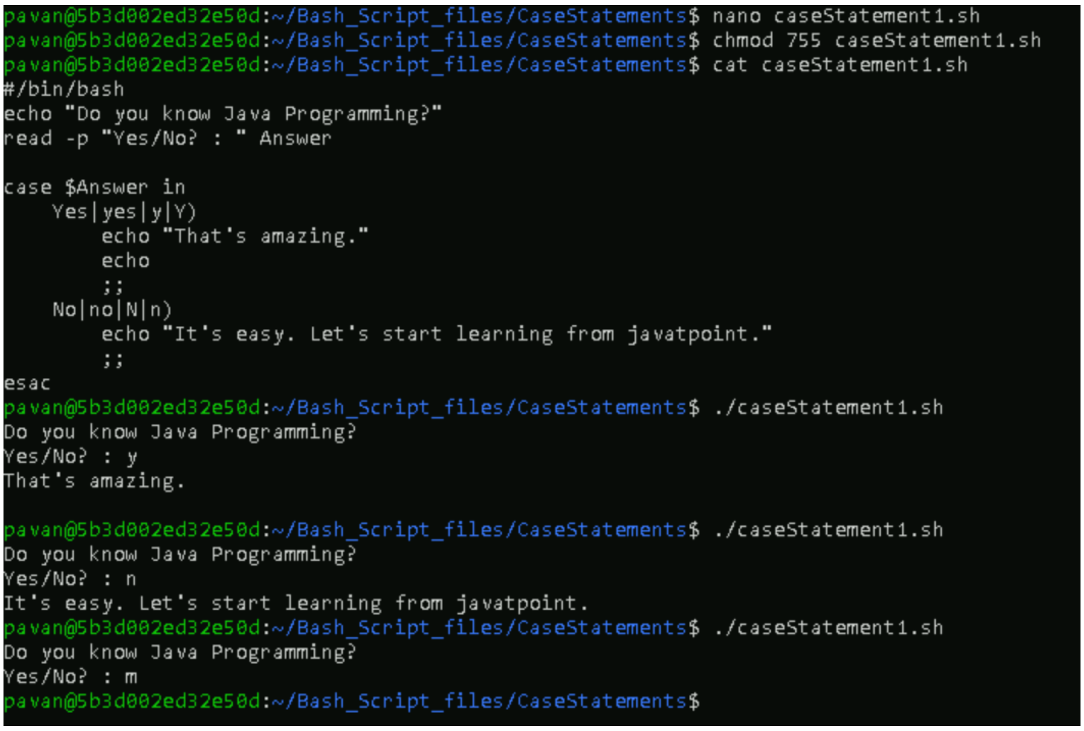
Thumati Pavan Venkata Narendra Kumar

EMP-ID-289219

BASH-Project-1

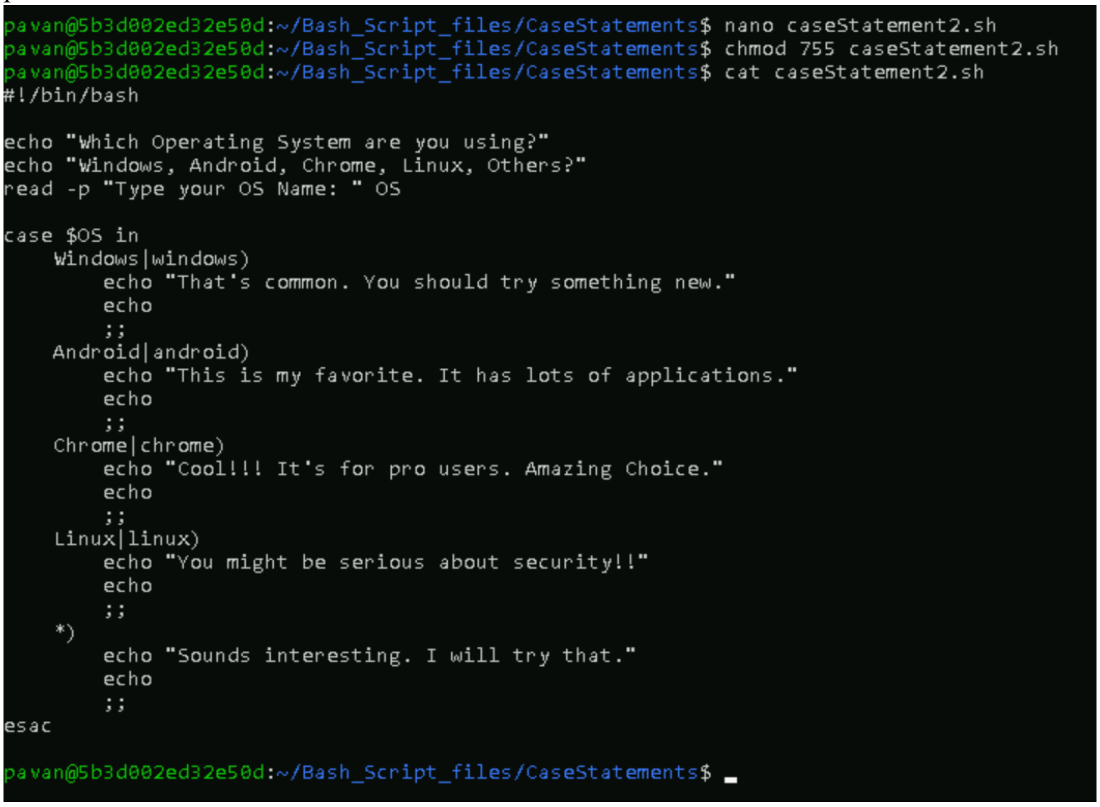
1. Bash CASE Statements.

1. To demonstrate the use of the case statement.

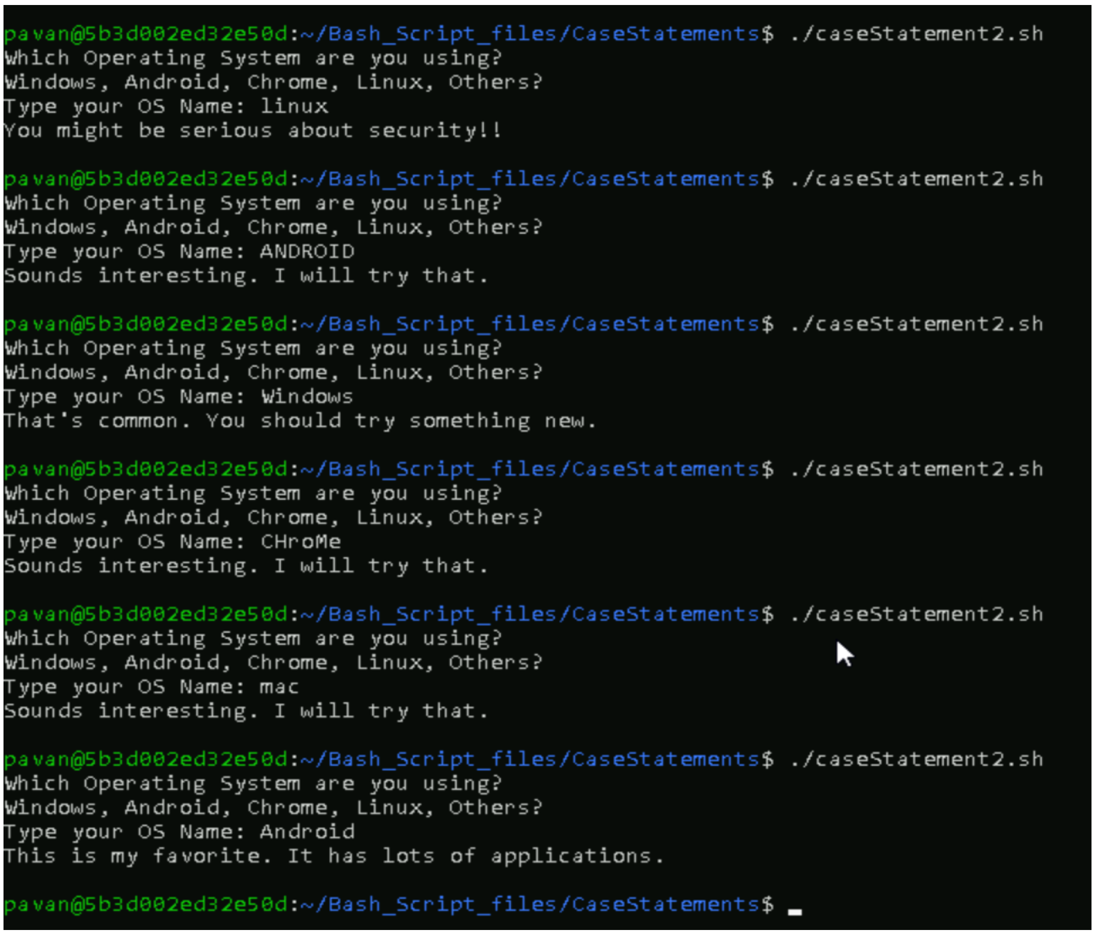


```
pavan@5b3d002ed32e50d:~/Bash_Script_files/CaseStatements$ nano caseStatement1.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/CaseStatements$ chmod 755 caseStatement1.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/CaseStatements$ cat caseStatement1.sh
#/bin/bash
echo "Do you know Java Programming?"
read -p "Yes/No? : " Answer

case $Answer in
    Yes|yes|y|Y)
        echo "That's amazing."
        echo
        ;;
    No|no|N|n)
        echo "It's easy. Let's start learning from javatpoint."
        ;;
esac
pavan@5b3d002ed32e50d:~/Bash_Script_files/CaseStatements$ ./caseStatement1.sh
Do you know Java Programming?
Yes/No? : y
That's amazing.

pavan@5b3d002ed32e50d:~/Bash_Script_files/CaseStatements$ ./caseStatement1.sh
Do you know Java Programming?
Yes/No? : n
It's easy. Let's start learning from javatpoint.
pavan@5b3d002ed32e50d:~/Bash_Script_files/CaseStatements$ ./caseStatement1.sh
Do you know Java Programming?
Yes/No? : m
pavan@5b3d002ed32e50d:~/Bash_Script_files/CaseStatements$
```

2. To demonstrate combined scenario where there is also a default case when no previous matched case is found.



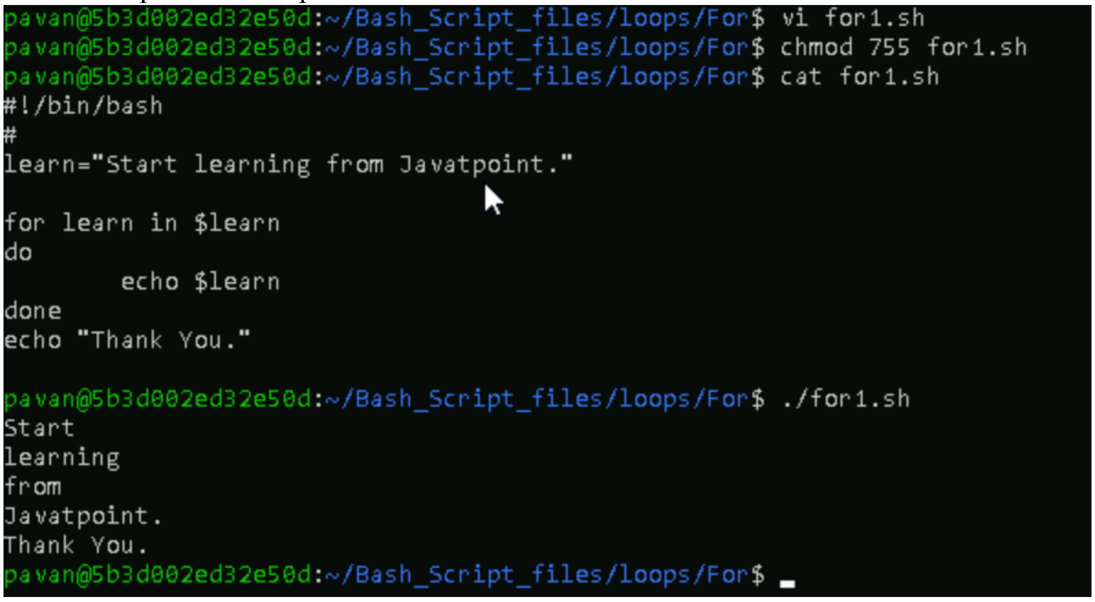
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/CaseStatements$ nano caseStatement2.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/CaseStatements$ chmod 755 caseStatement2.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/CaseStatements$ cat caseStatement2.sh
#!/bin/bash

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

pavan@5b3d002ed32e50d:~/Bash_Script_files/CaseStatements$ _
```



```
pavan@5b3d002ed32e50d:~/Bash_Script_files/CaseStatements$ ./caseStatement2.sh
Which Operating System are you using?
Windows, Android, Chrome, Linux, Others?
Type your OS Name: linux
You might be serious about security!!

pavan@5b3d002ed32e50d:~/Bash_Script_files/CaseStatements$ ./caseStatement2.sh
Which Operating System are you using?
Windows, Android, Chrome, Linux, Others?
Type your OS Name: ANDROID
Sounds interesting. I will try that.

pavan@5b3d002ed32e50d:~/Bash_Script_files/CaseStatements$ ./caseStatement2.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.

pavan@5b3d002ed32e50d:~/Bash_Script_files/CaseStatements$ ./caseStatement2.sh
Which Operating System are you using?
Windows, Android, Chrome, Linux, Others?
Type your OS Name: CHroMe
Sounds interesting. I will try that.

pavan@5b3d002ed32e50d:~/Bash_Script_files/CaseStatements$ ./caseStatement2.sh
Which Operating System are you using?
Windows, Android, Chrome, Linux, Others?
Type your OS Name: mac
Sounds interesting. I will try that.

pavan@5b3d002ed32e50d:~/Bash_Script_files/CaseStatements$ ./caseStatement2.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.

pavan@5b3d002ed32e50d:~/Bash_Script_files/CaseStatements$ _
```

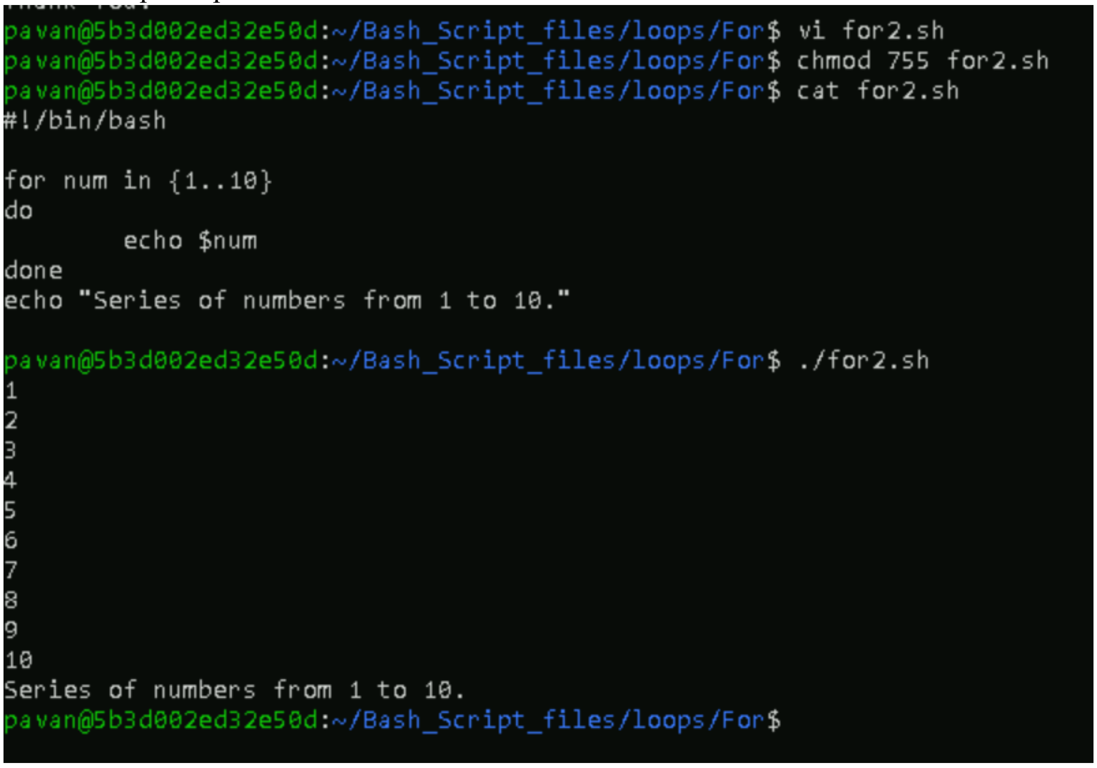
2. **Bash FOR loop.**

   1. Basic example of 'for loop'.



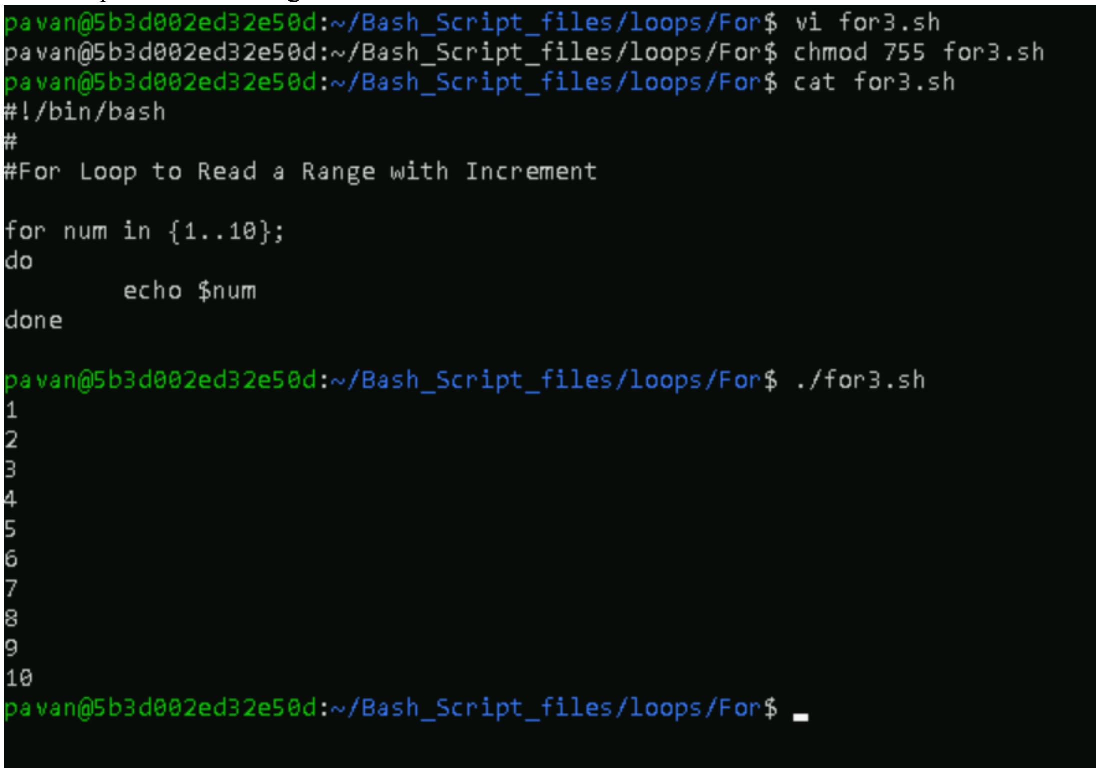
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/For$ vi for1.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/For$ chmod 755 for1.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/For$ cat for1.sh
#!/bin/bash
#
learn="Start learning from Javatpoint."

for learn in $learn
do
        echo $learn
done
echo "Thank You."

pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/For$ ./for1.sh
Start
learning
from
Javatpoint.
Thank You.
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/For$ _
```

