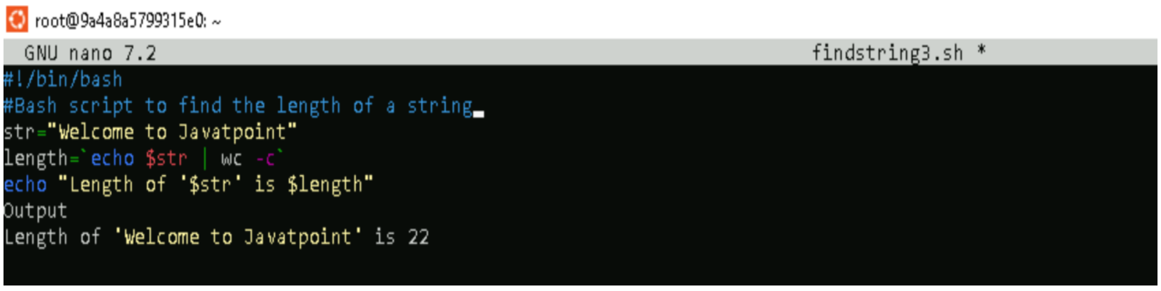
32.A Shell Program to calculate the length of a string using ‘wc’ command.

Step 1: Create an “findstring3.sh” script file using touch command

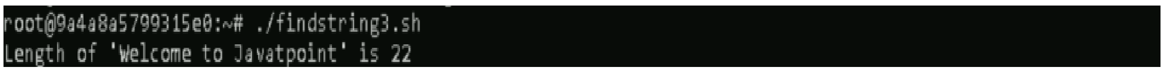
Step 2: Create a nano file to write the code


```
root@9a4a8a5799315e0:~# touch findstring3.sh
root@9a4a8a5799315e0:~# nano findstring3.sh
root@9a4a8a5799315e0:~# chmod +x findstring3.sh
```

Step 3: Write the Code in nano findstring3.sh script file


```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                    findstring3.sh *
#!/bin/bash
#Bash script to find the length of a string
str="Welcome to Javatpoint"
length=`echo $str | wc -c`
echo "Length of '$str' is $length"
Output
Length of 'Welcome to Javatpoint' is 22
```

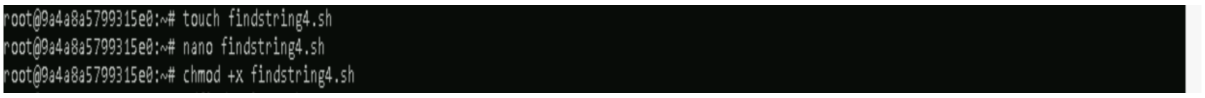
Step 4: Output


```
root@9a4a8a5799315e0:~# ./findstring3.sh
Length of 'Welcome to Javatpoint' is 22
```

33.A Shell Program to calculate the length of a string using ‘awk’ command.

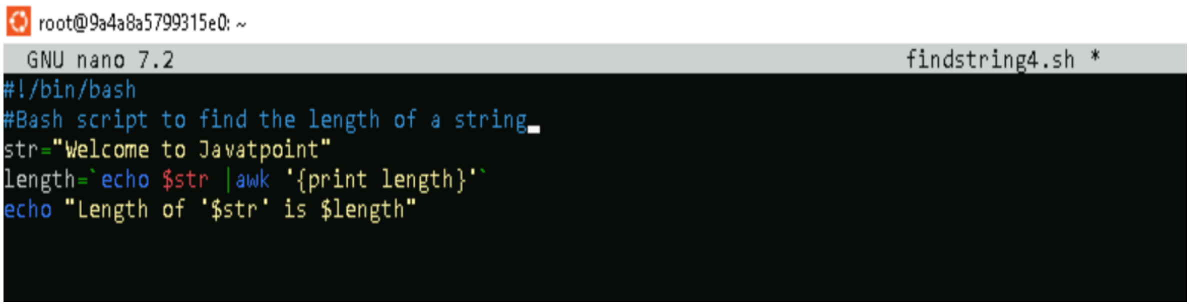
Step 1: Create an “findstring4.sh” script file using touch command

Step 2: Create a nano file to write the code



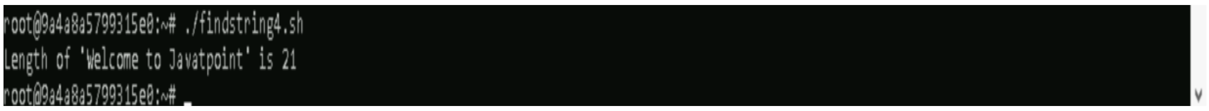
```
root@9a4a8a5799315e0:~# touch findstring4.sh
root@9a4a8a5799315e0:~# nano findstring4.sh
root@9a4a8a5799315e0:~# chmod +x findstring4.sh
```

Step 3: Write the Code in nano findstring4.sh script file



```
root@9a4a8a5799315e0: ~
 GNU nano 7.2                                          findstring4.sh *
#!/bin/bash
#Bash script to find the length of a string
str="Welcome to Javatpoint"
length=`echo $str |awk '{print length}'`
echo "Length of '$str' is $length"
```

Step 4: Output



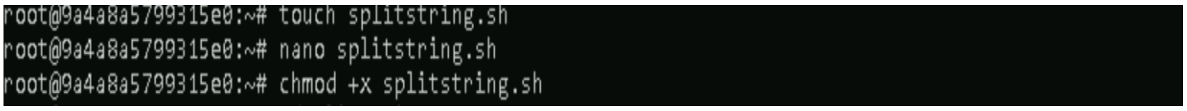
```
root@9a4a8a5799315e0:~# ./findstring4.sh
Length of 'Welcome to Javatpoint' is 21
root@9a4a8a5799315e0:~#
```

SPLIT STRING

34.A Shell Program to split a string using a space character delimiter.

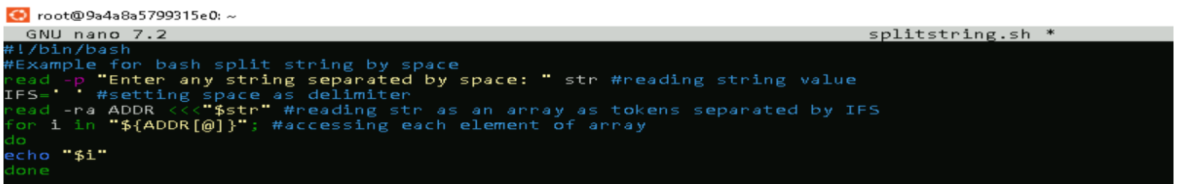
Step 1: Create an “splitstring.sh” script file using touch command

Step 2: Create a nano file to write the code



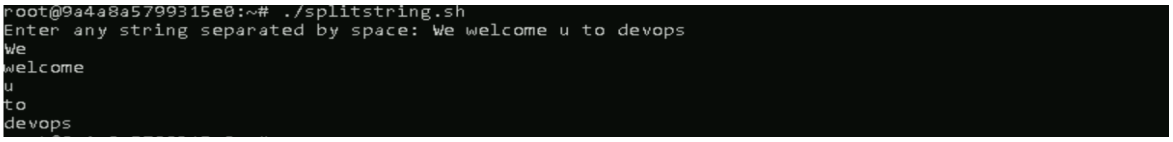
```
root@9a4a8a5799315e0:~# touch splitstring.sh
root@9a4a8a5799315e0:~# nano splitstring.sh
root@9a4a8a5799315e0:~# chmod +x splitstring.sh
```

Step 3: Write the Code in nano splitstring.sh script file



```
root@9a4a8a5799315e0: ~
 GNU nano 7.2                                          splitstring.sh *
#!/bin/bash
#Example for bash split string by space
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
```

Step 4: Output

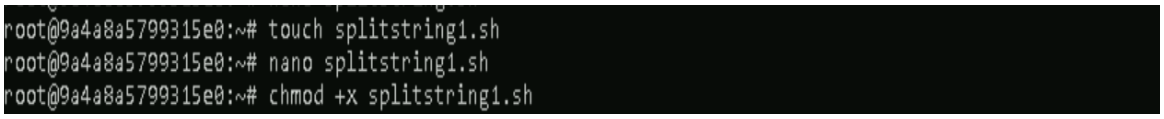


```
root@9a4a8a5799315e0:~# ./splitstring.sh
Enter any string separated by space: We welcome u to devops
We
welcome
u
to
devops
```

35.A Shell Program to split a string using comma(,) symbol character as a delimeter.

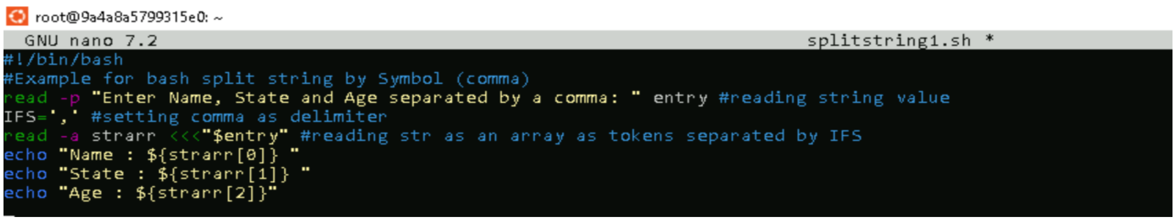
Step 1: Create an “splitstring1.sh” script file using touch command

Step 2: Create a nano file to write the code



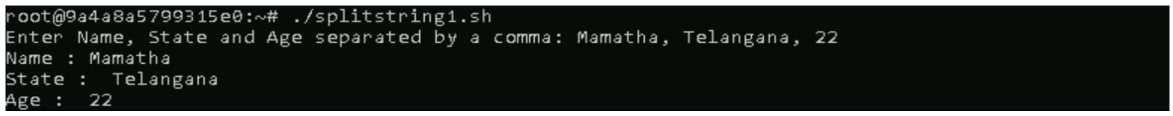
```
root@9a4a8a5799315e0:~# touch splitstring1.sh
root@9a4a8a5799315e0:~# nano splitstring1.sh
root@9a4a8a5799315e0:~# chmod +x splitstring1.sh
```

Step 3: Write the Code in nano splitstring1.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                              splitstring1.sh *
#!/bin/bash
#Example for bash split string by Symbol (comma)
read -p "Enter Name, State and Age separated by a comma: " entry #reading string value
IFS=',' #setting comma as delimiter
read -a strarr <<<"$entry" #reading str as an array as tokens separated by IFS
echo "Name : ${strarr[0]} "
echo "State : ${strarr[1]} "
echo "Age : ${strarr[2]}"
```

Step 4: Output

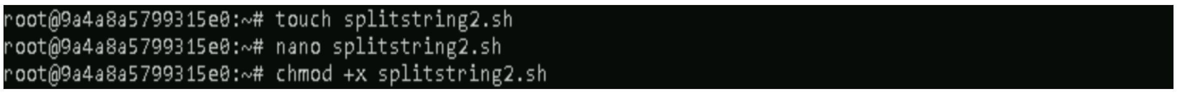