   2. Basic example to print a series of numbers from 1 t o 10.



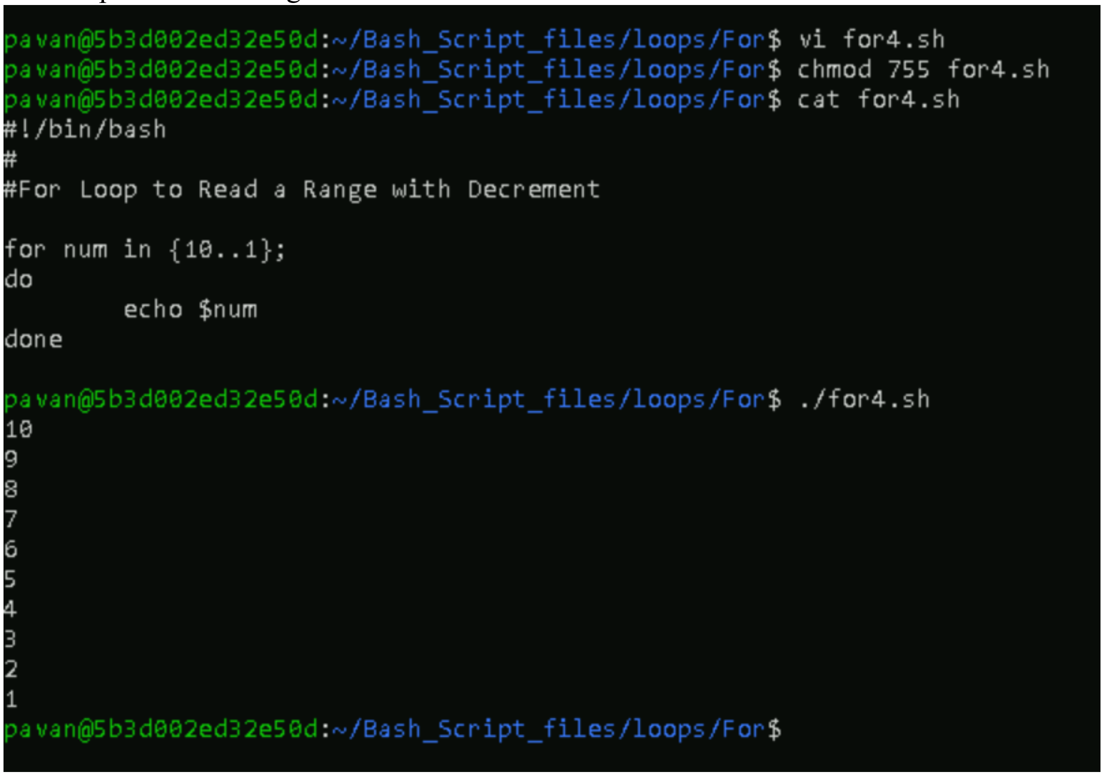
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/For$ vi for2.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/For$ chmod 755 for2.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/For$ cat for2.sh
#!/bin/bash

for num in {1..10}
do
        echo $num
done
echo "Series of numbers from 1 to 10."

pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/For$ ./for2.sh
1
2
3
4
5
6
7
8
9
10
Series of numbers from 1 to 10.
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/For$
```

3. For Loop to Read a Range with Increment.



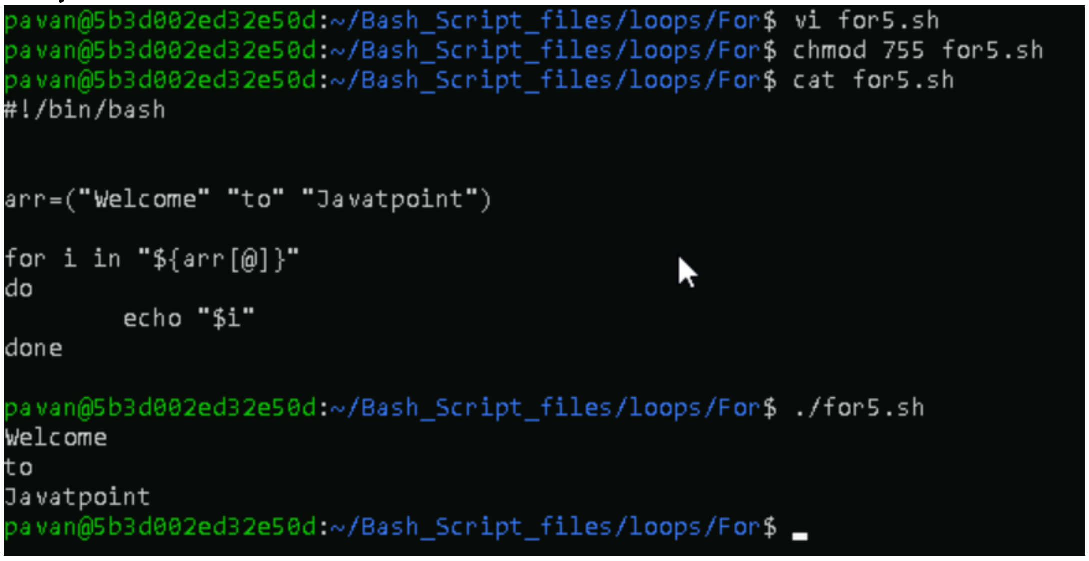
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/For$ vi for3.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/For$ chmod 755 for3.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/For$ cat for3.sh
#!/bin/bash
#
#For Loop to Read a Range with Increment

for num in {1..10};
do
        echo $num
done

pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/For$ ./for3.sh
1
2
3
4
5
6
7
8
9
10
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/For$ _
```

4. For Loop to Read a Range with Decrement.



```
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/For$ vi for4.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/For$ chmod 755 for4.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/For$ cat for4.sh
#!/bin/bash
#
#For Loop to Read a Range with Decrement

for num in {10..1};
do
        echo $num
done

pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/For$ ./for4.sh
10
9
8
7
6
5
4
3
2
1
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/For$
```

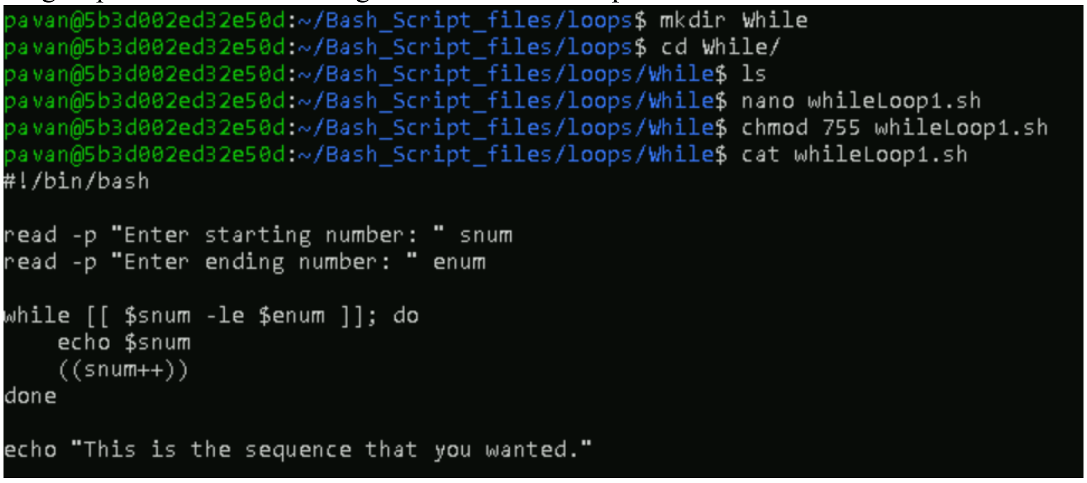
5. Array Declaration.



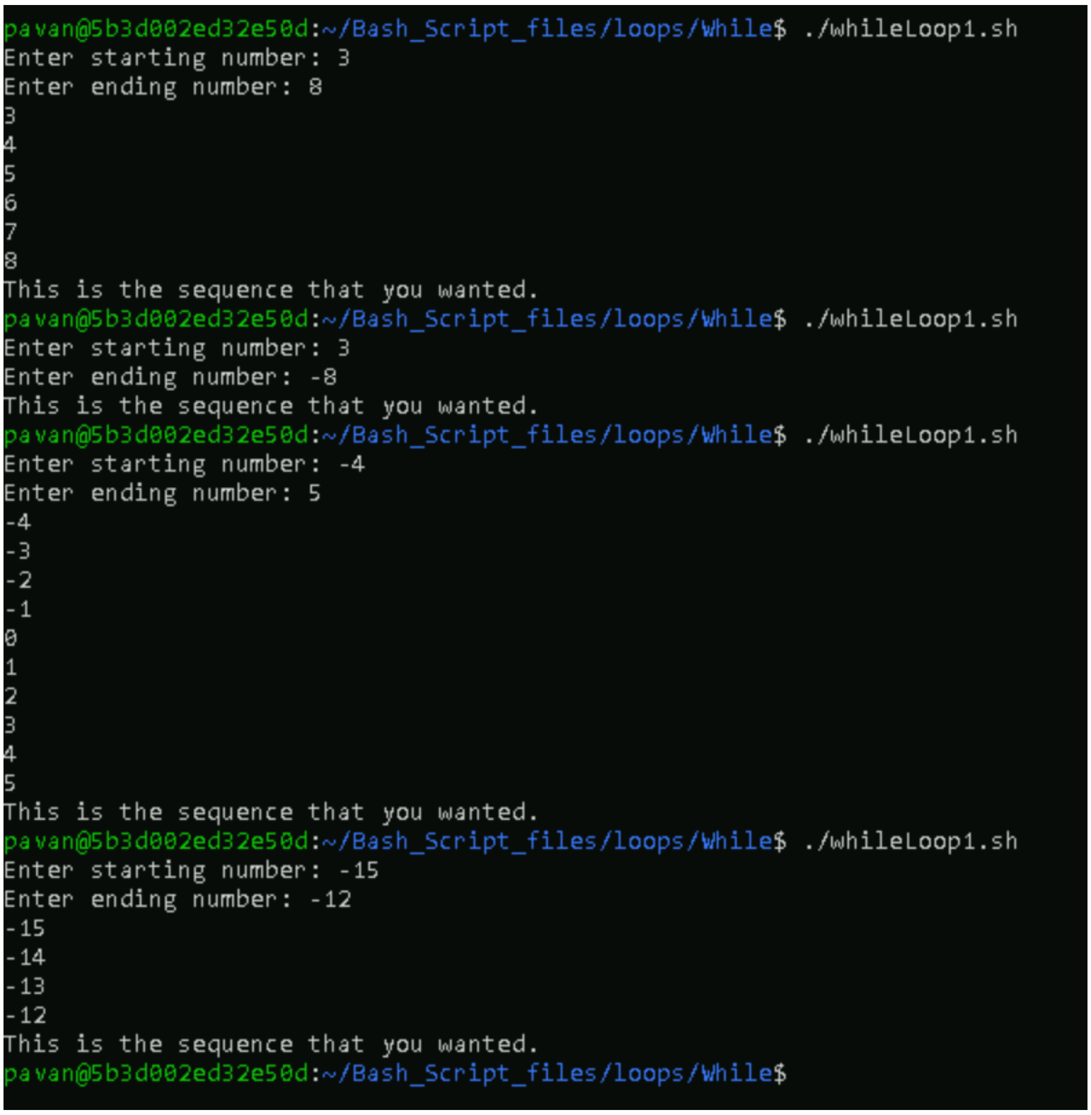
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/For$ vi for5.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/For$ chmod 755 for5.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/For$ cat for5.sh
#!/bin/bash


arr=("Welcome" "to" "Javatpoint")

for i in "${arr[@]}"
do
        echo "$i"
done

pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/For$ ./for5.sh
Welcome
to
Javatpoint
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/For$
```

3.BASH WHILE Loop.

1. To get specified numbers using BASH WHILE Loop.

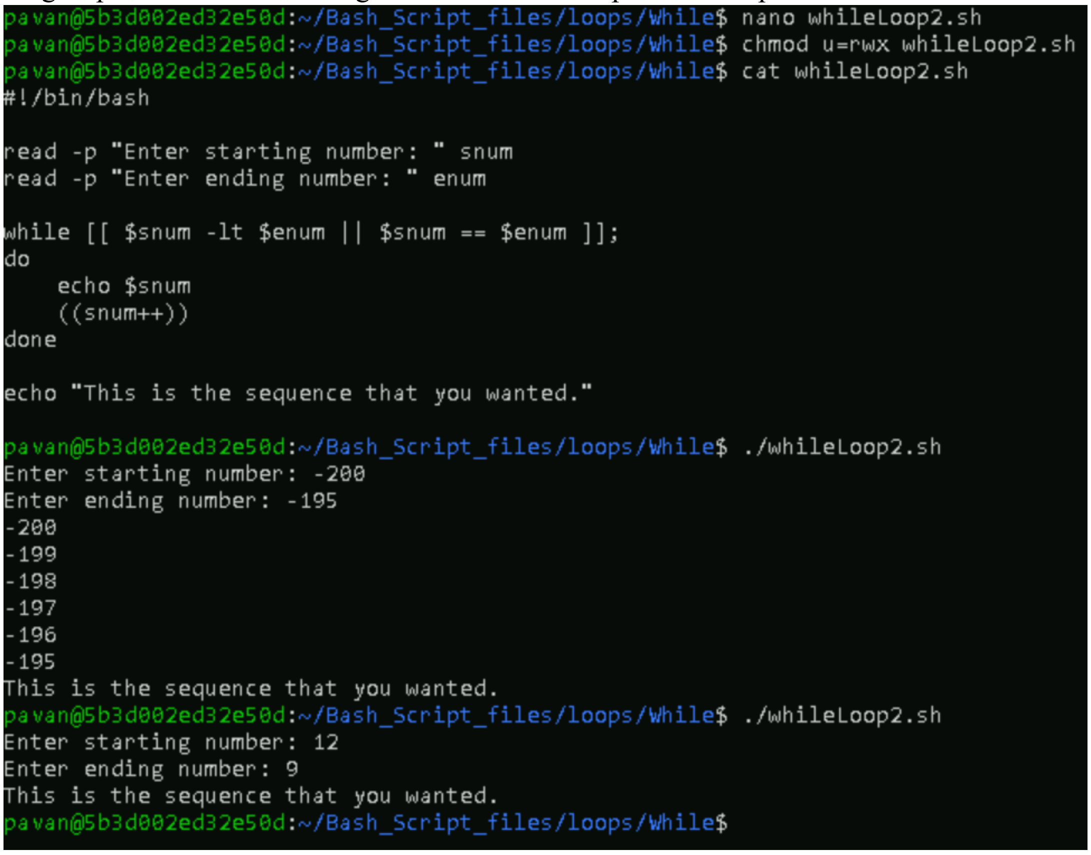


```
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops$ mkdir While
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops$ cd While/
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$ ls
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$ nano whileLoop1.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$ chmod 755 whileLoop1.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$ cat whileLoop1.sh
#!/bin/bash

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."
```



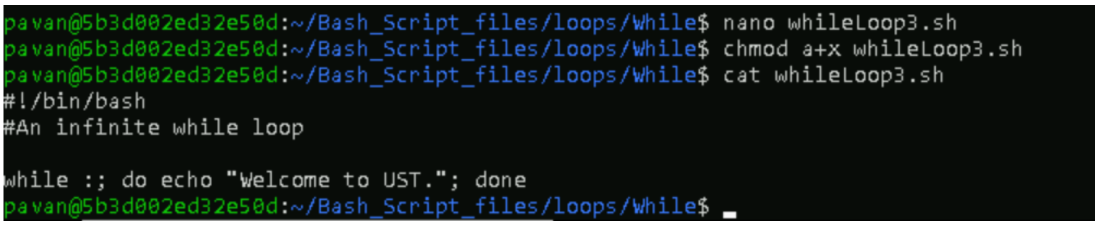
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$ ./whileLoop1.sh
Enter starting number: 3
Enter ending number: 8
3
4
5
6
7
8
This is the sequence that you wanted.
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$ ./whileLoop1.sh
Enter starting number: 3
Enter ending number: -8
This is the sequence that you wanted.
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$ ./whileLoop1.sh
Enter starting number: -4
Enter ending number: 5
-4
-3
-2
-1
0
1
2
3
4
5
This is the sequence that you wanted.
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$ ./whileLoop1.sh
Enter starting number: -15
Enter ending number: -12
-15
-14
-13
-12
This is the sequence that you wanted.
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$
```