```
root@9a4a8a5799315e0:~# ./splitstring1.sh
Enter Name, State and Age separated by a comma: Mamatha, Telangana, 22
Name : Mamatha
State :  Telangana
Age :  22
```

36. A Shell Program to split a string using colon(:) symbol character as a delimeter .

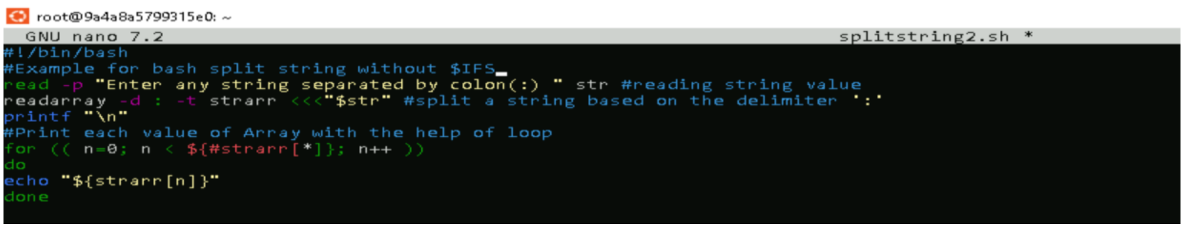
Step 1: Create an “splitstring2.sh” script file using touch command

Step 2: Create a nano file to write the code



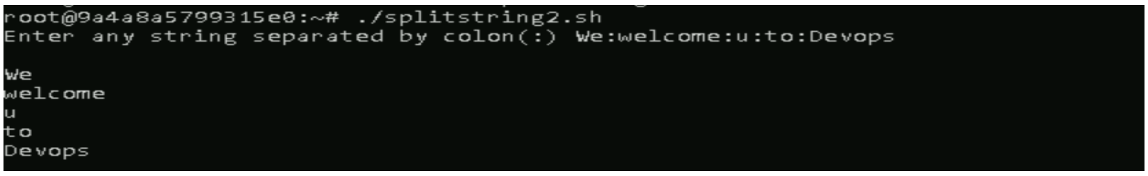
```
root@9a4a8a5799315e0:~# touch splitstring2.sh
root@9a4a8a5799315e0:~# nano splitstring2.sh
root@9a4a8a5799315e0:~# chmod +x splitstring2.sh
```

Step 3: Write the Code in nano splitstring2.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                              splitstring2.sh *
#!/bin/bash
#Example for bash split string without $IFS
read -p "Enter any string separated by colon(:) " str #reading string value
readarray -d : -t strarr <<<"$str" #split a string based on the delimiter ':'
printf "\n"
#Print each value of Array with the help of loop
for (( n=0; n < ${#strarr[*]}; n++ ))
do
echo "${strarr[n]}"
done
```

Step 4: Output



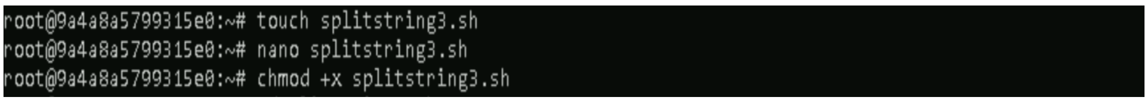
```
root@9a4a8a5799315e0:~# ./splitstring2.sh
Enter any string separated by colon(:) We:welcome:u:to:Devops

We
welcome
u
to
Devops
```

37.A Shell Program to split a string by using another string.

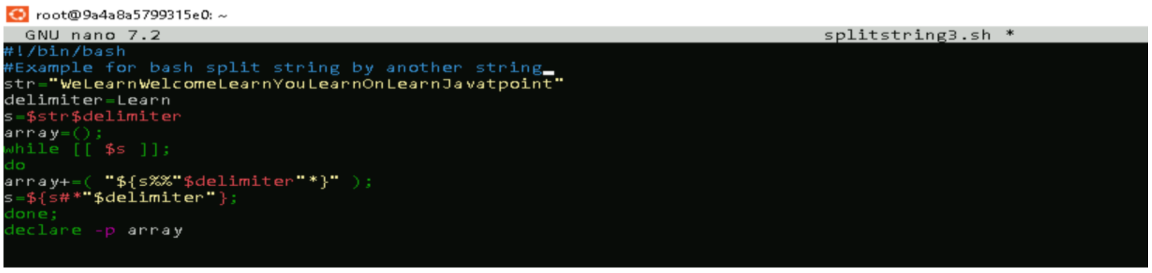
Step 1: Create an “splitstring3.sh” script file using touch command

Step 2: Create a nano file to write the code

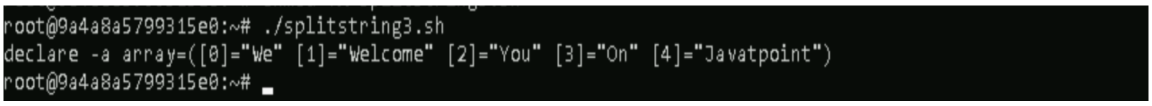


```
root@9a4a8a5799315e0:~# touch splitstring3.sh
root@9a4a8a5799315e0:~# nano splitstring3.sh
root@9a4a8a5799315e0:~# chmod +x splitstring3.sh
```

Step 3: Write the Code in nano splitstring3.sh script file


```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                    splitstring3.sh *
#!/bin/bash
#Example for bash split string by another string
str="WeLearnWelcomeLearnYouLearnOnLearnJavatpoint"
delimiter=Learn
s=$str$delimiter
array=();
while [[ $s ]];
do
array+=( "${s%%"$delimiter"*}" );
s=${s#*"$delimiter"};
done;
declare -p array
```

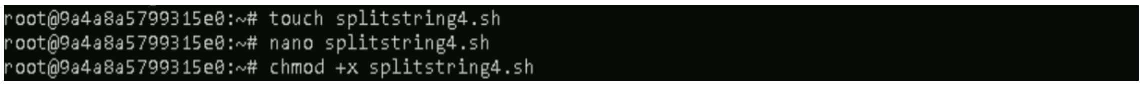
Step 4: Output


```
root@9a4a8a5799315e0:~# ./splitstring3.sh
declare -a array=([0]="We" [1]="Welcome" [2]="You" [3]="On" [4]="Javatpoint")
root@9a4a8a5799315e0:~# _
```

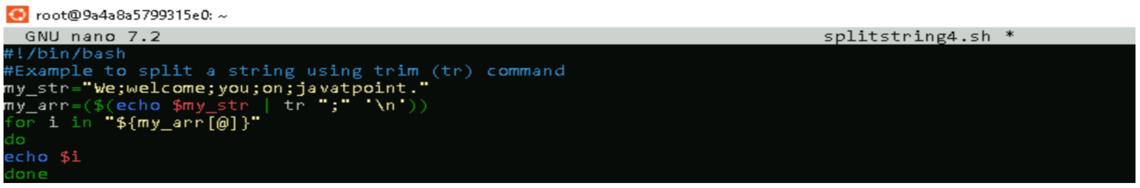
38.A Shell Program to split a string using 'Trim' command.

Step 1: Create an "splitstring4.sh" script file using touch command

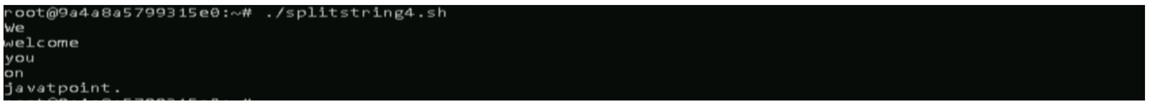
Step 2: Create a nano file to write the code


```
root@9a4a8a5799315e0:~# touch splitstring4.sh
root@9a4a8a5799315e0:~# nano splitstring4.sh
root@9a4a8a5799315e0:~# chmod +x splitstring4.sh
```

Step 3: Write the Code in nano splitstring4.sh script file


```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                    splitstring4.sh *
#!/bin/bash
#Example to split a string using trim (tr) command
my_str="We;welcome;you;on;javatpoint."
my_arr=($(echo $my_str | tr ";" '\n'))
for i in "${my_arr[@]}"
do
echo $i
done
```

Step 4: Output

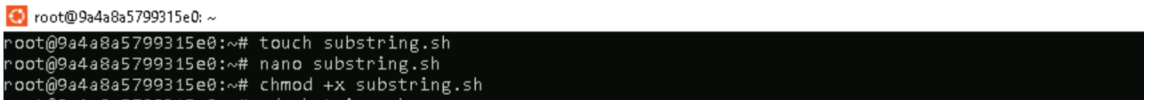

```
root@9a4a8a5799315e0:~# ./splitstring4.sh
We
welcome
you
on
javatpoint.
```

SUBSTRING

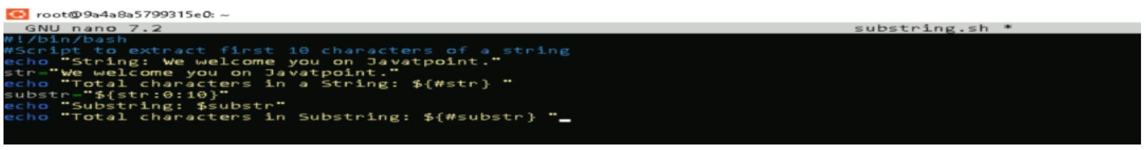
39.A Shell Program to extract a substring from the starting string.

Step 1: Create an "substring.sh" script file using touch command

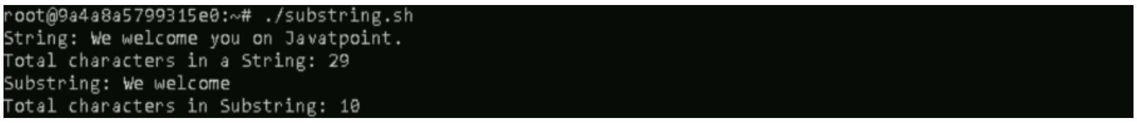
Step 2: Create a nano file to write the code


```
root@9a4a8a5799315e0: ~
root@9a4a8a5799315e0:~# touch substring.sh
root@9a4a8a5799315e0:~# nano substring.sh
root@9a4a8a5799315e0:~# chmod +x substring.sh
```

Step 3: Write the Code in nano substring.sh script file


```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                    substring.sh *
#!/bin/bash
#Script to extract first 10 characters of a string
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} "_
```

Step 4: Output

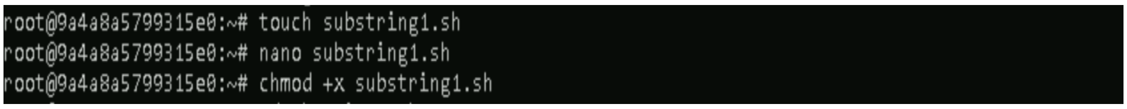


```
root@9a4a8a5799315e0:~# ./substring.sh
String: We welcome you on Javatpoint.
Total characters in a String: 29
Substring: We welcome
Total characters in Substring: 10
```

40.A Shell Program to extract a substring from specific character onwards .