2. To get specified numbers using BASH WHILE Loop with Multiple Conditions.

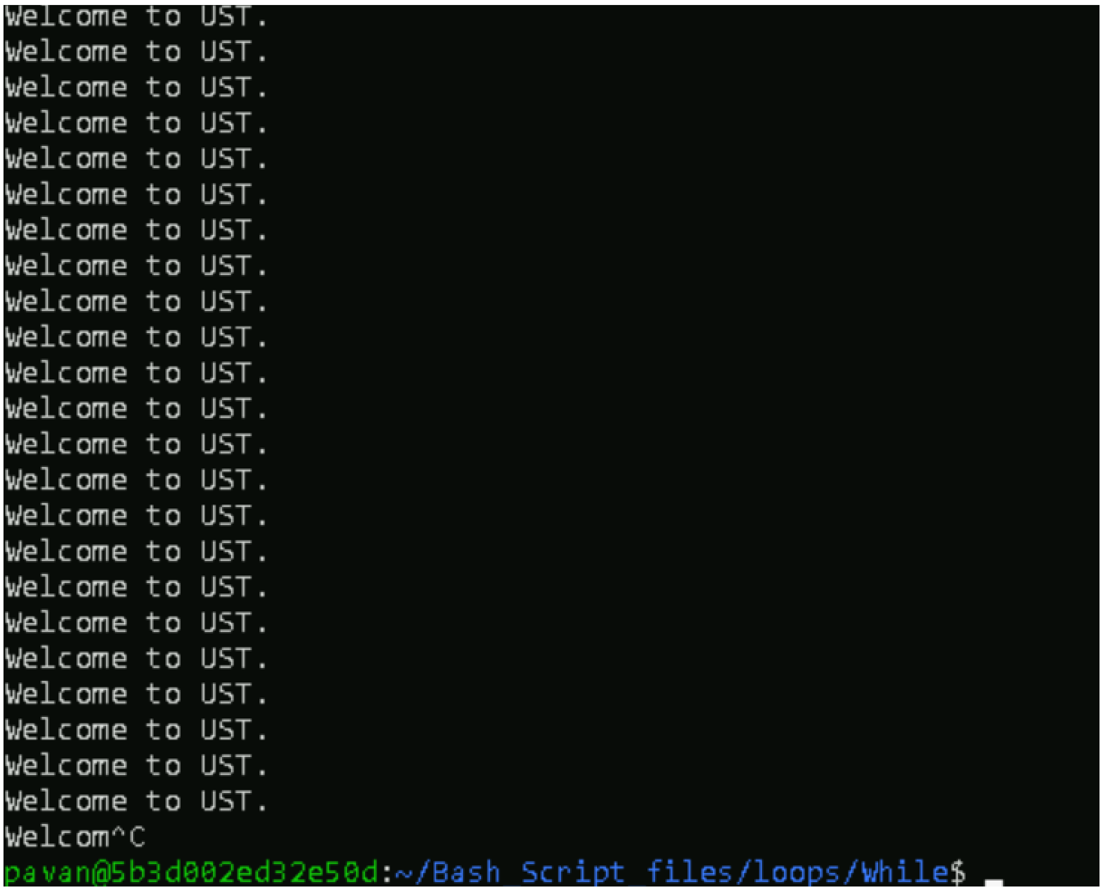


```
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$ nano whileLoop2.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$ chmod u=rwx whileLoop2.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$ cat whileLoop2.sh
#!/bin/bash

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."

pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$ ./whileLoop2.sh
Enter starting number: -200
Enter ending number: -195
-200
-199
-198
-197
-196
-195
This is the sequence that you wanted.
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$ ./whileLoop2.sh
Enter starting number: 12
Enter ending number: 9
This is the sequence that you wanted.
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$
```

3. An infinite while loop in single line & using CTRL + C to forcefully stop the execution.

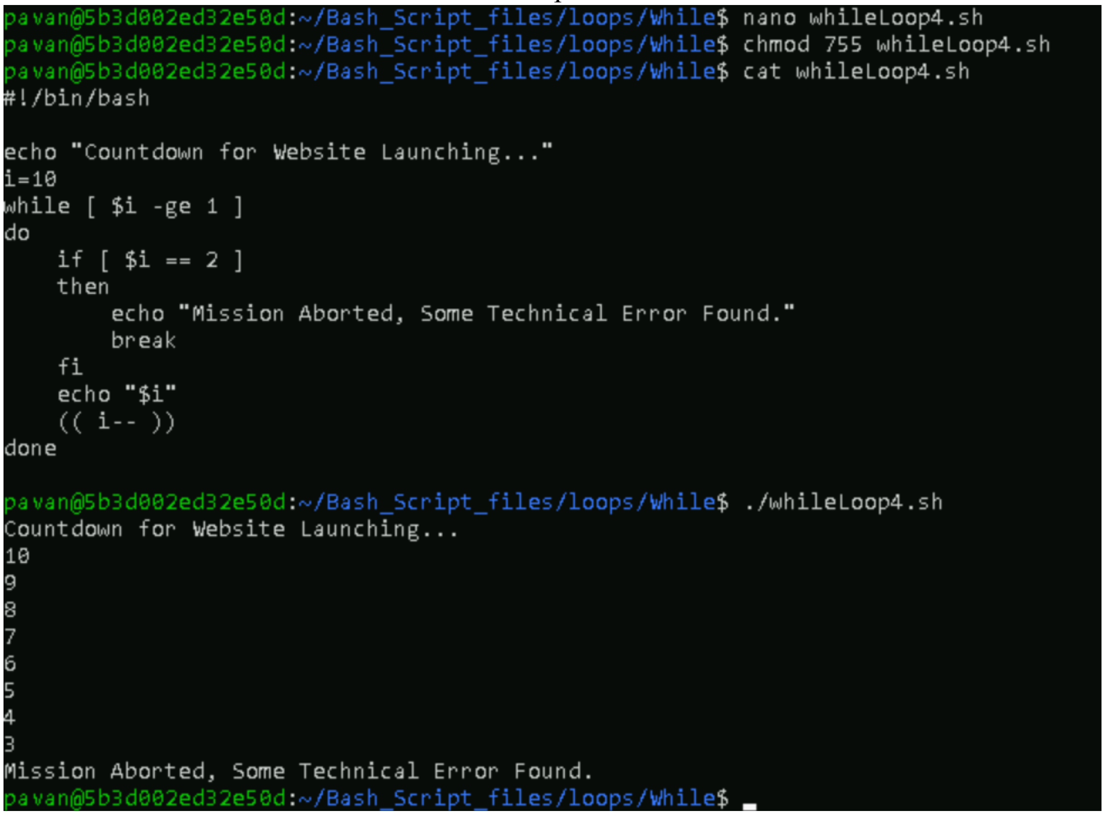


```
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$ nano whileLoop3.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$ chmod a+x whileLoop3.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$ cat whileLoop3.sh
#!/bin/bash
#An infinite while loop

while :; do echo "Welcome to UST."; done
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$ _
```



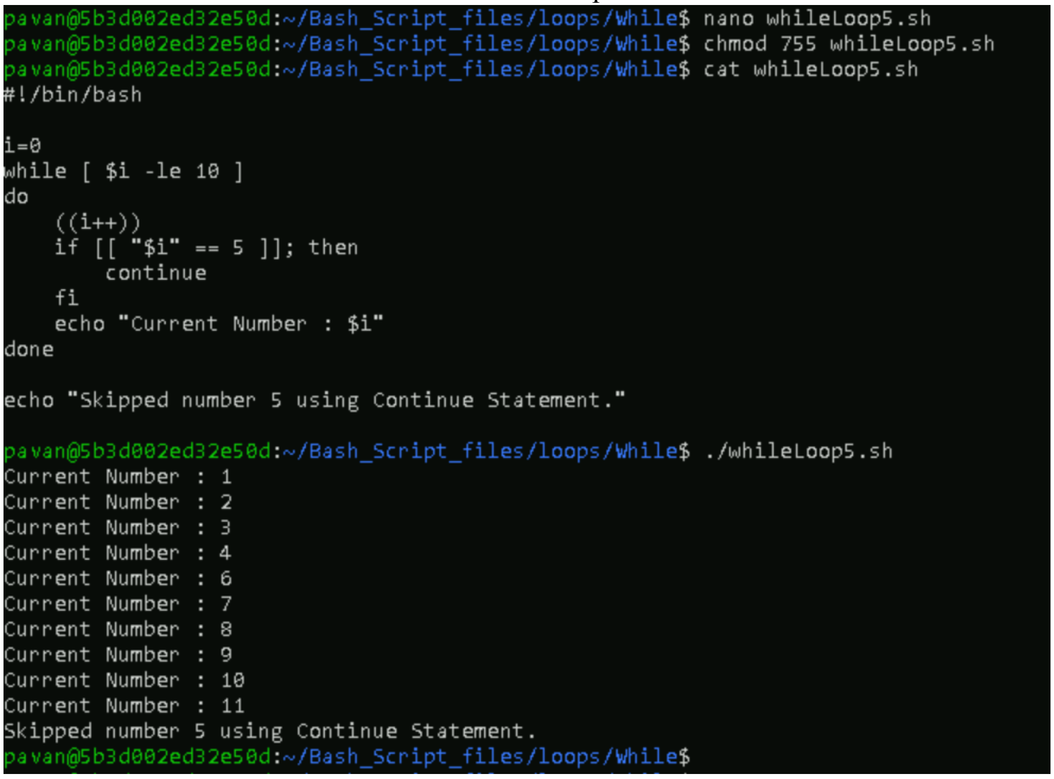
```
Welcome to UST.
Welcome to UST.
Welcome to UST.
Welcome to UST.
Welcome to UST.
Welcome to UST.
Welcome to UST.
Welcome to UST.
Welcome to UST.
Welcome to UST.
Welcome to UST.
Welcome to UST.
Welcome to UST.
Welcome to UST.
Welcome to UST.
Welcome to UST.
Welcome to UST.
Welcome to UST.
Welcome to UST.
Welcome to UST.
Welcome to UST.
Welcome to UST.
Welcome to UST.
Welcom^C
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$ _
```

4. To use Break Statement in BASH WHILE Loop.



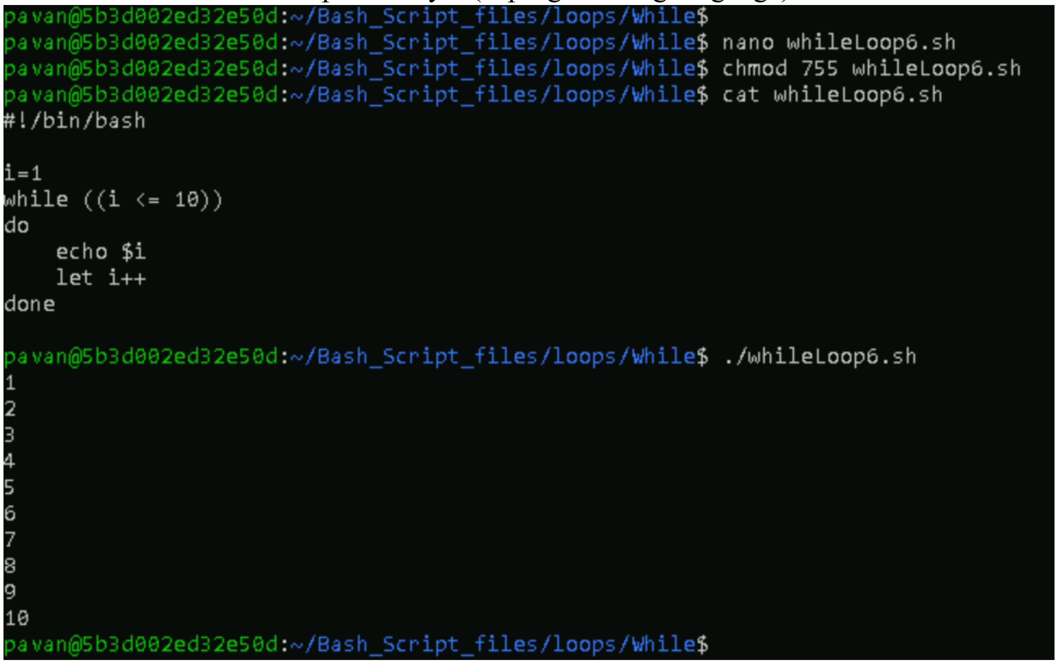
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$ nano whileLoop4.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$ chmod 755 whileLoop4.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$ cat whileLoop4.sh
#!/bin/bash

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

pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$ ./whileLoop4.sh
Countdown for Website Launching...
10
9
8
7
6
5
4
3
Mission Aborted, Some Technical Error Found.
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$ _
```

5. To use Continue Statement in BASH WHILE Loop.



```
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$ nano whileLoop5.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$ chmod 755 whileLoop5.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$ cat whileLoop5.sh
#!/bin/bash

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."

pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$ ./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.
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$
```

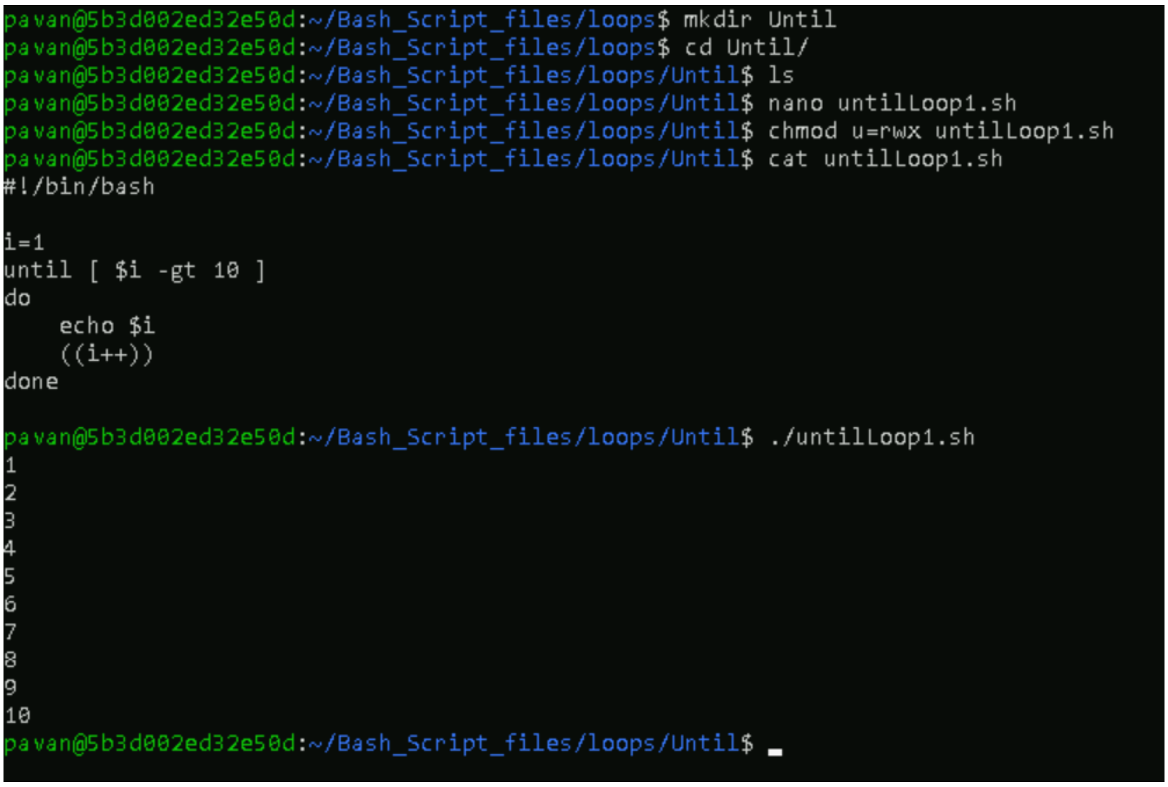
6. To use BASH WHILE Loop in C-Style (C programming language).



```
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$ nano whileLoop6.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$ chmod 755 whileLoop6.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$ cat whileLoop6.sh
#!/bin/bash

i=1
while ((i <= 10))
do
    echo $i
    let i++
done

pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$ ./whileLoop6.sh
1
2
3
4
5
6
7
8
9
10
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/While$
```

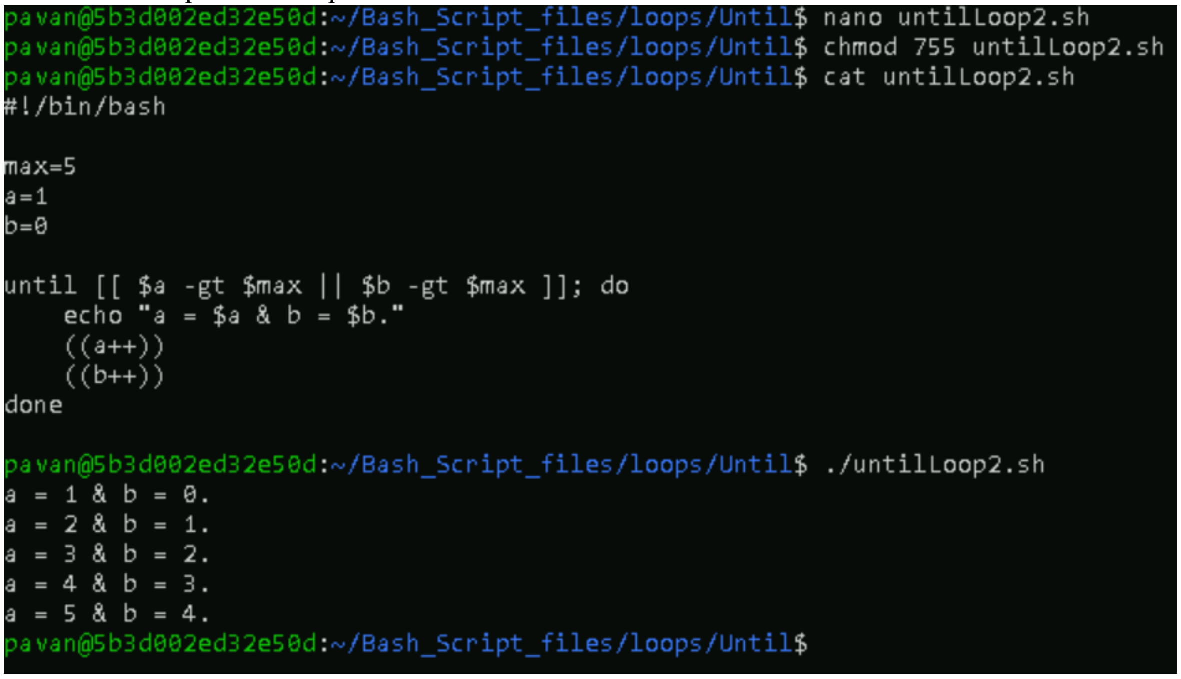
4.BASH UNTIL Loop

1. Bash Until Loop with a single condition.