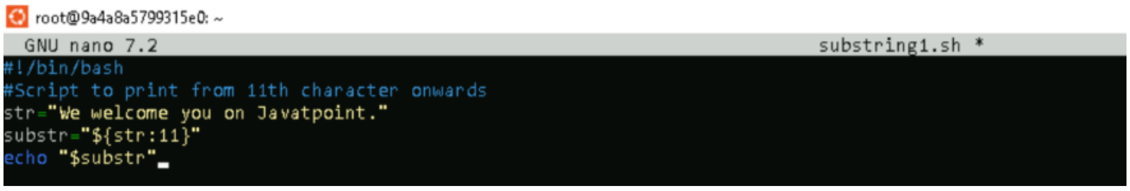
Step 1: Create an “substring1.sh” script file using touch command

Step 2: Create a nano file to write the code



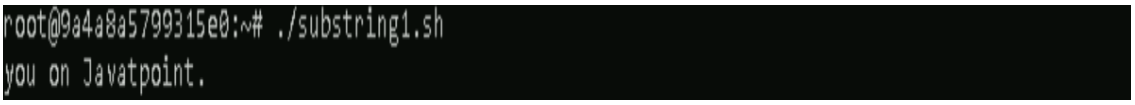
```
root@9a4a8a5799315e0:~# touch substring1.sh
root@9a4a8a5799315e0:~# nano substring1.sh
root@9a4a8a5799315e0:~# chmod +x substring1.sh
```

Step 3: Write the Code in nano substring1.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                          substring1.sh *
#!/bin/bash
#Script to print from 11th character onwards
str="We welcome you on Javatpoint."
substr="${str:11}"
echo "$substr"
```

Step 4: Output

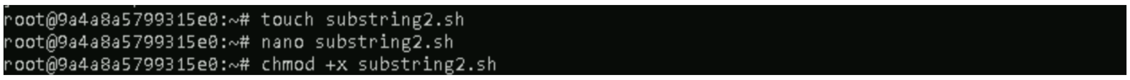


```
root@9a4a8a5799315e0:~# ./substring1.sh
you on Javatpoint.
```

41.A Shell Program to extract a single character from the string.

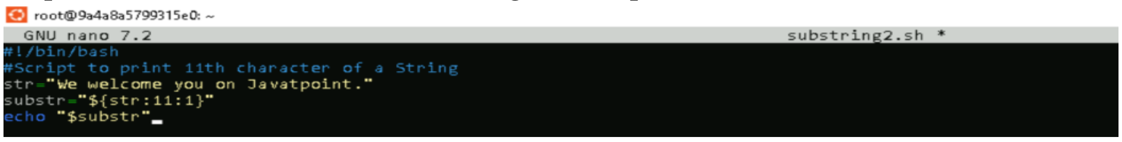
Step 1: Create an “substring2.sh” script file using touch command

Step 2: Create a nano file to write the code



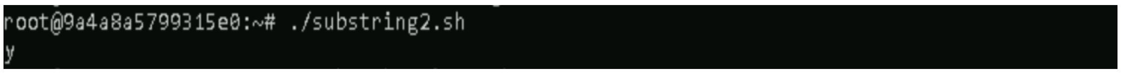
```
root@9a4a8a5799315e0:~# touch substring2.sh
root@9a4a8a5799315e0:~# nano substring2.sh
root@9a4a8a5799315e0:~# chmod +x substring2.sh
```

Step 3: Write the Code in nano substring2.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                        substring2.sh *
#!/bin/bash
#Script to print 11th character of a String
str="We welcome you on Javatpoint."
substr="${str:11:1}"
echo "$substr"
```

Step 4: Output

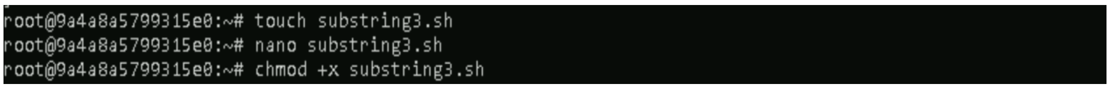


```
root@9a4a8a5799315e0:~# ./substring2.sh
y
```

42.A Shell Program to extract the specific characters from the last of the string.

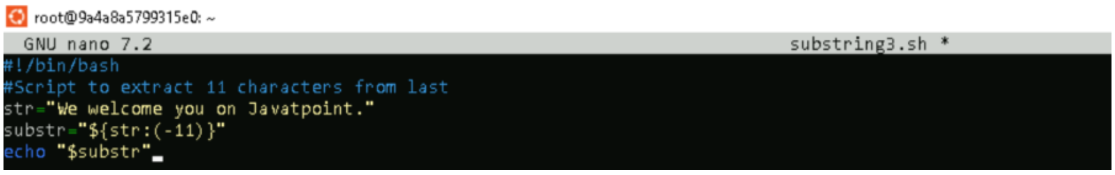
Step 1: Create an “substring3.sh” script file using touch command

Step 2: Create a nano file to write the code



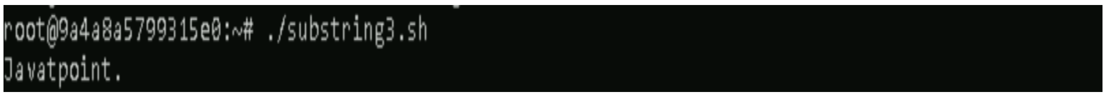
```
root@9a4a8a5799315e0:~# touch substring3.sh
root@9a4a8a5799315e0:~# nano substring3.sh
root@9a4a8a5799315e0:~# chmod +x substring3.sh
```

Step 3: Write the Code in nano substring3.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                  substring3.sh *
#!/bin/bash
#Script to extract 11 characters from last
str="We welcome you on Javatpoint."
substr="${str:(-11)}"
echo "$substr"
```

Step 4: Output



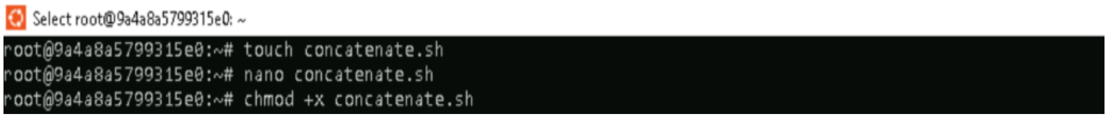
```
root@9a4a8a5799315e0:~# ./substring3.sh
Javatpoint.
```

STRING CONCATENATION

43.A Shell Program to demonstrate 'String Concatenation' with variables side by side.

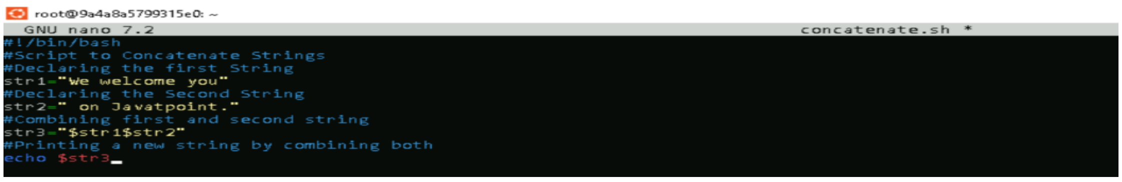
Step 1: Create an "concatenate.sh" script file using touch command

Step 2: Create a nano file to write the code



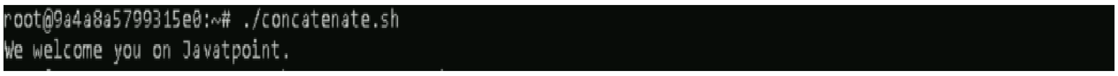
```
Select root@9a4a8a5799315e0: ~
root@9a4a8a5799315e0:~# touch concatenate.sh
root@9a4a8a5799315e0:~# nano concatenate.sh
root@9a4a8a5799315e0:~# chmod +x concatenate.sh
```

Step 3: Write the Code in nano concatenate.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                  concatenate.sh *
#!/bin/bash
#Script to Concatenate Strings
#Declaring the first String
str1="We welcome you"
#Declaring the Second String
str2=" on Javatpoint."
#Combining first and second string
str3="$str1$str2"
#Printing a new string by combining both
echo $str3
```

Step 4: Output

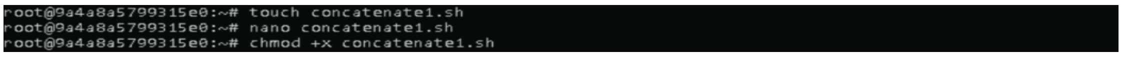


```
root@9a4a8a5799315e0:~# ./concatenate.sh
We welcome you on Javatpoint.
```

44. A Shell Program to demonstrate 'String Concatenation' using double quotes.

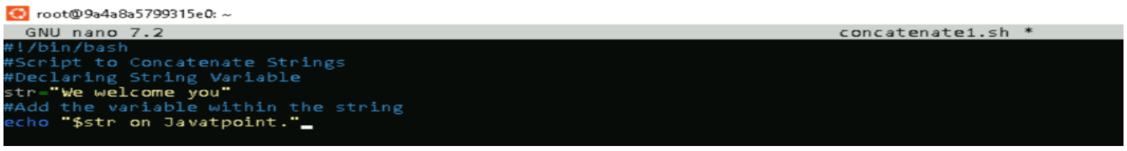
Step 1: Create an "concatenate1.sh" script file using touch command

Step 2: Create a nano file to write the code



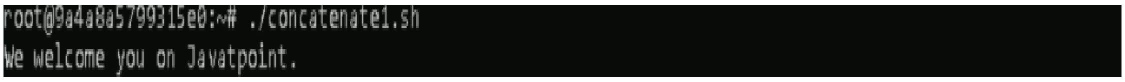
```
root@9a4a8a5799315e0:~# touch concatenate1.sh
root@9a4a8a5799315e0:~# nano concatenate1.sh
root@9a4a8a5799315e0:~# chmod +x concatenate1.sh
```

Step 3: Write the Code in nano concatenate1.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                  concatenate1.sh *
#!/bin/bash
#Script to Concatenate Strings
#Declaring String Variable
str="We welcome you"
#Add the variable within the string
echo "$str on Javatpoint."
```

Step 4: Output

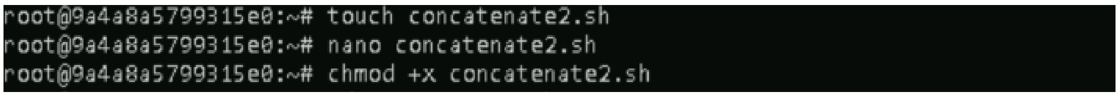


```
root@9a4a8a5799315e0:~# ./concatenate1.sh
We welcome you on Javatpoint.
```

45. A Shell Program to demonstrate ‘String Concatenation’ using append operator with loop.

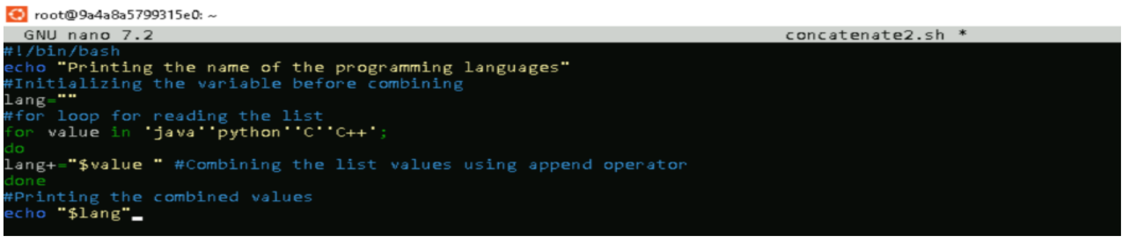
Step 1: Create an “concatenate2.sh” script file using touch command

Step 2: Create a nano file to write the code



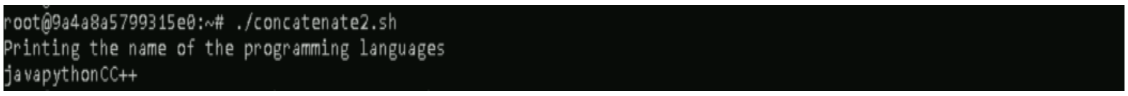
```
root@9a4a8a5799315e0:~# touch concatenate2.sh
root@9a4a8a5799315e0:~# nano concatenate2.sh
root@9a4a8a5799315e0:~# chmod +x concatenate2.sh
```

Step 3: Write the Code in nano concatenate2.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                        concatenate2.sh *
#!/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+="$value " #Combining the list values using append operator
done
#Printing the combined values
echo "$lang"
```

Step 4: Output

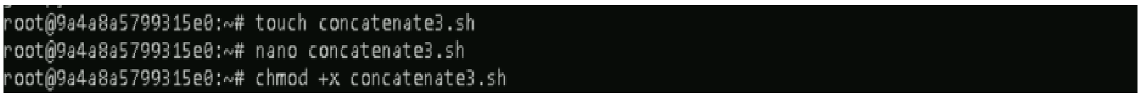


```
root@9a4a8a5799315e0:~# ./concatenate2.sh
Printing the name of the programming languages
javapythonCC++
```

46. A Shell Program to demonstrate ‘String Concatenation’ using Printf Function.

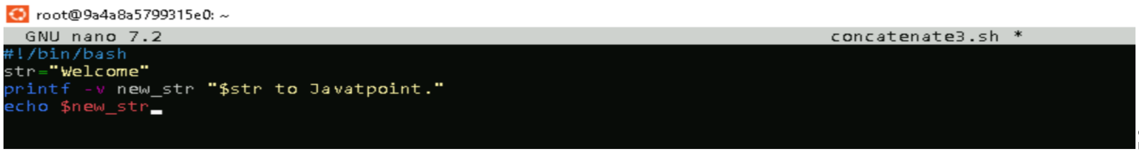
Step 1: Create an “concatenate3.sh” script file using touch command

Step 2: Create a nano file to write the code



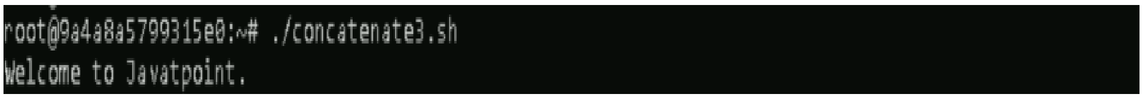
```
root@9a4a8a5799315e0:~# touch concatenate3.sh
root@9a4a8a5799315e0:~# nano concatenate3.sh
root@9a4a8a5799315e0:~# chmod +x concatenate3.sh
```

Step 3: Write the Code in nano concatenate3.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                        concatenate3.sh *
#!/bin/bash
str="Welcome"
printf -v new_str "$str to Javatpoint."
echo $new_str
```

Step 4: Output

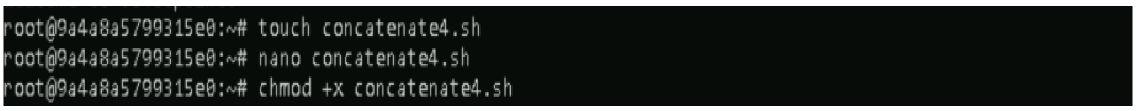


```
root@9a4a8a5799315e0:~# ./concatenate3.sh
Welcome to Javatpoint.
```

47. A Shell Program to demonstrate ‘String Concatenation’ using literal strings.

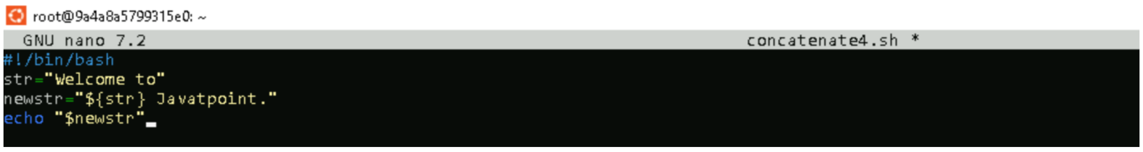
Step 1: Create an “concatenate4.sh” script file using touch command

Step 2: Create a nano file to write the code



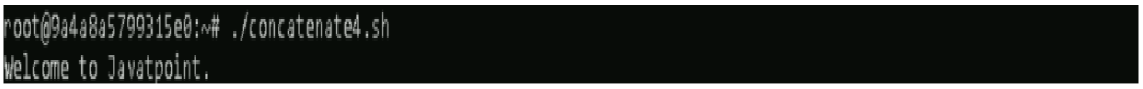
```
root@9a4a8a5799315e0:~# touch concatenate4.sh
root@9a4a8a5799315e0:~# nano concatenate4.sh
root@9a4a8a5799315e0:~# chmod +x concatenate4.sh
```

Step 3: Write the Code in nano concatenate4.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                   concatenate4.sh *
#!/bin/bash
str="Welcome to"
newstr="${str} Javatpoint."
echo "$newstr"
```

Step 4: Output

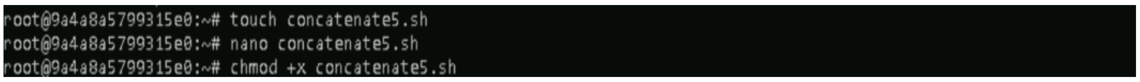


```
root@9a4a8a5799315e0:~# ./concatenate4.sh
Welcome to Javatpoint.
```

48. A Shell Program to demonstrate ‘String Concatenation’ using underscore.

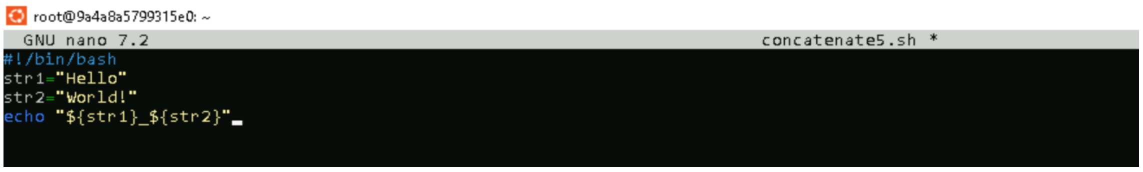
Step 1: Create an “concatenate5.sh” script file using touch command

Step 2: Create a nano file to write the code



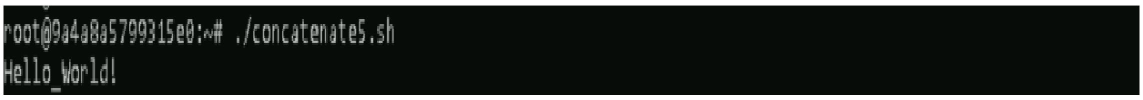
```
root@9a4a8a5799315e0:~# touch concatenate5.sh
root@9a4a8a5799315e0:~# nano concatenate5.sh
root@9a4a8a5799315e0:~# chmod +x concatenate5.sh
```

Step 3: Write the Code in nano concatenate5.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                   concatenate5.sh *
#!/bin/bash
str1="Hello"
str2="World!"
echo "${str1}_${str2}"
```

Step 4: Output

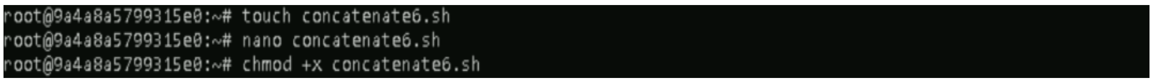


```
root@9a4a8a5799315e0:~# ./concatenate5.sh
Hello_World!
```

49. A Shell Program to demonstrate ‘String Concatenation’ using any character.

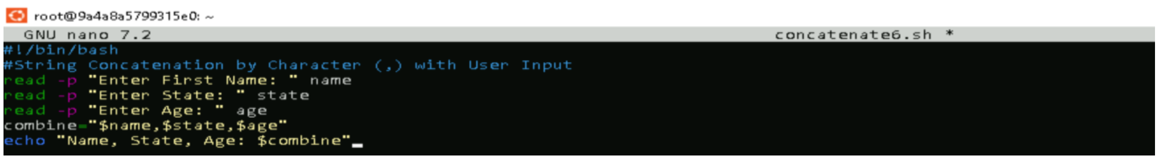
Step 1: Create an “concatenate6.sh” script file using touch command

Step 2: Create a nano file to write the code



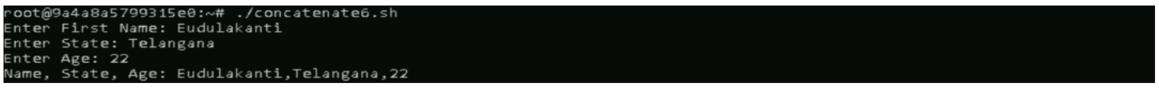
```
root@9a4a8a5799315e0:~# touch concatenate6.sh
root@9a4a8a5799315e0:~# nano concatenate6.sh
root@9a4a8a5799315e0:~# chmod +x concatenate6.sh
```

Step 3: Write the Code in nano concatenate6.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                   concatenate6.sh *
#!/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"
```

Step 4: Output



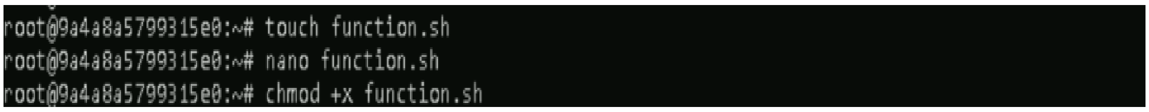
```
root@9a4a8a5799315e0:~# ./concatenate6.sh
Enter First Name: Eudulakanti
Enter State: Telangana
Enter Age: 22
Name, State, Age: Eudulakanti,Telangana,22
```

STRING FUNCTIONS

50. A Shell Program to demonstrate 'String Function' .

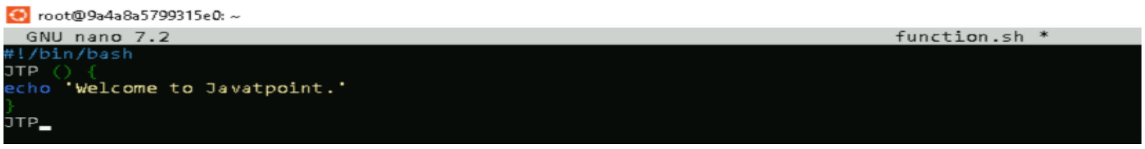
Step 1: Create an "function.sh" script file using touch command

Step 2: Create a nano file to write the code



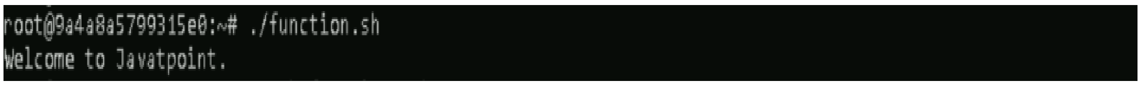
```
root@9a4a8a5799315e0:~# touch function.sh
root@9a4a8a5799315e0:~# nano function.sh
root@9a4a8a5799315e0:~# chmod +x function.sh
```

Step 3: Write the Code in nano concatenate6.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                    function.sh *
#!/bin/bash
JTP () {
echo 'Welcome to Javatpoint.'
}
JTP
```

Step 4: Output

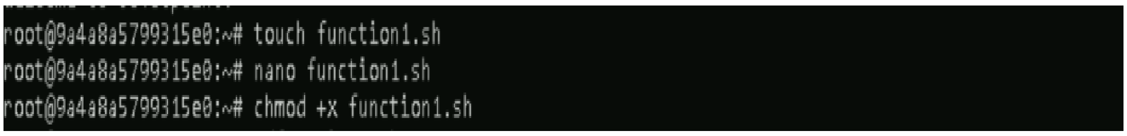