```
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops$ mkdir Until
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops$ cd Until/
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/Until$ ls
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/Until$ nano untilLoop1.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/Until$ chmod u=rwx untilLoop1.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/Until$ cat untilLoop1.sh
#!/bin/bash

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

pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/Until$ ./untilLoop1.sh
1
2
3
4
5
6
7
8
9
10
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/Until$
```

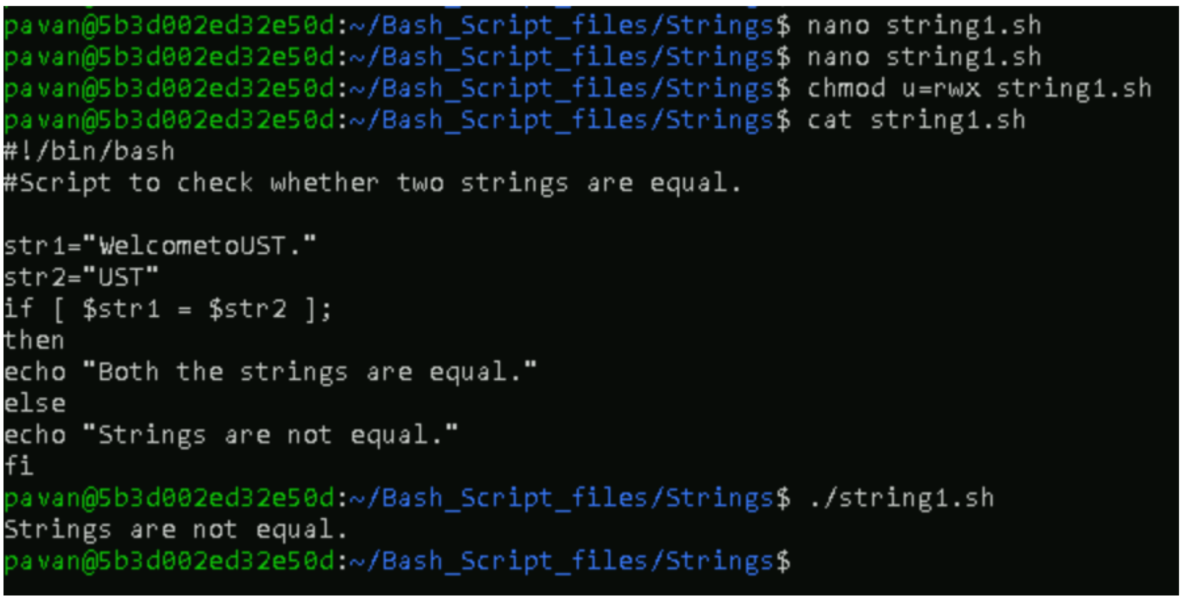
2. Bash Until Loop with multiple conditions.



```
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/Until$ nano untilLoop2.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/Until$ chmod 755 untilLoop2.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/Until$ cat untilLoop2.sh
#!/bin/bash

max=5
a=1
b=0

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

pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/Until$ ./untilLoop2.sh
a = 1 & b = 0.
a = 2 & b = 1.
a = 3 & b = 2.
a = 4 & b = 3.
a = 5 & b = 4.
pavan@5b3d002ed32e50d:~/Bash_Script_files/loops/Until$
```

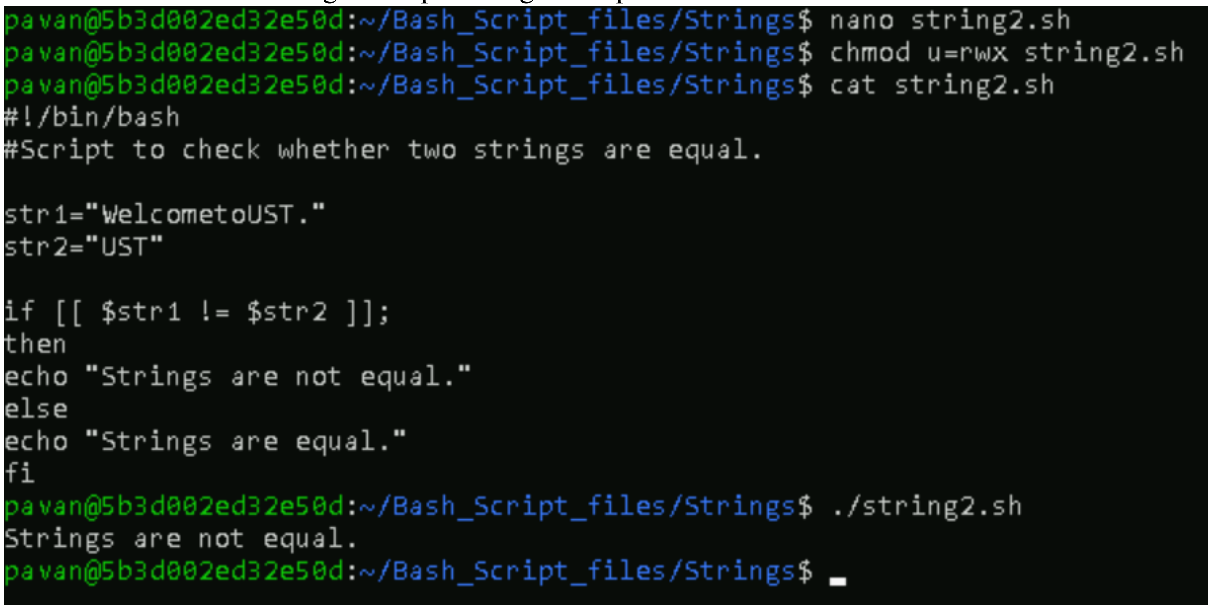
5.BASH Strings

1. To check whether two strings are equal using “=” operator.



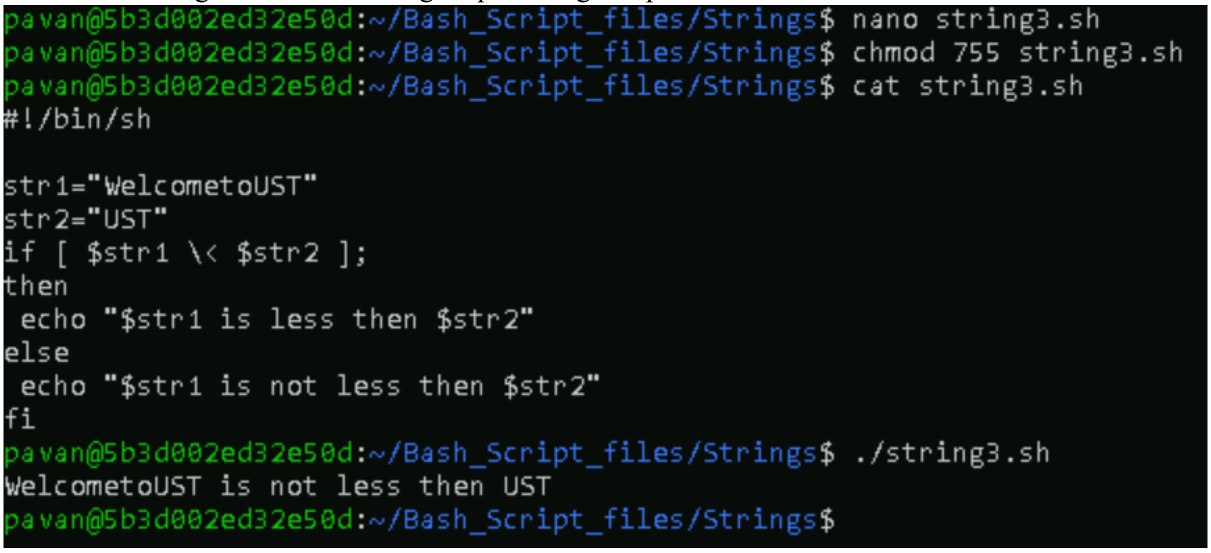
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings$ nano string1.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings$ nano string1.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings$ chmod u=rwx string1.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings$ cat string1.sh
#!/bin/bash
#Script to check whether two strings are equal.

str1="WelcometoUST."
str2="UST"
if [ $str1 = $str2 ];
then
echo "Both the strings are equal."
else
echo "Strings are not equal."
fi
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings$ ./string1.sh
Strings are not equal.
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings$
```

2. To check whether two strings are equal using “!=” operator



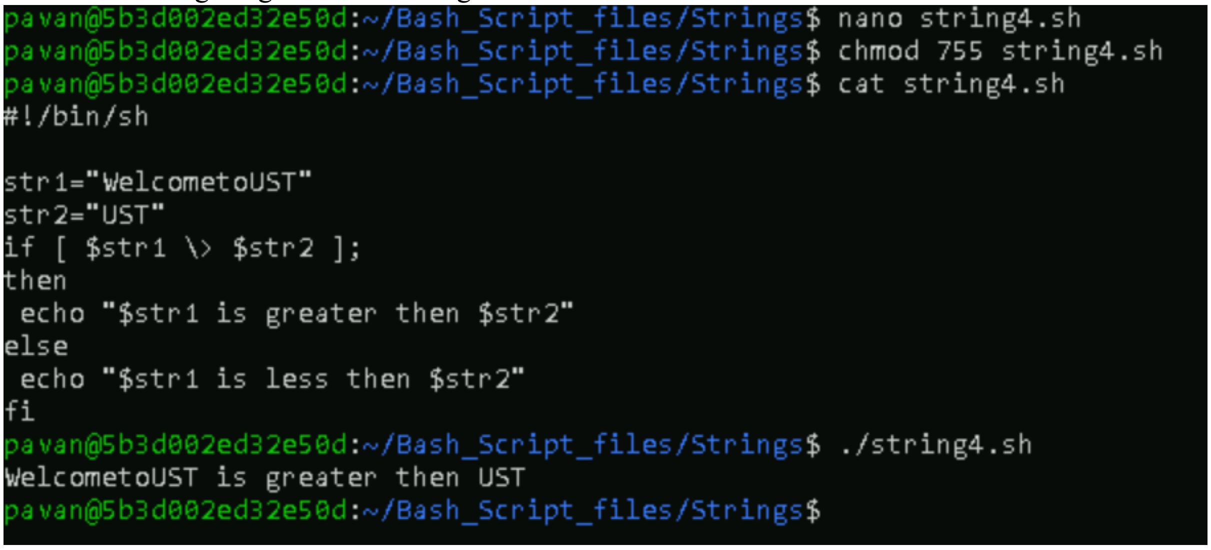
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings$ nano string2.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings$ chmod u=rwx string2.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings$ cat string2.sh
#!/bin/bash
#Script to check whether two strings are equal.

str1="WelcometoUST."
str2="UST"

if [[ $str1 != $str2 ]];
then
echo "Strings are not equal."
else
echo "Strings are equal."
fi
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings$ ./string2.sh
Strings are not equal.
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings$ _
```

3. To check if string1 is less than string2 equal using “” operator.



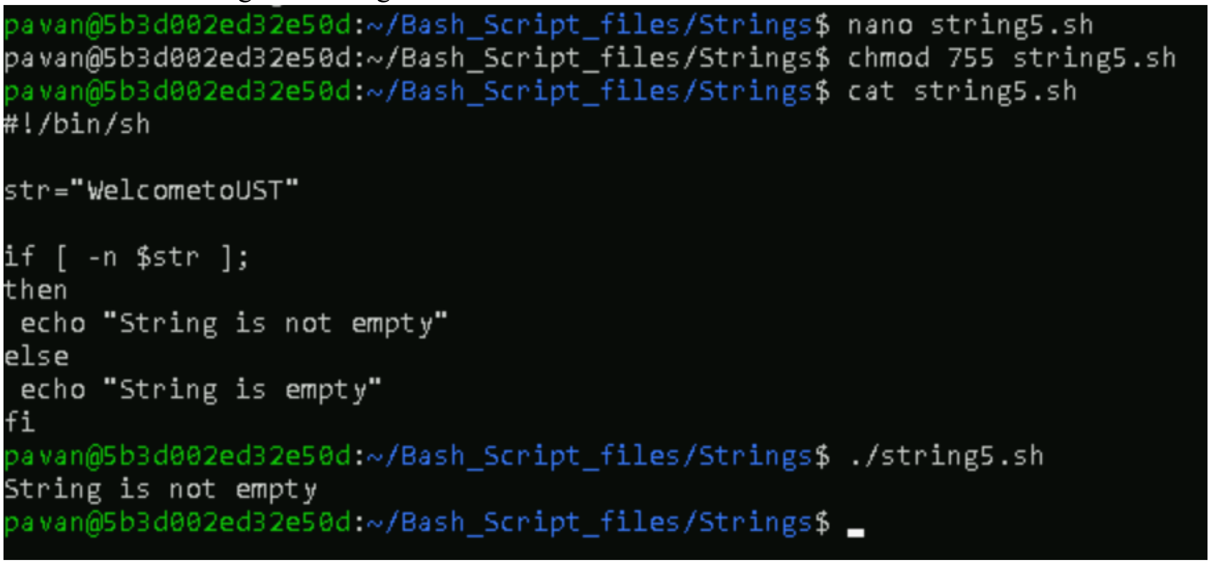
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings$ nano string3.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings$ chmod 755 string3.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings$ cat string3.sh
#!/bin/sh

str1="WelcometoUST"
str2="UST"
if [ $str1 \< $str2 ];
then
 echo "$str1 is less then $str2"
else
 echo "$str1 is not less then $str2"
fi
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings$ ./string3.sh
WelcometoUST is not less then UST
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings$
```

4. To check if string1 is greater than string2.



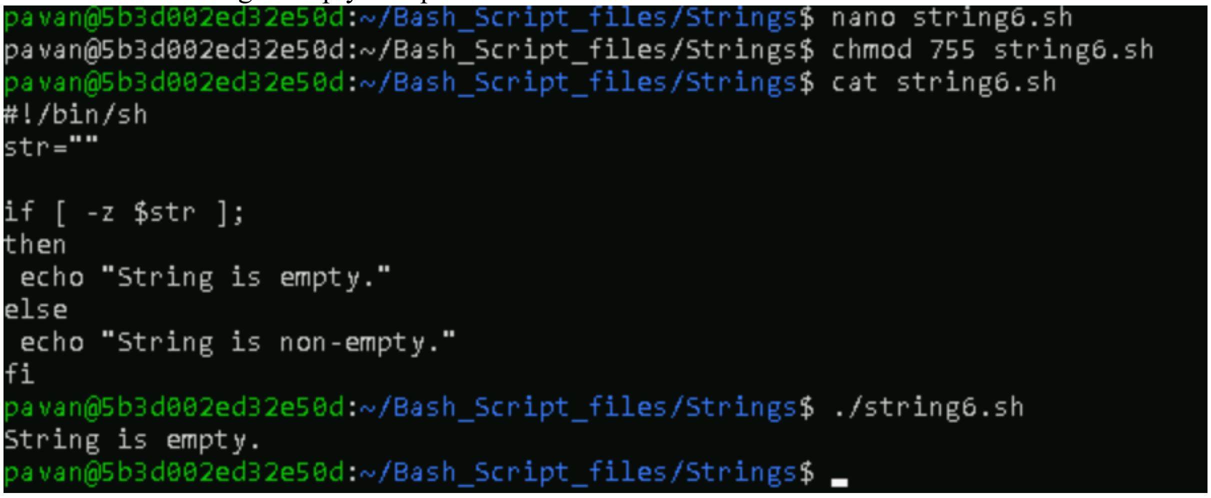
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings$ nano string4.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings$ chmod 755 string4.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings$ cat string4.sh
#!/bin/sh

str1="WelcometoUST"
str2="UST"
if [ $str1 \> $str2 ];
then
 echo "$str1 is greater then $str2"
else
 echo "$str1 is less then $str2"
fi
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings$ ./string4.sh
WelcometoUST is greater then UST
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings$
```

5. To check if the string is zero or greater than zero.



```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings$ nano string5.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings$ chmod 755 string5.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings$ cat string5.sh
#!/bin/sh

str="WelcometoUST"

if [ -n $str ];
then
 echo "String is not empty"
else
 echo "String is empty"
fi
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings$ ./string5.sh
String is not empty
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings$ 
```

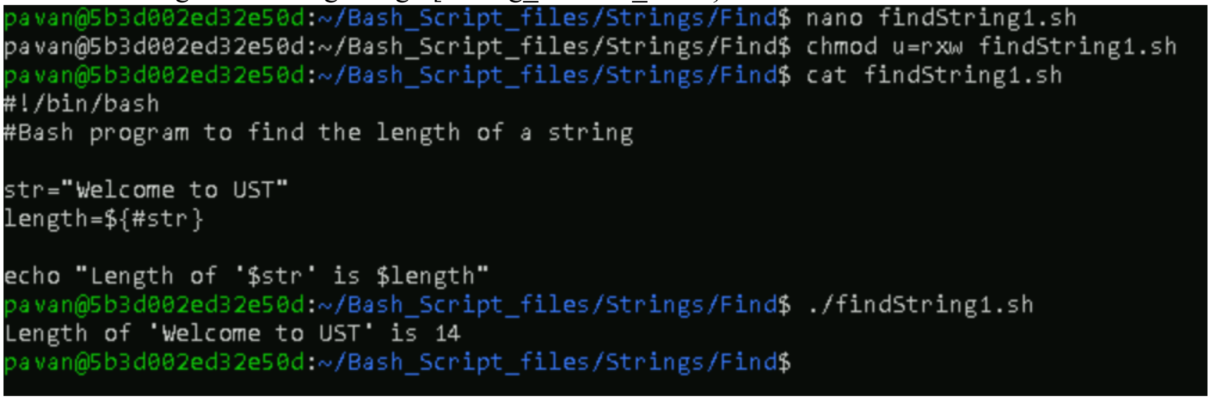
6. To check if the string is empty or equal to zero.