```
root@9a4a8a5799315e0:~# ./function.sh
Welcome to Javatpoint.
```

51. A Shell Program to demonstrate 'String Function' .

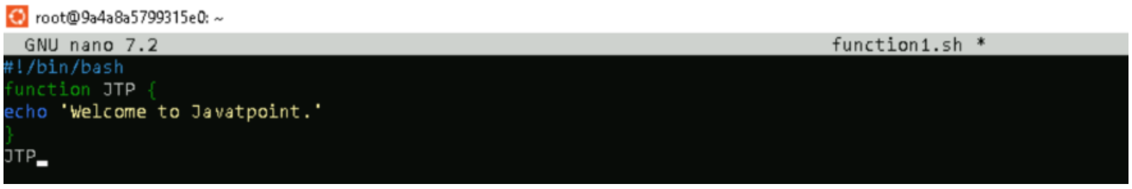
Step 1: Create an "function1.sh" script file using touch command

Step 2: Create a nano file to write the code



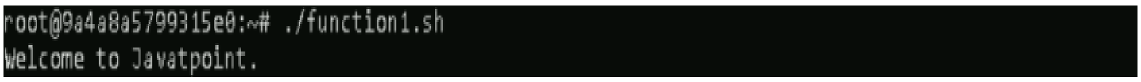
```
root@9a4a8a5799315e0:~# touch function1.sh
root@9a4a8a5799315e0:~# nano function1.sh
root@9a4a8a5799315e0:~# chmod +x function1.sh
```

Step 3: Write the Code in nano function1.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                function1.sh *
#!/bin/bash
function JTP {
echo 'Welcome to Javatpoint.'
}
JTP
```

Step 4: Output

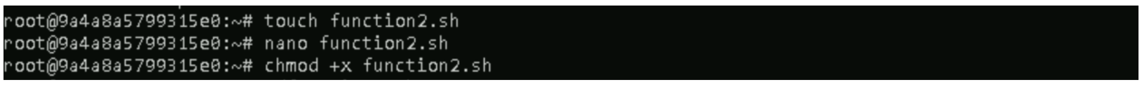


```
root@9a4a8a5799315e0:~# ./function1.sh
Welcome to Javatpoint.
```

52. A Shell Program to demonstrate 'String Functions by passing arguments'.

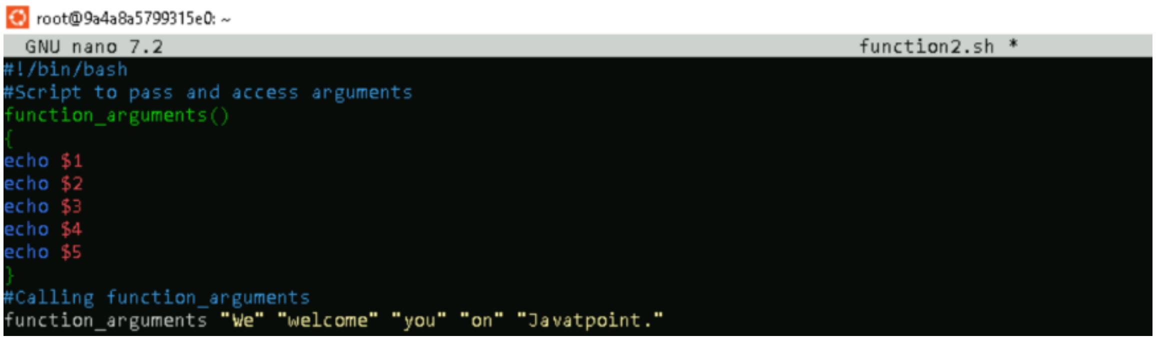
Step 1: Create an "function2.sh" script file using touch command

Step 2: Create a nano file to write the code



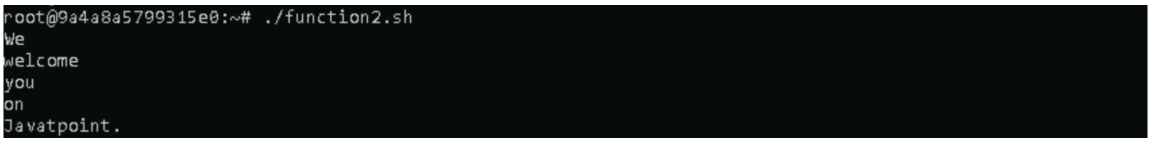
```
root@9a4a8a5799315e0:~# touch function2.sh
root@9a4a8a5799315e0:~# nano function2.sh
root@9a4a8a5799315e0:~# chmod +x function2.sh
```

Step 3: Write the Code in nano function2.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                  function2.sh *
#!/bin/bash
#Script to pass and access arguments
function_arguments()
{
echo $1
echo $2
echo $3
echo $4
echo $5
}
#Calling function_arguments
function_arguments "We" "welcome" "you" "on" "Javatpoint."
```

Step 4: Output

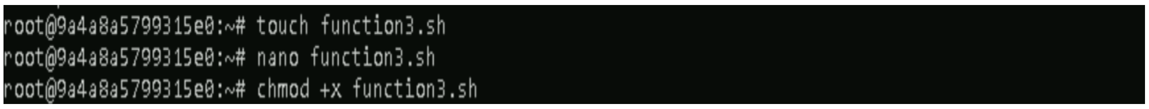


```
root@9a4a8a5799315e0:~# ./function2.sh
We
welcome
you
on
Javatpoint.
```

53. A Shell Program to demonstrate 'variable scope' in bash scripting.

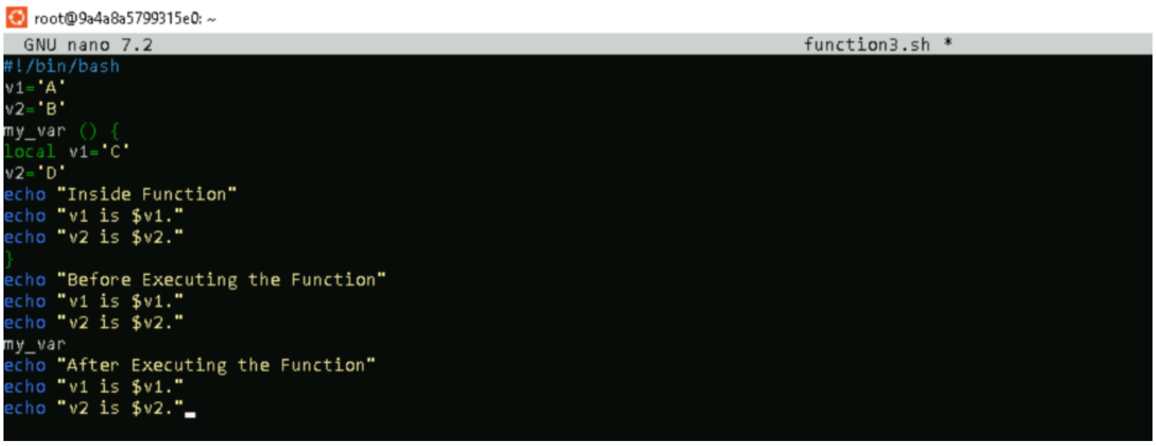
Step 1: Create an "function3.sh" script file using touch command

Step 2: Create a nano file to write the code



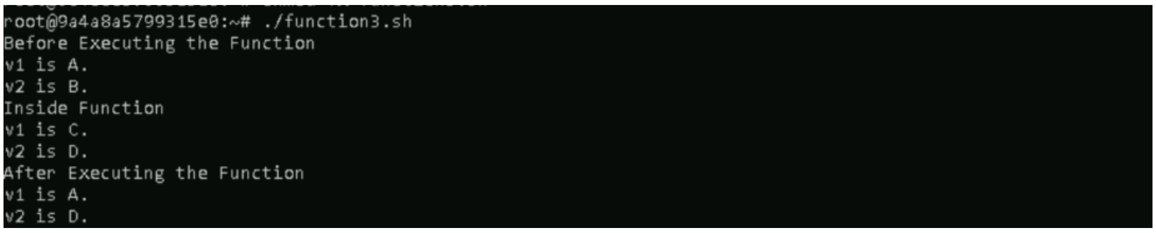
```
root@9a4a8a5799315e0:~# touch function3.sh
root@9a4a8a5799315e0:~# nano function3.sh
root@9a4a8a5799315e0:~# chmod +x function3.sh
```

Step 3: Write the Code in nano function3.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                              function3.sh *
#!/bin/bash
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."
```

Step 4: Output

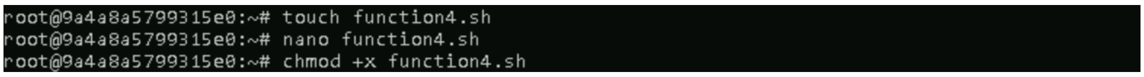


```
root@9a4a8a5799315e0:~# ./function3.sh
Before Executing the Function
v1 is A.
v2 is B.
Inside Function
v1 is C.
v2 is D.
After Executing the Function
v1 is A.
v2 is D.
```

54. A Shell Program to demonstrate 'String Function with return values' .

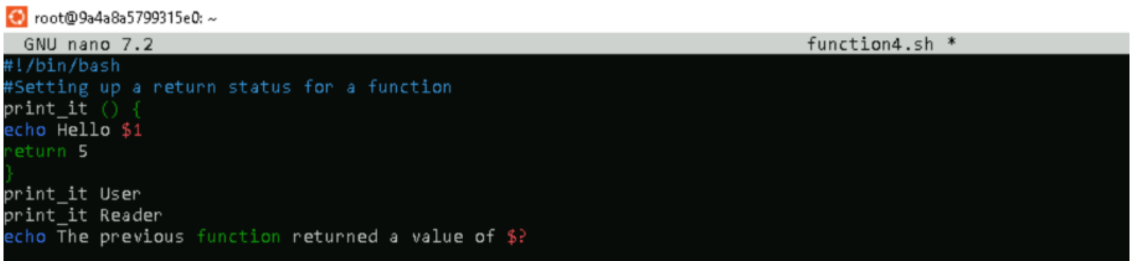
Step 1: Create an "function4.sh" script file using touch command

Step 2: Create a nano file to write the code



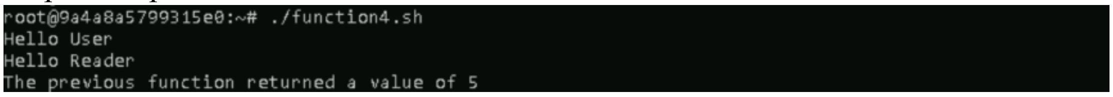
```
root@9a4a8a5799315e0:~# touch function4.sh
root@9a4a8a5799315e0:~# nano function4.sh
root@9a4a8a5799315e0:~# chmod +x function4.sh
```

Step 3: Write the Code in nano function4.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                  function4.sh *
#!/bin/bash
#Setting up a return status for a function
print_it () {
echo Hello $1
return 5
}
print_it User
print_it Reader
echo The previous function returned a value of $?
```

Step 4: Output

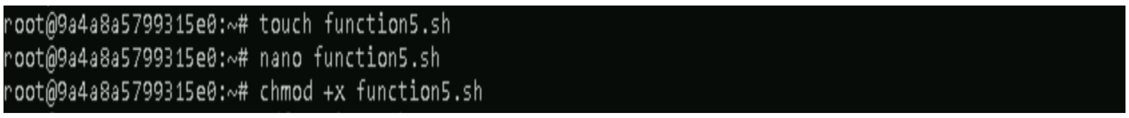