```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings$ nano string6.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings$ chmod 755 string6.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings$ cat string6.sh
#!/bin/sh
str=""

if [ -z $str ];
then
 echo "String is empty."
else
 echo "String is non-empty."
fi
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings$ ./string6.sh
String is empty.
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings$ 
```

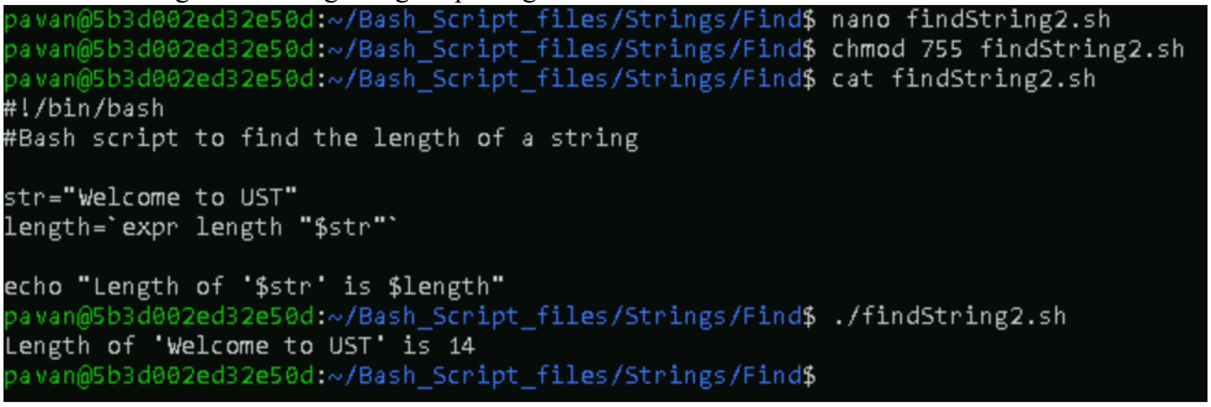
6.BASH FIND String Length

1. To find the length of a string using $[#string_variable_name}.



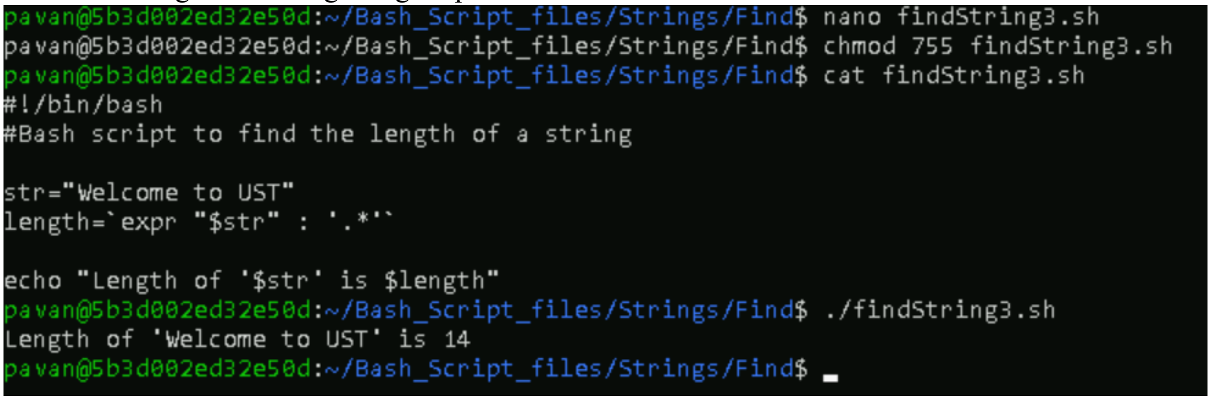
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/Find$ nano findString1.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/Find$ chmod u=rxw findString1.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/Find$ cat findString1.sh
#!/bin/bash
#Bash program to find the length of a string

str="Welcome to UST"
length=${#str}

echo "Length of '$str' is $length"
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/Find$ ./findString1.sh
Length of 'Welcome to UST' is 14
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/Find$
```

2. To find the length of a string using `expr length "$str"`.



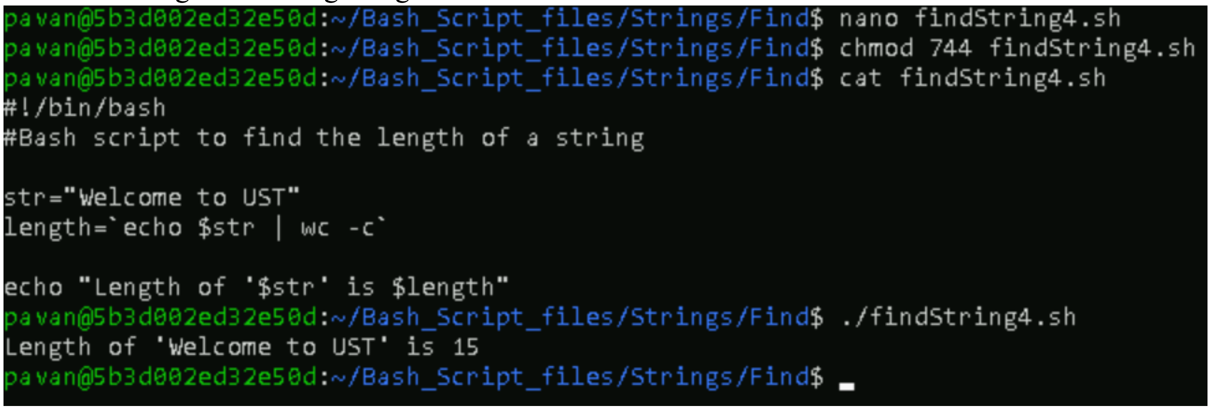
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/Find$ nano findString2.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/Find$ chmod 755 findString2.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/Find$ cat findString2.sh
#!/bin/bash
#Bash script to find the length of a string

str="Welcome to UST"
length=`expr length "$str"`

echo "Length of '$str' is $length"
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/Find$ ./findString2.sh
Length of 'Welcome to UST' is 14
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/Find$
```

3. To find the length of a string using `expr "$str": ' .*'`.



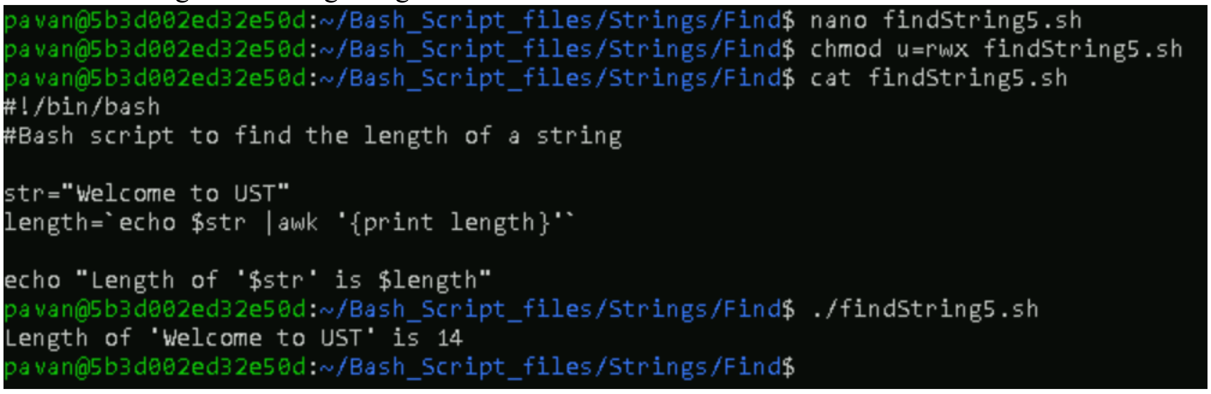
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/Find$ nano findString3.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/Find$ chmod 755 findString3.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/Find$ cat findString3.sh
#!/bin/bash
#Bash script to find the length of a string

str="Welcome to UST"
length=`expr "$str" : '.*'`

echo "Length of '$str' is $length"
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/Find$ ./findString3.sh
Length of 'Welcome to UST' is 14
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/Find$ _
```

4. To find the length of a string using `wc` command.



```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/Find$ nano findString4.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/Find$ chmod 744 findString4.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/Find$ cat findString4.sh
#!/bin/bash
#Bash script to find the length of a string

str="Welcome to UST"
length=`echo $str | wc -c`

echo "Length of '$str' is $length"
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/Find$ ./findString4.sh
Length of 'Welcome to UST' is 15
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/Find$ _
```

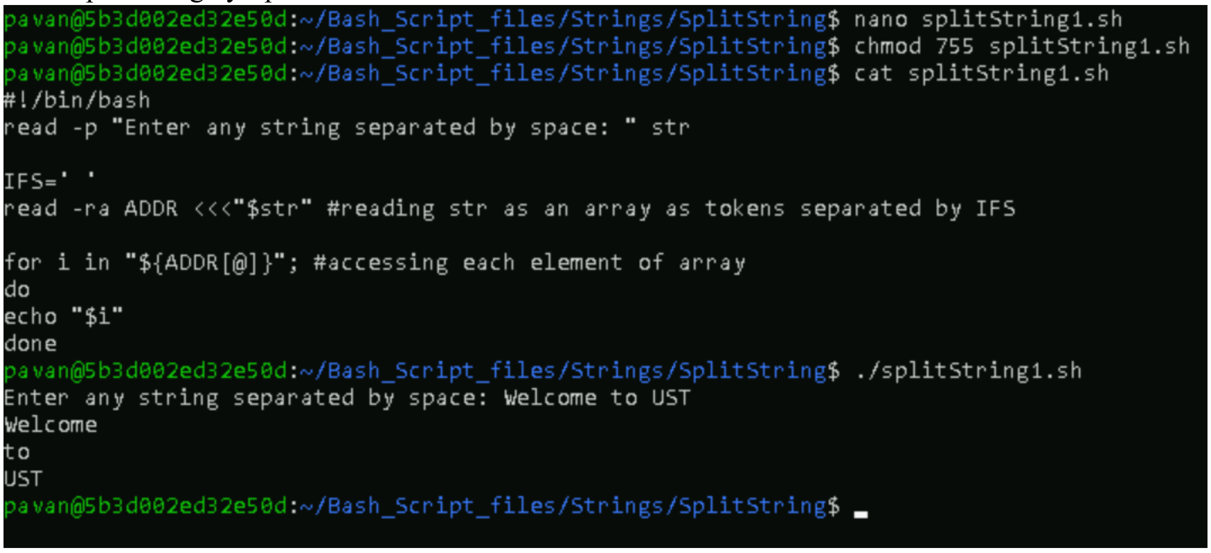
5. To find the length of a string using `awk` command.



```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/Find$ nano findString5.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/Find$ chmod u=rwx findString5.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/Find$ cat findString5.sh
#!/bin/bash
#Bash script to find the length of a string

str="Welcome to UST"
length=`echo $str |awk '{print length}'`

echo "Length of '$str' is $length"
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/Find$ ./findString5.sh
Length of 'Welcome to UST' is 14
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/Find$
```

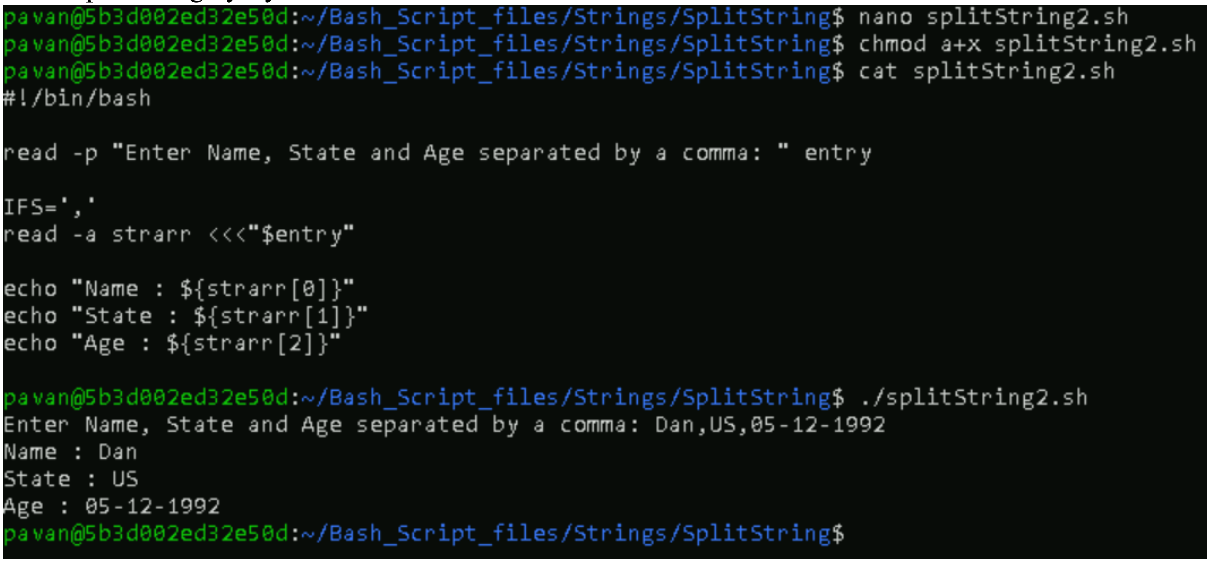
7. Bash Split String

1. Bash Split String by Space.



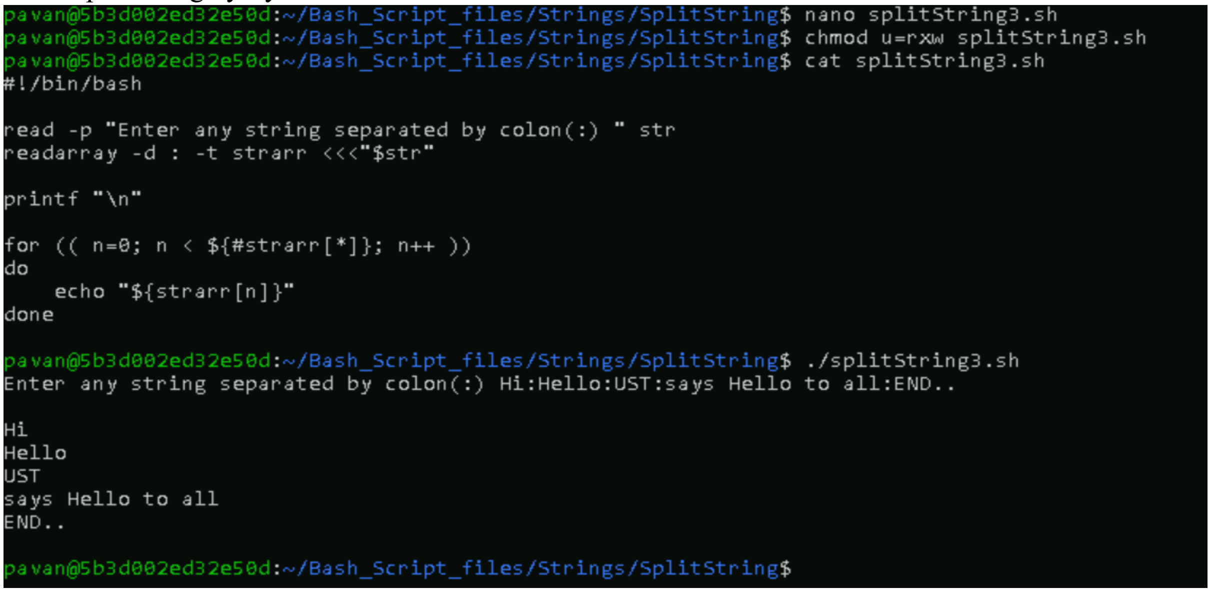
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SplitString$ nano splitString1.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SplitString$ chmod 755 splitString1.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SplitString$ cat splitString1.sh
#!/bin/bash
read -p "Enter any string separated by space: " str

IFS=' '
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
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SplitString$ ./splitString1.sh
Enter any string separated by space: Welcome to UST
Welcome
to
UST
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SplitString$ _
```

2. Bash Split String by Symbol.



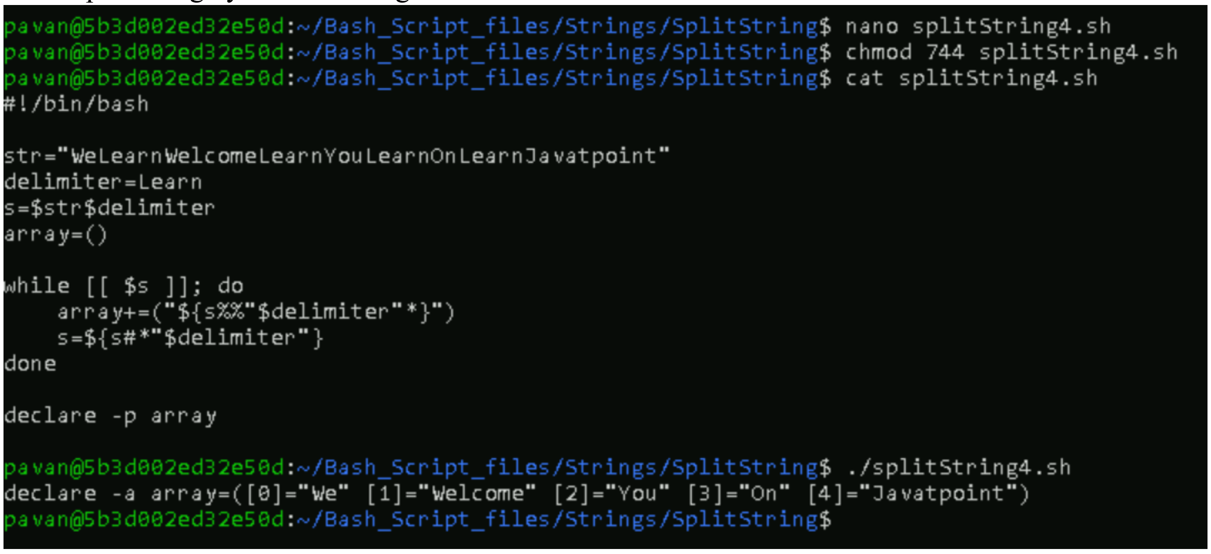
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SplitString$ nano splitString2.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SplitString$ chmod a+x splitString2.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SplitString$ cat splitString2.sh
#!/bin/bash

read -p "Enter Name, State and Age separated by a comma: " entry

IFS=','
read -a strarr <<<"$entry"

echo "Name : ${strarr[0]}"
echo "State : ${strarr[1]}"
echo "Age : ${strarr[2]}"

pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SplitString$ ./splitString2.sh
Enter Name, State and Age separated by a comma: Dan,US,05-12-1992
Name : Dan
State : US
Age : 05-12-1992
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SplitString$
```