```
root@9a4a8a5799315e0:~# ./function4.sh
Hello User
Hello Reader
The previous function returned a value of 5
```

55. A Shell Program to demonstrate 'String Function with return values'.

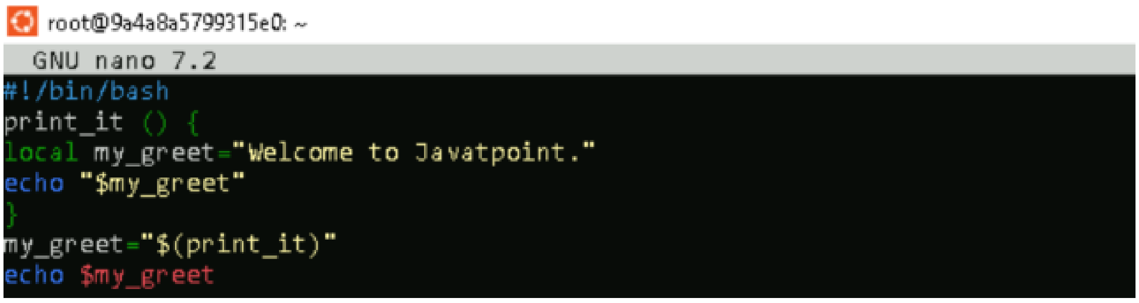
Step 1: Create an "function5.sh" script file using touch command

Step 2: Create a nano file to write the code



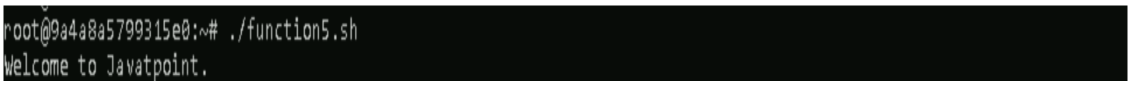
```
root@9a4a8a5799315e0:~# touch function5.sh
root@9a4a8a5799315e0:~# nano function5.sh
root@9a4a8a5799315e0:~# chmod +x function5.sh
```

Step 3: Write the Code in nano function5.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2
#!/bin/bash
print_it () {
local my_greet="Welcome to Javatpoint."
echo "$my_greet"
}
my_greet="$(print_it)"
echo $my_greet
```

Step 4: Output

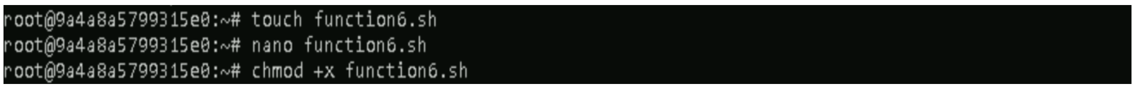


```
root@9a4a8a5799315e0:~# ./function5.sh
Welcome to Javatpoint.
```

56. A Shell Program to demonstrate 'String Function' to override the echo command and add the time stamp in the form of the argument to the echo command .

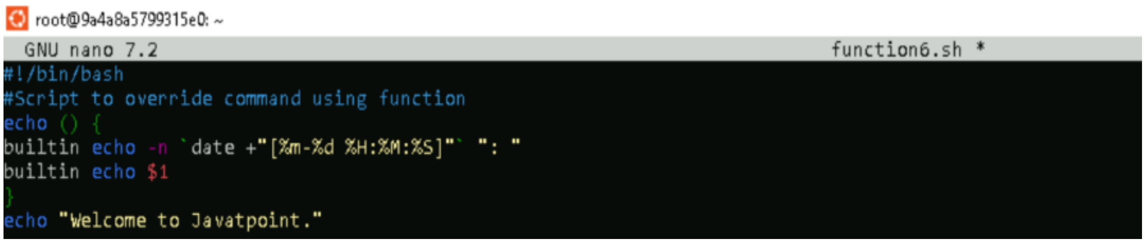
Step 1: Create an "function6.sh" script file using touch command

Step 2: Create a nano file to write the code



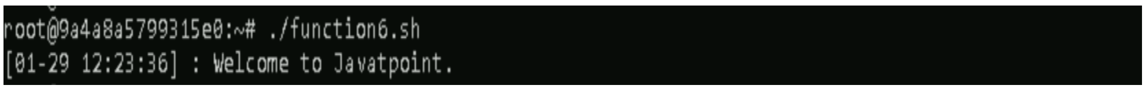
```
root@9a4a8a5799315e0:~# touch function6.sh
root@9a4a8a5799315e0:~# nano function6.sh
root@9a4a8a5799315e0:~# chmod +x function6.sh
```

Step 3: Write the Code in nano function6.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                function6.sh *
#!/bin/bash
#Script to override command using function
echo () {
builtin echo -n `date +"[%m-%d %H:%M:%S]"` ": "
builtin echo $1
}
echo "Welcome to Javatpoint."
```

Step 4: Output



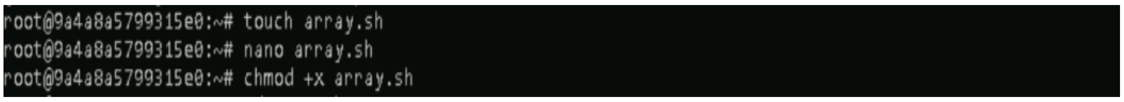
```
root@9a4a8a5799315e0:~# ./function6.sh
[01-29 12:23:36] : Welcome to Javatpoint.
```

ARRAY STRING

57. A Shell Program to print an element of an array with an index of 2.

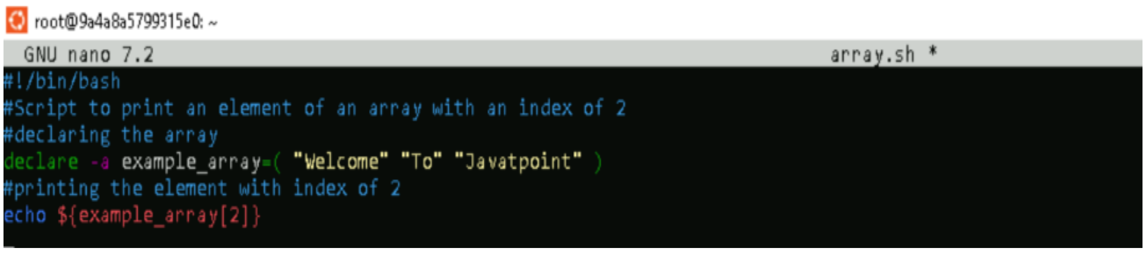
Step 1: Create an “array.sh” script file using touch command

Step 2: Create a nano file to write the code



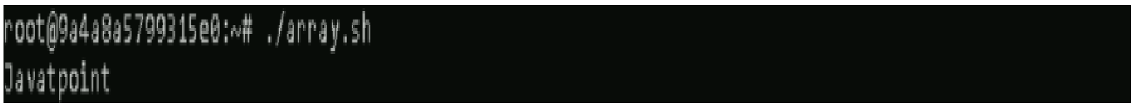
```
root@9a4a8a5799315e0:~# touch array.sh
root@9a4a8a5799315e0:~# nano array.sh
root@9a4a8a5799315e0:~# chmod +x array.sh
```

Step 3: Write the Code in nano array.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                array.sh *
#!/bin/bash
#Script to print an element of an array with an index of 2
#declaring the array
declare -a example_array=( "Welcome" "To" "Javatpoint" )
#printing the element with index of 2
echo ${example_array[2]}
```

Step 4: Output

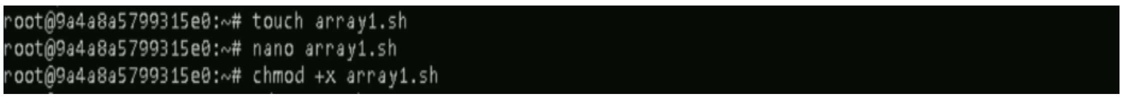


```
root@9a4a8a5799315e0:~# ./array.sh
Javatpoint
```

58. A Shell Program to print all the elements of the array.

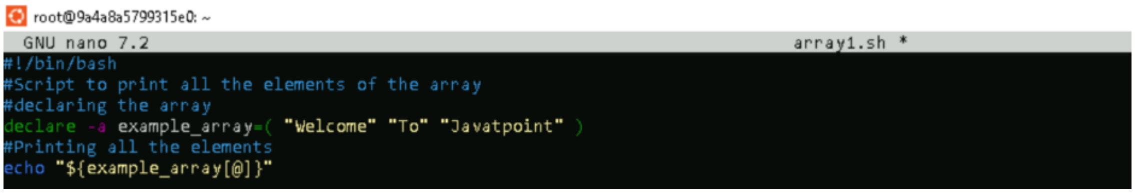
Step 1: Create an “array1.sh” script file using touch command

Step 2: Create a nano file to write the code



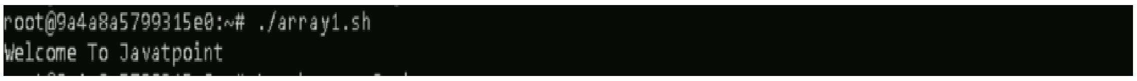
```
root@9a4a8a5799315e0:~# touch array1.sh
root@9a4a8a5799315e0:~# nano array1.sh
root@9a4a8a5799315e0:~# chmod +x array1.sh
```

Step 3: Write the Code in nano array1.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                    array1.sh *
#!/bin/bash
#Script to print all the elements of the array
#declaring the array
declare -a example_array=( "Welcome" "To" "Javatpoint" )
#Printing all the elements
echo "${example_array[@]}"
```

Step 4: Output

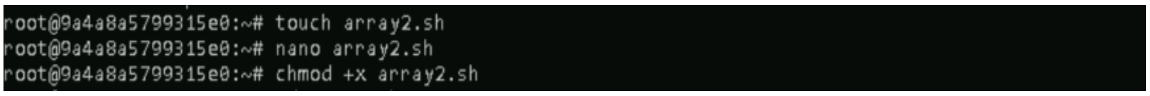


```
root@9a4a8a5799315e0:~# ./array1.sh
Welcome To Javatpoint
```

59. A Shell Program to print the keys of an array.

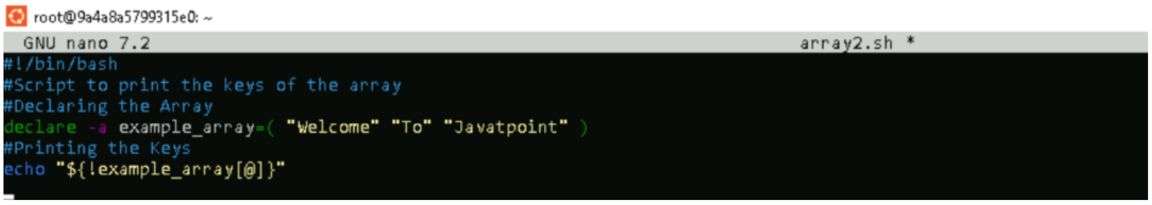
Step 1: Create an “array2.sh” script file using touch command

Step 2: Create a nano file to write the code



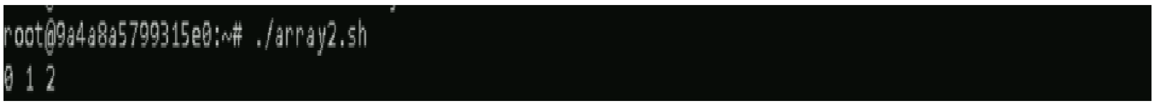
```
root@9a4a8a5799315e0:~# touch array2.sh
root@9a4a8a5799315e0:~# nano array2.sh
root@9a4a8a5799315e0:~# chmod +x array2.sh
```

Step 3: Write the Code in nano array2.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                    array2.sh *
#!/bin/bash
#Script to print the keys of the array
#Declaring the Array
declare -a example_array=( "Welcome" "To" "Javatpoint" )
#Printing the Keys
echo "${!example_array[@]}"
```

Step 4: Output

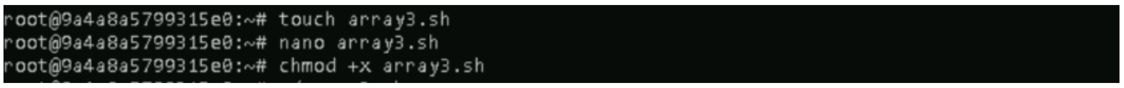


```
root@9a4a8a5799315e0:~# ./array2.sh
0 1 2
```

60. A Shell Program to count the number of elements contained in the array.

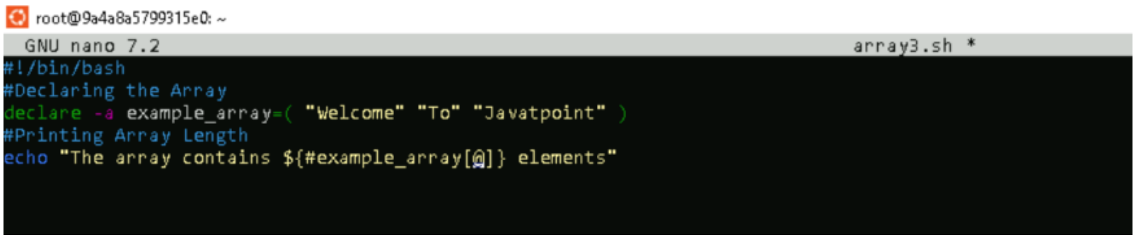
Step 1: Create an “array3.sh” script file using touch command

Step 2: Create a nano file to write the code



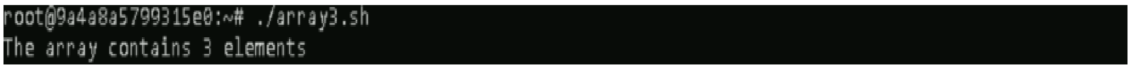
```
root@9a4a8a5799315e0:~# touch array3.sh
root@9a4a8a5799315e0:~# nano array3.sh
root@9a4a8a5799315e0:~# chmod +x array3.sh
```

Step 3: Write the Code in nano array3.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                    array3.sh *
#!/bin/bash
#Declaring the Array
declare -a example_array=( "Welcome" "To" "Javatpoint" )
#Printing Array Length
echo "The array contains ${#example_array[@]} elements"
```

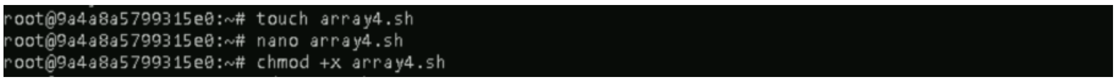
Step 4: Output


```
root@9a4a8a5799315e0:~# ./array3.sh
The array contains 3 elements
```

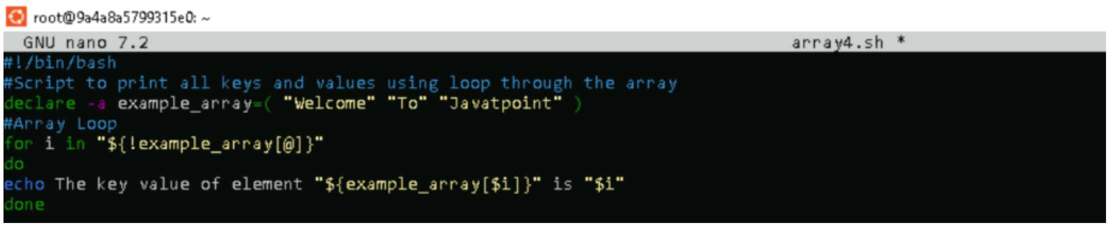
61. A Shell Program to iterate over each item in an array by using for loop.

Step 1: Create an “array4.sh” script file using touch command

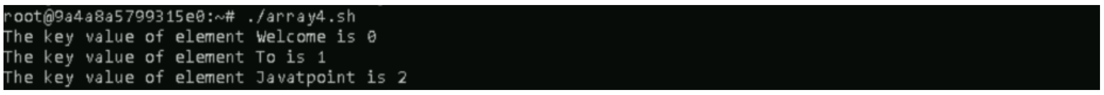
Step 2: Create a nano file to write the code


```
root@9a4a8a5799315e0:~# touch array4.sh
root@9a4a8a5799315e0:~# nano array4.sh
root@9a4a8a5799315e0:~# chmod +x array4.sh
```

Step 3: Write the Code in nano array4.sh script file


```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                              array4.sh *
#!/bin/bash
#Script to print all keys and values using loop through the array
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
```

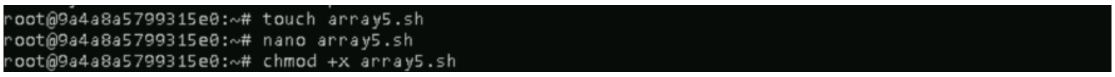
Step 4: Output


```
root@9a4a8a5799315e0:~# ./array4.sh
The key value of element Welcome is 0
The key value of element To is 1
The key value of element Javatpoint is 2
```

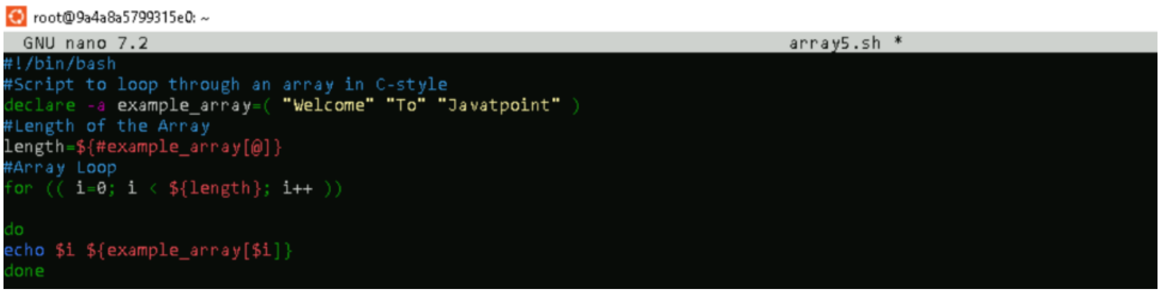
62. A Shell Program to loop through the array to retrieve the length of the array and use the C-style loop.

Step 1: Create an “array5.sh” script file using touch command

Step 2: Create a nano file to write the code

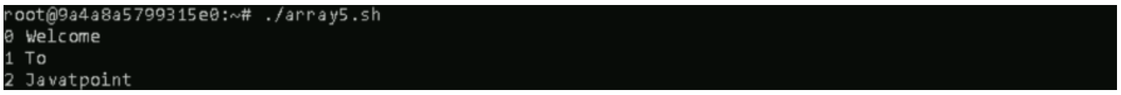

```
root@9a4a8a5799315e0:~# touch array5.sh
root@9a4a8a5799315e0:~# nano array5.sh
root@9a4a8a5799315e0:~# chmod +x array5.sh
```

Step 3: Write the Code in nano array5.sh script file


```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                              array5.sh *
#!/bin/bash
#Script to loop through an array in C-style
declare -a example_array=( "Welcome" "To" "Javatpoint" )
#Length of the Array
length=${#example_array[@]}
#Array Loop
for (( i=0; i < ${length}; i++ ))

do
echo $i ${example_array[$i]}
done
```

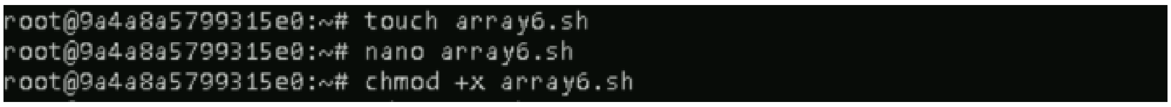
Step 4: Output


```
root@9a4a8a5799315e0:~# ./array5.sh
0 Welcome
1 To
2 Javatpoint
```

63. A Shell Program to add new element to an array in bash script.

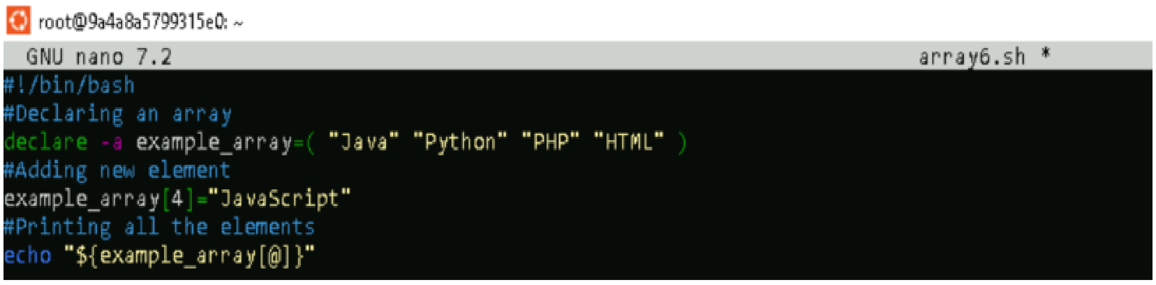
Step 1: Create an “array6.sh” script file using touch command

Step 2: Create a nano file to write the code



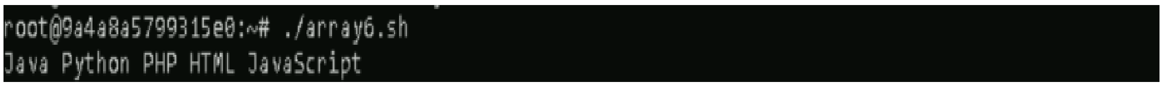
```
root@9a4a8a5799315e0:~# touch array6.sh
root@9a4a8a5799315e0:~# nano array6.sh
root@9a4a8a5799315e0:~# chmod +x array6.sh
```

Step 3: Write the Code in nano array6.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                    array6.sh *
#!/bin/bash
#Declaring an array
declare -a example_array=( "Java" "Python" "PHP" "HTML" )
#Adding new element
example_array[4]="JavaScript"
#Printing all the elements
echo "${example_array[@]}"
```

Step 4: Output

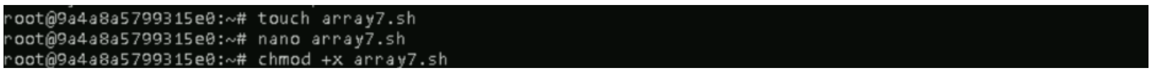


```
root@9a4a8a5799315e0:~# ./array6.sh
Java Python PHP HTML JavaScript
```