3. Bash Split String by Symbol without $IFS variable.



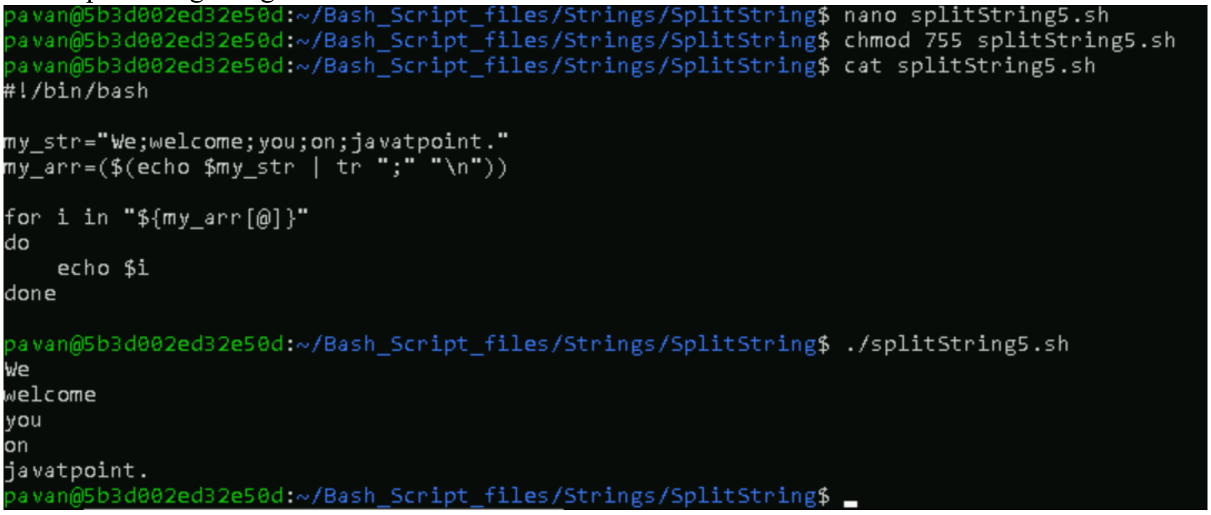
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SplitString$ nano splitString3.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SplitString$ chmod u=rxw splitString3.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SplitString$ cat splitString3.sh
#!/bin/bash

read -p "Enter any string separated by colon(:) " str
readarray -d : -t strarr <<<"$str"

printf "\n"

for (( n=0; n < ${#strarr[*]}; n++ ))
do
    echo "${strarr[n]}"
done

pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SplitString$ ./splitString3.sh
Enter any string separated by colon(:) Hi:Hello:UST:says Hello to all:END..

Hi
Hello
UST
says Hello to all
END..

pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SplitString$
```

4. Bash Split String by another string without $IFS variable.



```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SplitString$ nano splitString4.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SplitString$ chmod 744 splitString4.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SplitString$ cat splitString4.sh
#!/bin/bash

str="WeLearnWelcomeLearnYouLearnOnLearnJavatpoint"
delimiter=Learn
s=$str$delimiter
array=()

while [[ $s ]]; do
    array+=("${s%%"$delimiter"*}")
    s=${s#*"$delimiter"}
done

declare -p array

pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SplitString$ ./splitString4.sh
declare -a array=([0]="We" [1]="Welcome" [2]="You" [3]="On" [4]="Javatpoint")
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SplitString$
```

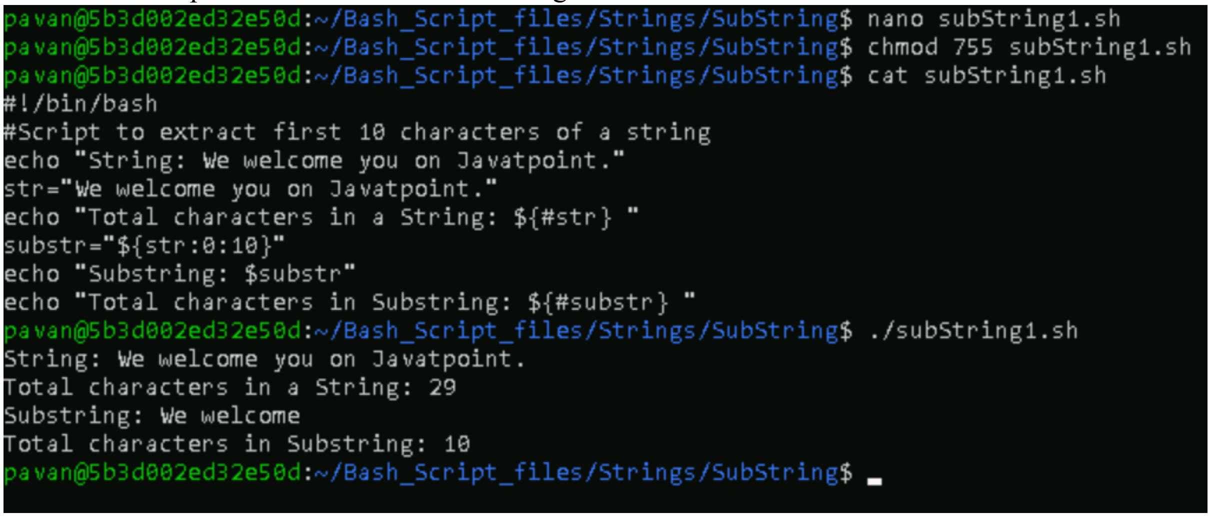
5. Bash Split String using Trim Command.



```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SplitString$ nano splitString5.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SplitString$ chmod 755 splitString5.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SplitString$ cat splitString5.sh
#!/bin/bash

my_str="We;welcome;you;on;javatpoint."
my_arr=($(echo $my_str | tr ";" "\n"))

for i in "${my_arr[@]}"
do
    echo $i
done

pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SplitString$ ./splitString5.sh
We
welcome
you
on
javatpoint.
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SplitString$
```

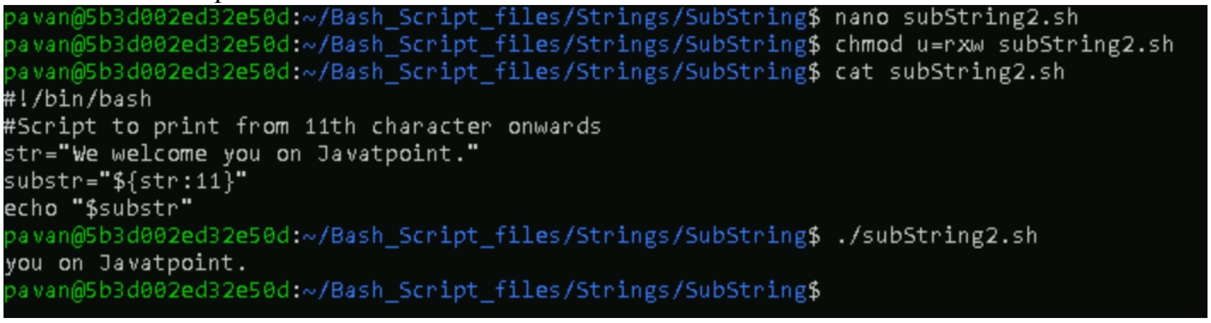
8. Bash Substring

1. To Extract till Specific Characters from Starting.



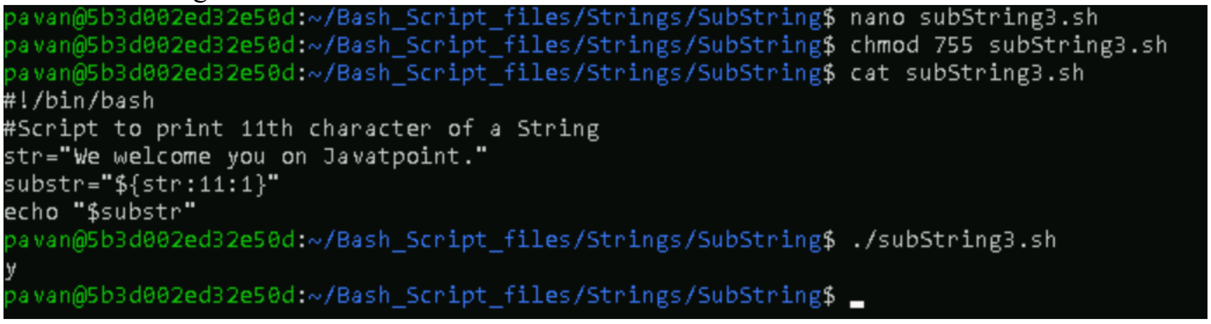
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SubString$ nano subString1.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SubString$ chmod 755 subString1.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SubString$ cat subString1.sh
#!/bin/bash
#Script to extract first 10 characters of 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: ${#substr} "
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SubString$ ./subString1.sh
String: We welcome you on Javatpoint.
Total characters in a String: 29
Substring: We welcome
Total characters in Substring: 10
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SubString$
```

2. To Extract from Specific Character onwards.



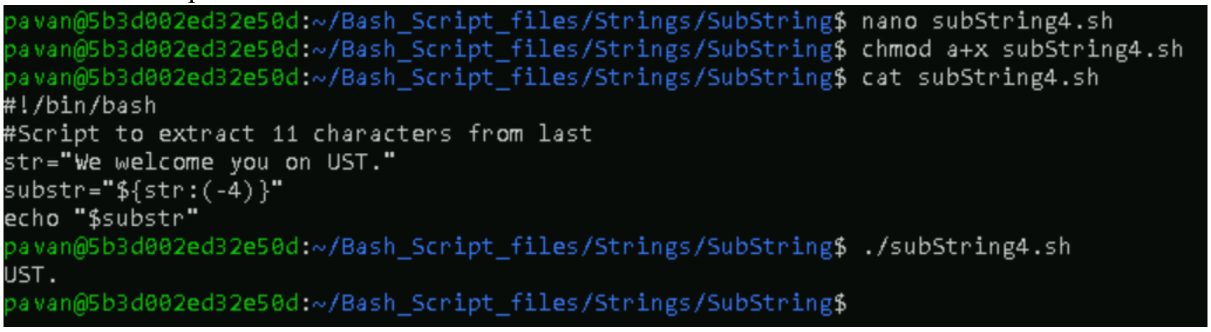
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SubString$ nano subString2.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SubString$ chmod u=rxw subString2.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SubString$ cat subString2.sh
#!/bin/bash
#Script to print from 11th character onwards
str="We welcome you on Javatpoint."
substr="${str:11}"
echo "$substr"
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SubString$ ./subString2.sh
you on Javatpoint.
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SubString$
```

3. To Extract a Single Character.



```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SubString$ nano subString3.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SubString$ chmod 755 subString3.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SubString$ cat subString3.sh
#!/bin/bash
#Script to print 11th character of a String
str="We welcome you on Javatpoint."
substr="${str:11:1}"
echo "$substr"
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SubString$ ./subString3.sh
y
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SubString$
```

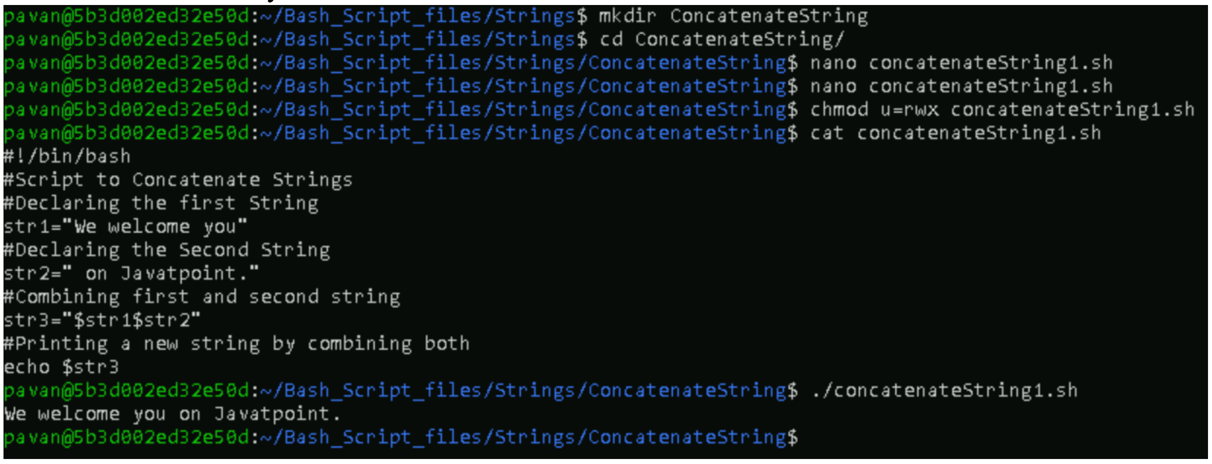
4. To Extract the specific characters from last.



```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SubString$ nano subString4.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SubString$ chmod a+x subString4.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SubString$ cat subString4.sh
#!/bin/bash
#Script to extract 11 characters from last
str="We welcome you on UST."
substr="${str:(-4)}"
echo "$substr"
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SubString$ ./subString4.sh
UST.
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/SubString$
```

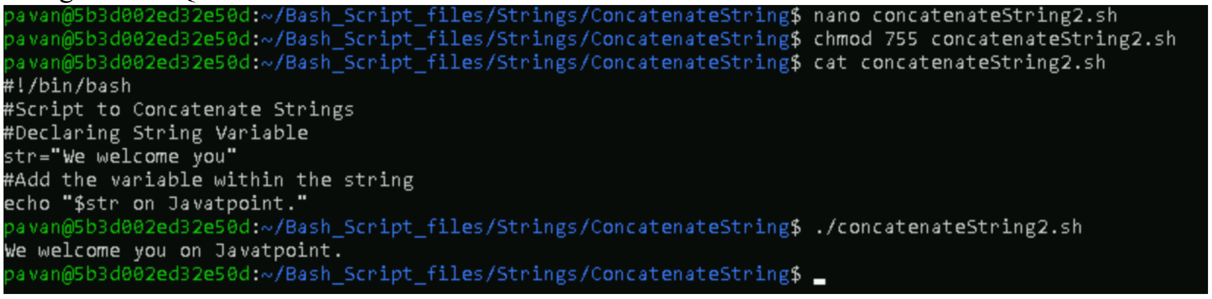
9. Bash Concatenate String

1. Write Variables Side by Side.



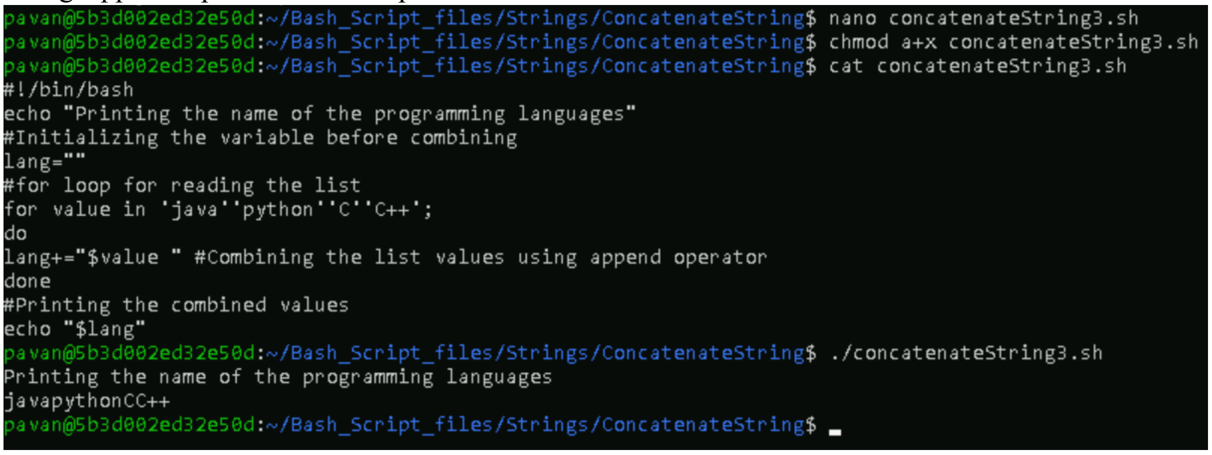
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings$ mkdir ConcatenateString
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings$ cd ConcatenateString/
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/ConcatenateString$ nano concatenateString1.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/ConcatenateString$ nano concatenateString1.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/ConcatenateString$ chmod u=rwx concatenateString1.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/ConcatenateString$ cat concatenateString1.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 $str3
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/ConcatenateString$ ./concatenateString1.sh
We welcome you on Javatpoint.
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/ConcatenateString$
```

2. Using Double Quotes.



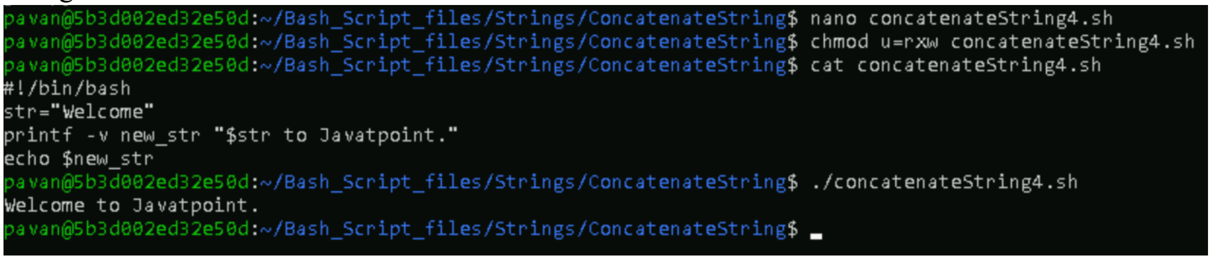
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/ConcatenateString$ nano concatenateString2.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/ConcatenateString$ chmod 755 concatenateString2.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/ConcatenateString$ cat concatenateString2.sh
#!/bin/bash
#Script to Concatenate Strings
#Declaring String Variable
str="We welcome you"
#Add the variable within the string
echo "$str on Javatpoint."
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/ConcatenateString$ ./concatenateString2.sh
We welcome you on Javatpoint.
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/ConcatenateString$
```

3. Using Append Operator with Loop.



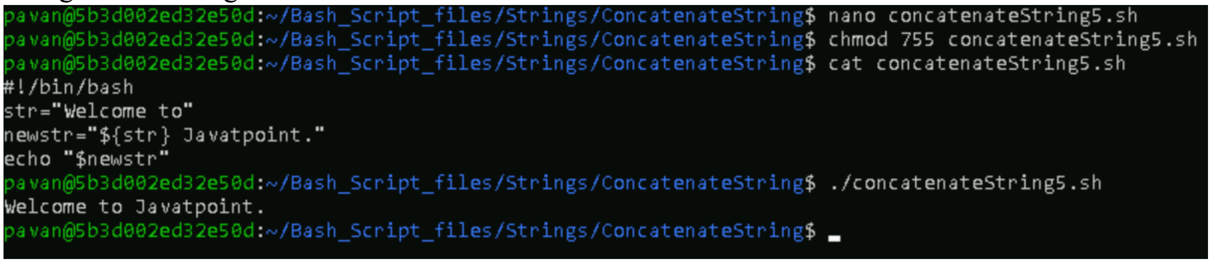
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/ConcatenateString$ nano concatenateString3.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/ConcatenateString$ chmod a+x concatenateString3.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/ConcatenateString$ cat concatenateString3.sh
#!/bin/bash
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"
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/ConcatenateString$ ./concatenateString3.sh
Printing the name of the programming languages
javapythonCC++
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/ConcatenateString$
```

4. Using the Printf Function.



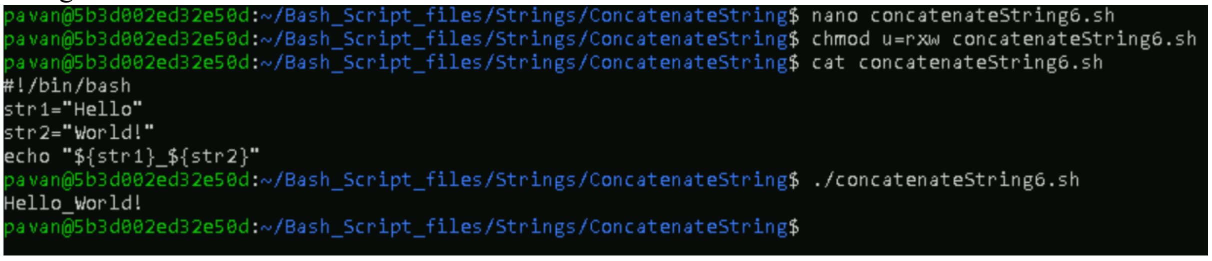
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/ConcatenateString$ nano concatenateString4.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/ConcatenateString$ chmod u=rxw concatenateString4.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/ConcatenateString$ cat concatenateString4.sh
#!/bin/bash
str="Welcome"
printf -v new_str "$str to Javatpoint."
echo $new_str
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/ConcatenateString$ ./concatenateString4.sh
Welcome to Javatpoint.
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/ConcatenateString$
```

5. Using Literal Strings.



```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/ConcatenateString$ nano concatenateString5.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/ConcatenateString$ chmod 755 concatenateString5.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/ConcatenateString$ cat concatenateString5.sh
#!/bin/bash
str="Welcome to"
newstr="${str} Javatpoint."
echo "$newstr"
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/ConcatenateString$ ./concatenateString5.sh
Welcome to Javatpoint.
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/ConcatenateString$
```

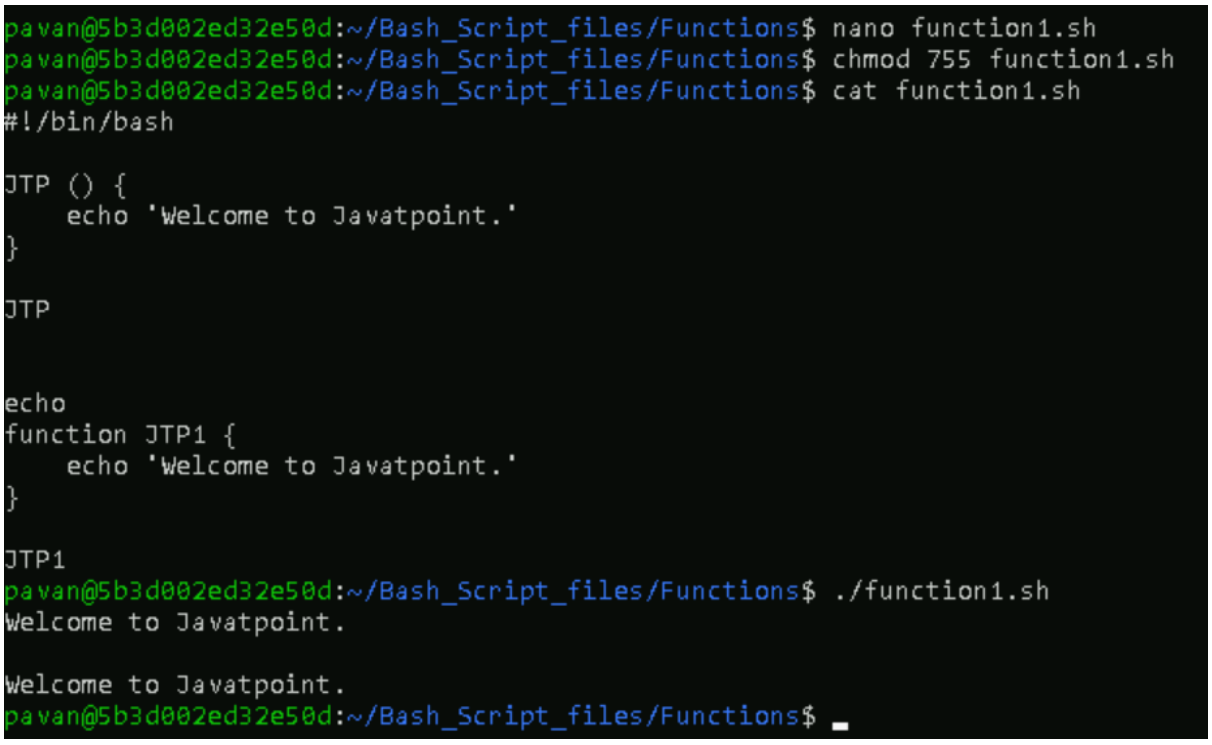
6. Using Underscore.



```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/ConcatenateString$ nano concatenateString6.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/ConcatenateString$ chmod u=rxw concatenateString6.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/ConcatenateString$ cat concatenateString6.sh
#!/bin/bash
str1="Hello"
str2="World!"
echo "${str1}_${str2}"
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/ConcatenateString$ ./concatenateString6.sh
Hello_World!
pavan@5b3d002ed32e50d:~/Bash_Script_files/Strings/ConcatenateString$
```

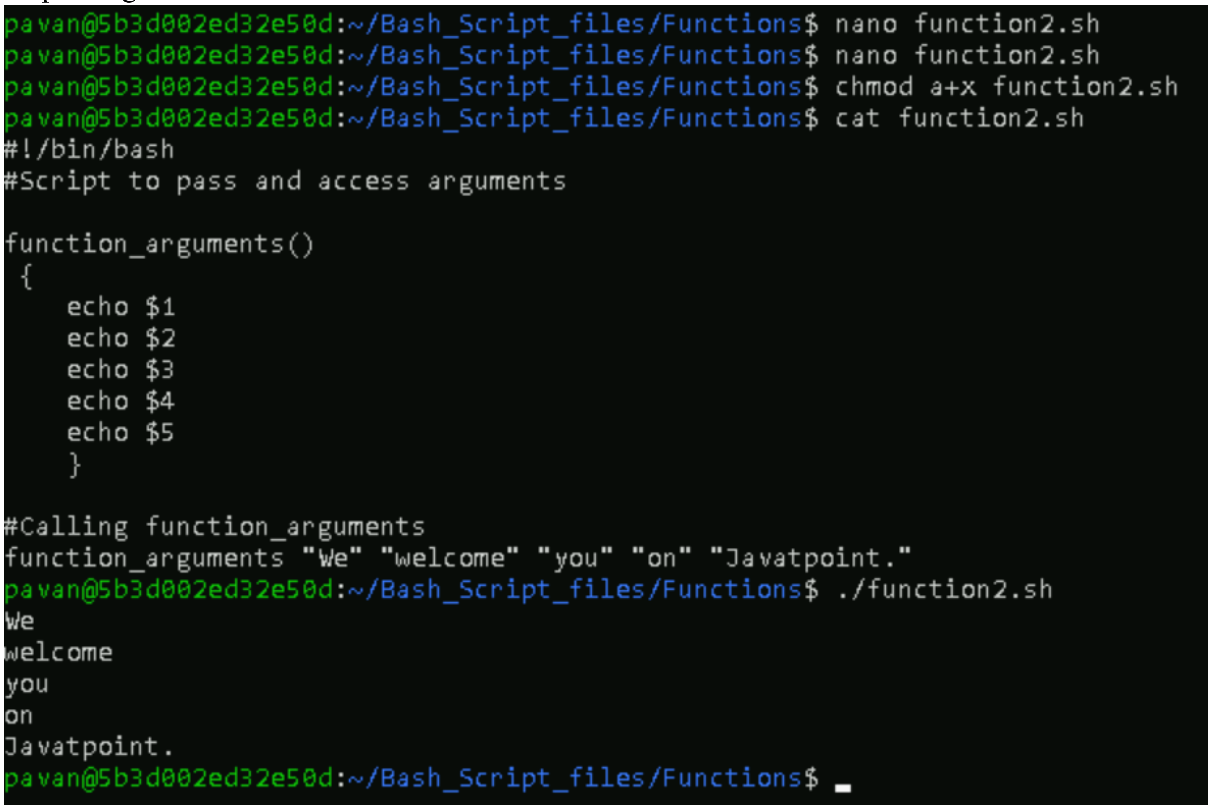
10. Bash Functions

1. To declare functions.



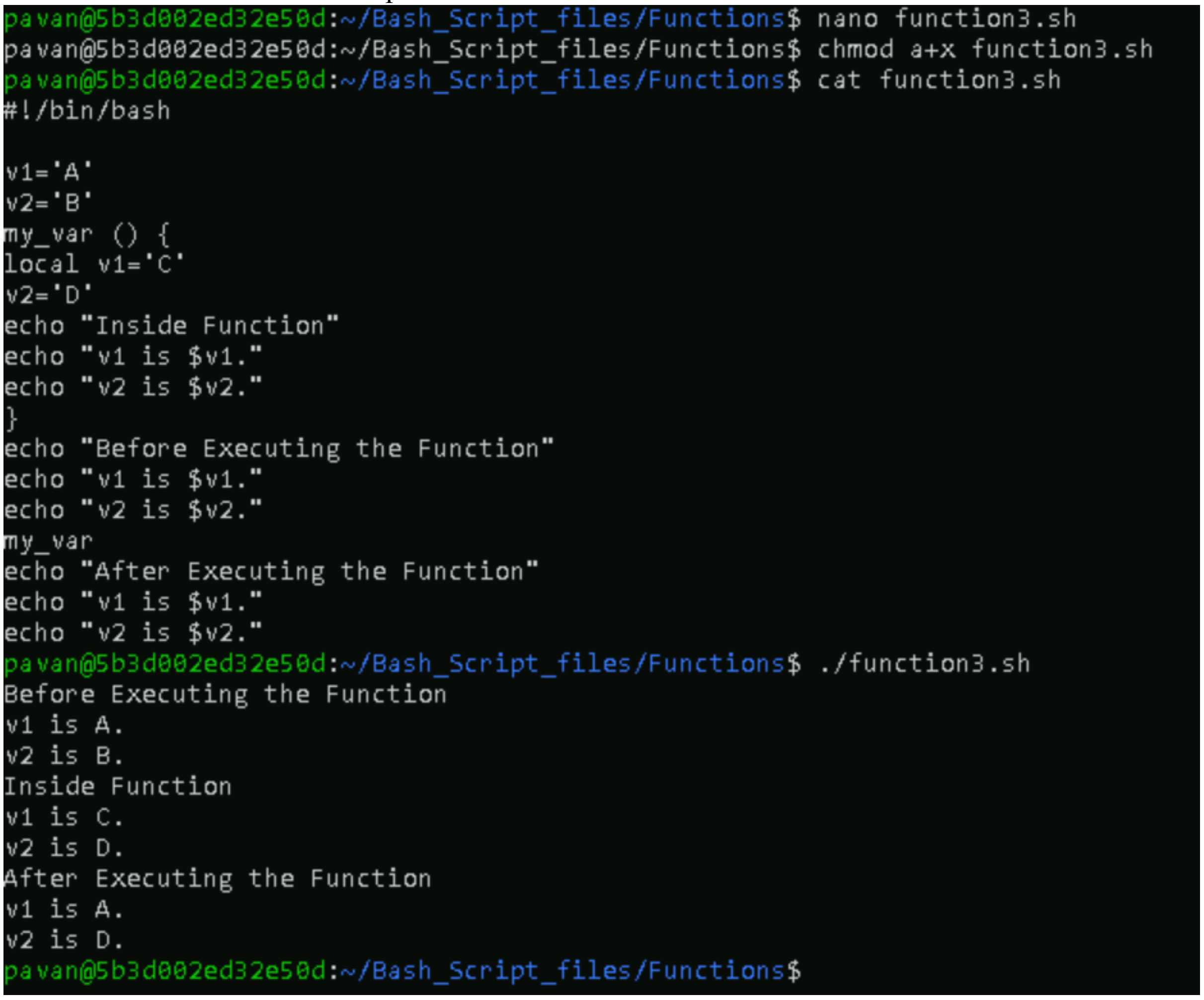
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Functions$ nano function1.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Functions$ chmod 755 function1.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Functions$ cat function1.sh
#!/bin/bash

JTP () {
    echo 'Welcome to Javatpoint.'
}

JTP


echo
function JTP1 {
    echo 'Welcome to Javatpoint.'
}

JTP1
pavan@5b3d002ed32e50d:~/Bash_Script_files/Functions$ ./function1.sh
Welcome to Javatpoint.

Welcome to Javatpoint.
pavan@5b3d002ed32e50d:~/Bash_Script_files/Functions$ _
```