64. A Shell Program to add one or multiple elements in the array in bash script.

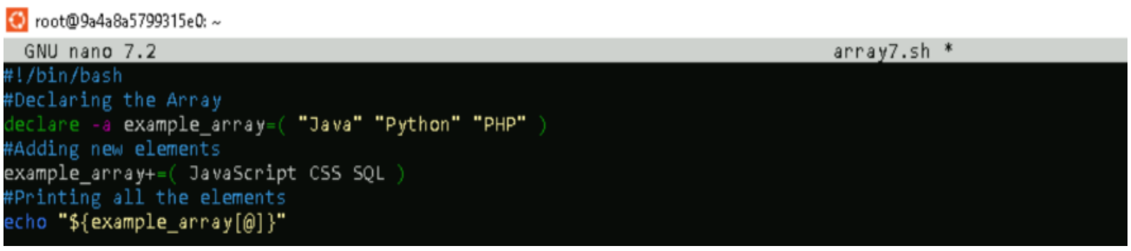
Step 1: Create an “array7.sh” script file using touch command

Step 2: Create a nano file to write the code



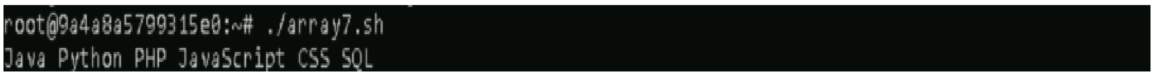
```
root@9a4a8a5799315e0:~# touch array7.sh
root@9a4a8a5799315e0:~# nano array7.sh
root@9a4a8a5799315e0:~# chmod +x array7.sh
```

Step 3: Write the Code in nano array7.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                              array7.sh *
#!/bin/bash
#Declaring the Array
declare -a example_array=( "Java" "Python" "PHP" )
#Adding new elements
example_array+=( JavaScript CSS SQL )
#Printing all the elements
echo "${example_array[@]}"
```

Step 4: Output

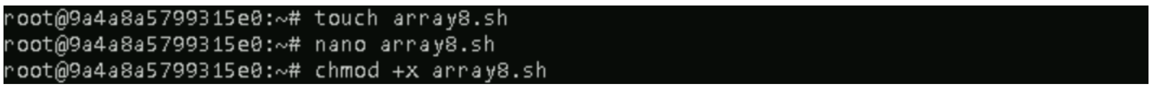


```
root@9a4a8a5799315e0:~# ./array7.sh
Java Python PHP JavaScript CSS SQL
```

65. A Shell Program to update the array element by assigning a new value to the existing array by its index value.

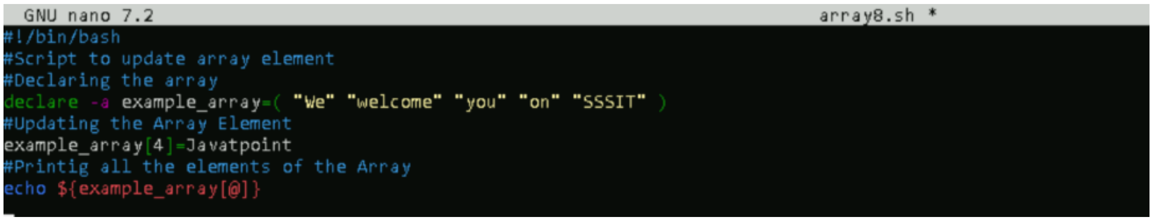
Step 1: Create an “array8.sh” script file using touch command

Step 2: Create a nano file to write the code



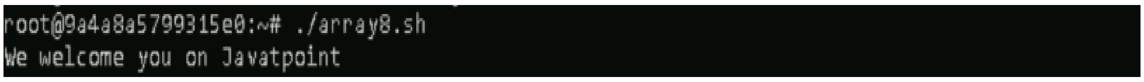
```
root@9a4a8a5799315e0:~# touch array8.sh
root@9a4a8a5799315e0:~# nano array8.sh
root@9a4a8a5799315e0:~# chmod +x array8.sh
```

Step 3: Write the Code in nano array8.sh script file



```
  GNU nano 7.2                                                   array8.sh *
#!/bin/bash
#Script to update array element
#Declaring the array
declare -a example_array=( "We" "welcome" "you" "on" "SSSIT" )
#Updating the Array Element
example_array[4]=Javatpoint
#Printig all the elements of the Array
echo ${example_array[@]}
```

Step 4: Output

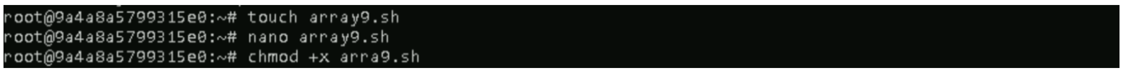


```
root@9a4a8a5799315e0:~# ./array8.sh
We welcome you on Javatpoint
```

66. A Shell Program to delete the element from the array.

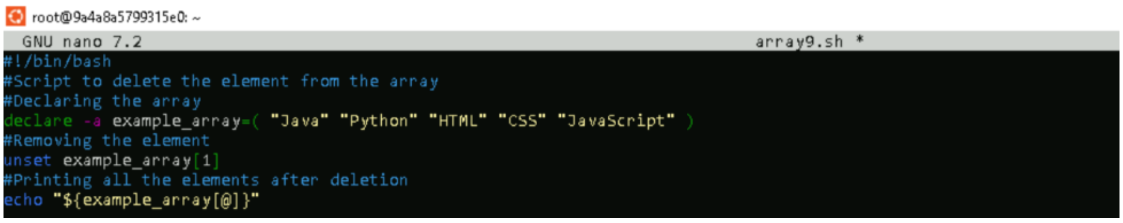
Step 1: Create an “array9.sh” script file using touch command

Step 2: Create a nano file to write the code



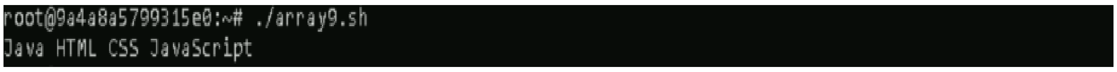
```
root@9a4a8a5799315e0:~# touch array9.sh
root@9a4a8a5799315e0:~# nano array9.sh
root@9a4a8a5799315e0:~# chmod +x arra9.sh
```

Step 3: Write the Code in nano array9.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                              array9.sh *
#!/bin/bash
#Script to delete the element from the array
#Declaring the array
declare -a example_array=( "Java" "Python" "HTML" "CSS" "JavaScript" )
#Removing the element
unset example_array[1]
#Printing all the elements after deletion
echo "${example_array[@]}"
```

Step 4: Output

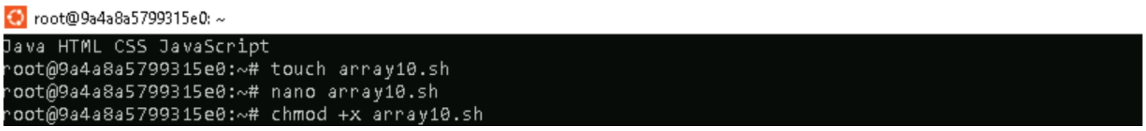


```
root@9a4a8a5799315e0:~# ./array9.sh
Java HTML CSS JavaScript
```

67. A Shell Program to delete the entire array in bash script.

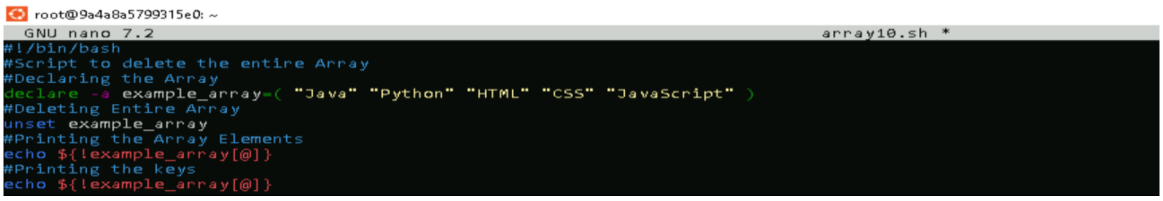
Step 1: Create an “array10.sh” script file using touch command

Step 2: Create a nano file to write the code



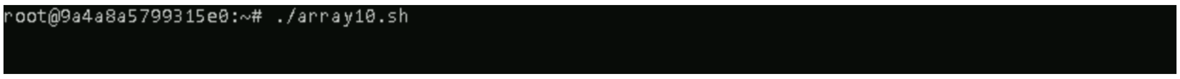
```
root@9a4a8a5799315e0: ~
Java HTML CSS JavaScript
root@9a4a8a5799315e0:~# touch array10.sh
root@9a4a8a5799315e0:~# nano array10.sh
root@9a4a8a5799315e0:~# chmod +x array10.sh
```

Step 3: Write the Code in nano array10.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                   array10.sh *
#!/bin/bash
#Script to delete the entire Array
#Declaring the Array
declare -a example_array=( "Java" "Python" "HTML" "CSS" "JavaScript" )
#Deleting Entire Array
unset example_array
#Printing the Array Elements
echo ${!example_array[@]}
#Printing the keys
echo ${!example_array[@]}
```

Step 4: Output

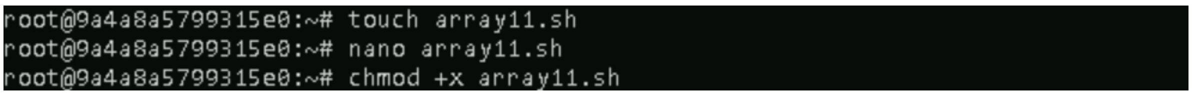


```
root@9a4a8a5799315e0:~# ./array10.sh
```

68. A Shell Program to slice the array elements from a given start index to the ending index.

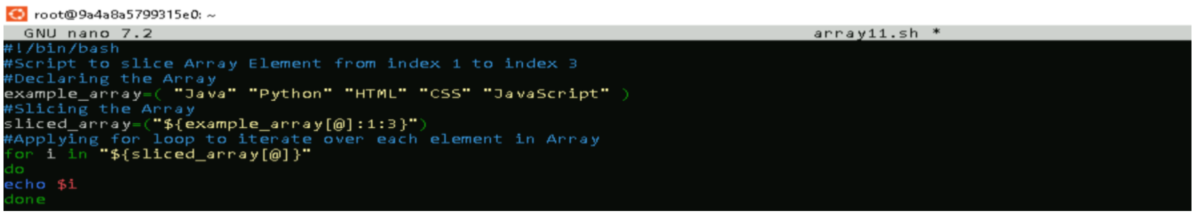
Step 1: Create an “array11.sh” script file using touch command

Step 2: Create a nano file to write the code



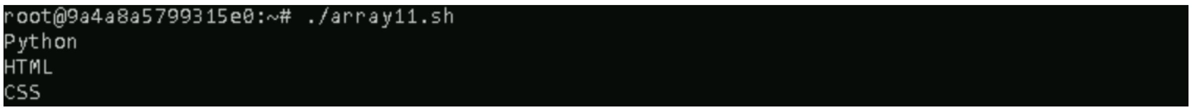
```
root@9a4a8a5799315e0:~# touch array11.sh
root@9a4a8a5799315e0:~# nano array11.sh
root@9a4a8a5799315e0:~# chmod +x array11.sh
```

Step 3: Write the Code in nano array11.sh script file



```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                                      array11.sh *
#!/bin/bash
#Script to slice Array Element from index 1 to index 3
#Declaring the Array
example_array=( "Java" "Python" "HTML" "CSS" "JavaScript" )
#Slicing the Array
sliced_array=("${example_array[@]:1:3}")
#Applying for loop to iterate over each element in Array
for i in "${sliced_array[@]}"
do
echo $i
done
```

Step 4: Output



```
root@9a4a8a5799315e0:~# ./array11.sh
Python
HTML
CSS
```