2. To pass arguments to functions.



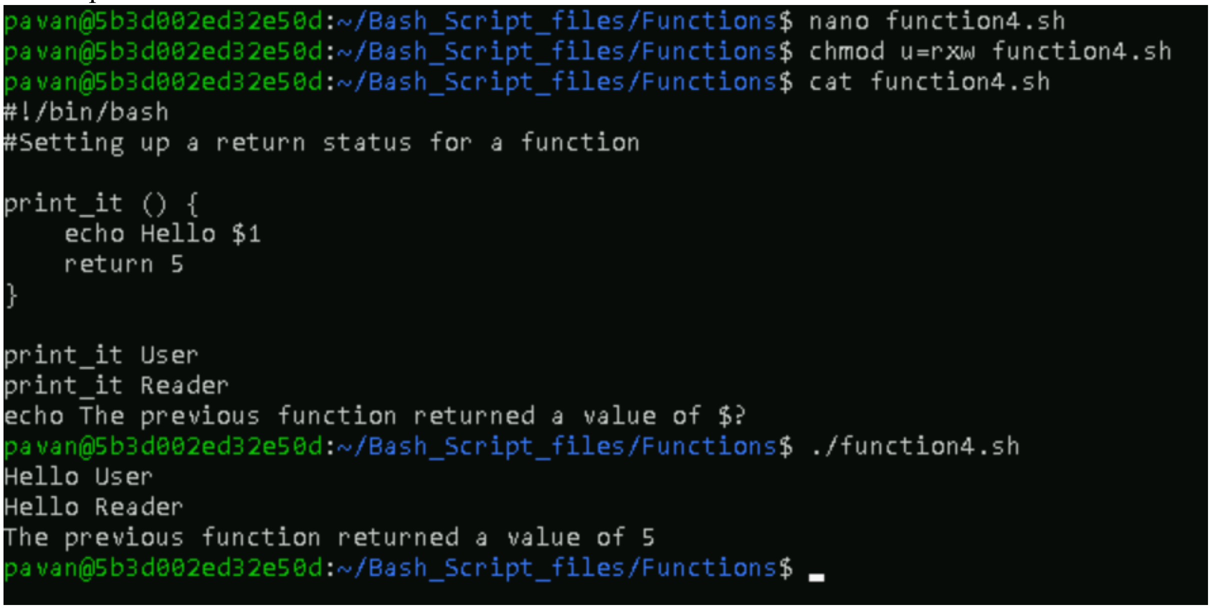
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Functions$ nano function2.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Functions$ nano function2.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Functions$ chmod a+x function2.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Functions$ cat function2.sh
#!/bin/bash
#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."
pavan@5b3d002ed32e50d:~/Bash_Script_files/Functions$ ./function2.sh
We
welcome
you
on
Javatpoint.
pavan@5b3d002ed32e50d:~/Bash_Script_files/Functions$ _
```

3. To understand how variables scope works.



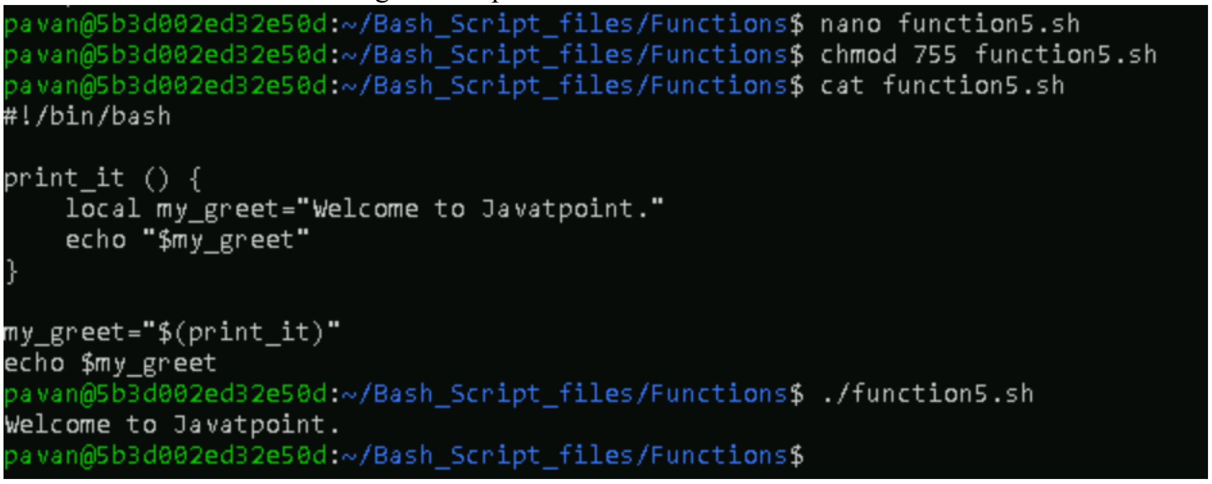
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Functions$ nano function3.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Functions$ chmod a+x function3.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Functions$ cat function3.sh
#!/bin/bash

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 $v2."
pavan@5b3d002ed32e50d:~/Bash_Script_files/Functions$ ./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.
pavan@5b3d002ed32e50d:~/Bash_Script_files/Functions$
```

4. To setup a return status for a function.



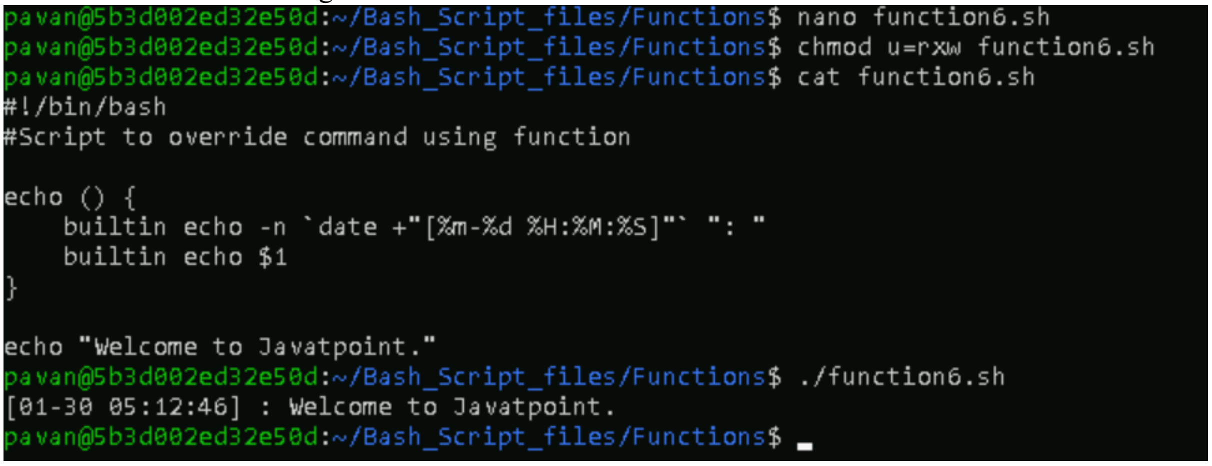
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Functions$ nano function4.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Functions$ chmod u=rxw function4.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Functions$ cat function4.sh
#!/bin/bash
#Setting up a return status for a function

print_it () {
    echo Hello $1
    return 5
}

print_it User
print_it Reader
echo The previous function returned a value of $?
pavan@5b3d002ed32e50d:~/Bash_Script_files/Functions$ ./function4.sh
Hello User
Hello Reader
The previous function returned a value of 5
pavan@5b3d002ed32e50d:~/Bash_Script_files/Functions$ _
```

5. To send the value to stdout using echo or printf commands.



```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Functions$ nano function5.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Functions$ chmod 755 function5.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Functions$ cat function5.sh
#!/bin/bash

print_it () {
    local my_greet="Welcome to Javatpoint."
    echo "$my_greet"
}

my_greet="$(print_it)"
echo $my_greet
pavan@5b3d002ed32e50d:~/Bash_Script_files/Functions$ ./function5.sh
Welcome to Javatpoint.
pavan@5b3d002ed32e50d:~/Bash_Script_files/Functions$
```

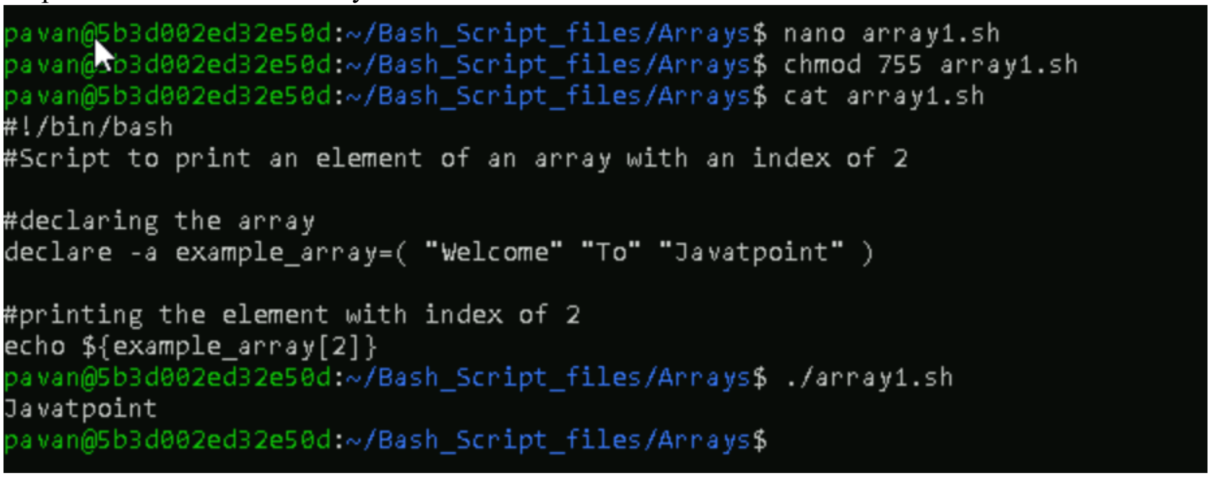
6. To override command using function.



```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Functions$ nano function6.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Functions$ chmod u=rxw function6.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Functions$ cat 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."
pavan@5b3d002ed32e50d:~/Bash_Script_files/Functions$ ./function6.sh
[01-30 05:12:46] : Welcome to Javatpoint.
pavan@5b3d002ed32e50d:~/Bash_Script_files/Functions$ _
```

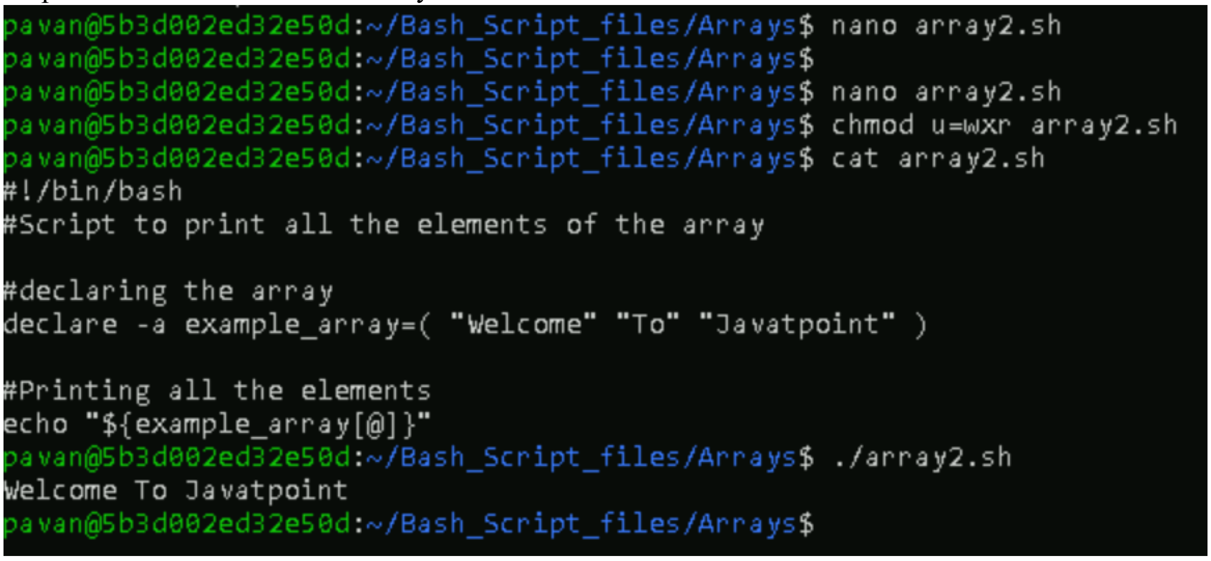
11. Bash Array

1. To print an element of an array with an index of 2.



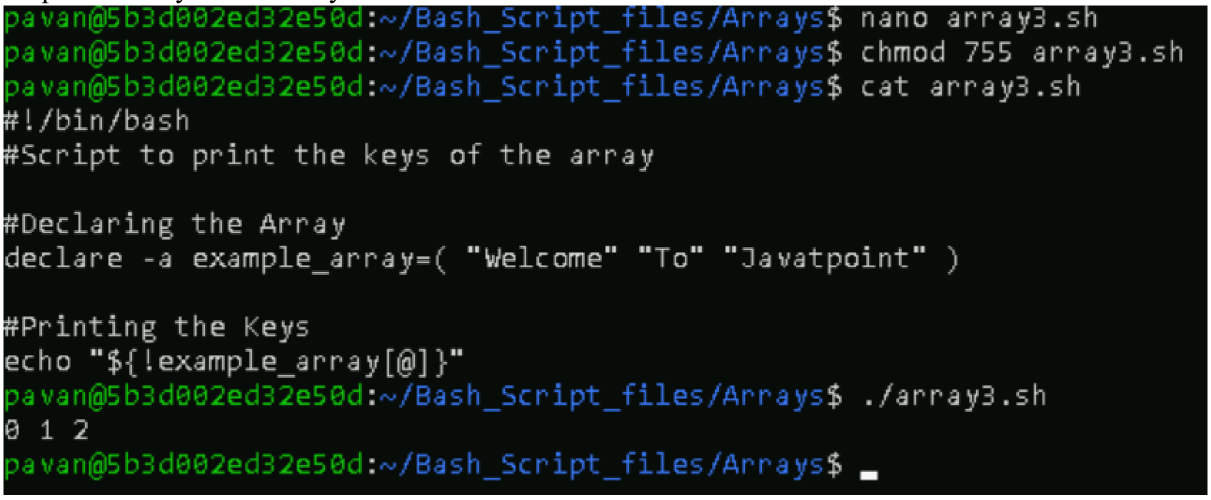
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ nano array1.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ chmod 755 array1.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ cat array1.sh
#!/bin/bash
#Script to print an element of an array with an index of 2

#declaring the array
declare -a example_array=( "Welcome" "To" "Javatpoint" )

#printing the element with index of 2
echo ${example_array[2]}
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ ./array1.sh
Javatpoint
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$
```

2. To print all the elements of the array.



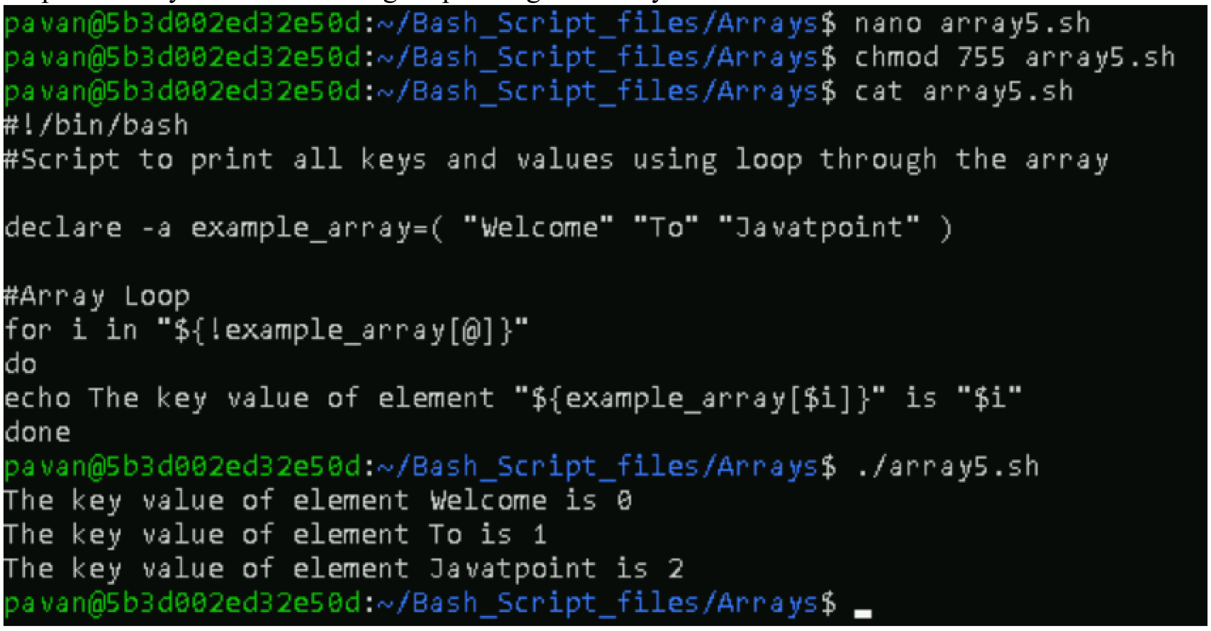
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ nano array2.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ nano array2.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ chmod u=wxr array2.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ cat array2.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 "${example_array[@]}"
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ ./array2.sh
Welcome To Javatpoint
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$
```

3. To print the keys of the array.



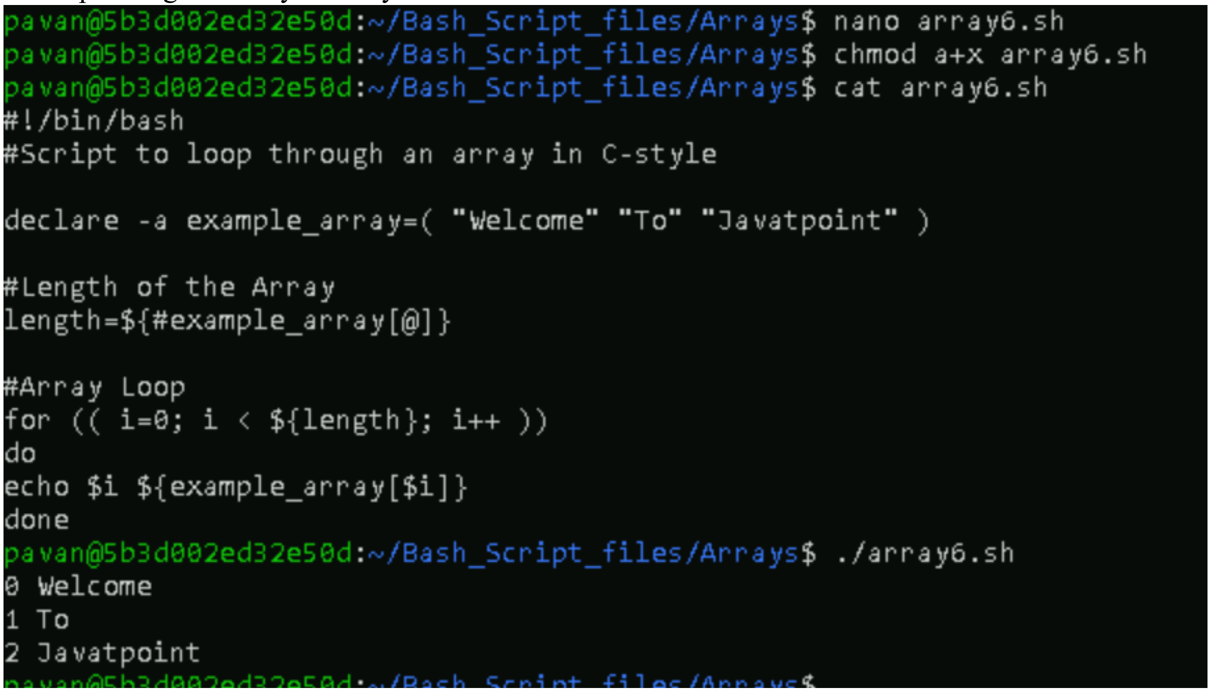
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ nano array3.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ chmod 755 array3.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ cat array3.sh
#!/bin/bash
#Script to print the keys of the array

#Declaring the Array
declare -a example_array=( "Welcome" "To" "Javatpoint" )

#Printing the Keys
echo "${!example_array[@]}"
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ ./array3.sh
0 1 2
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ _
```

4. To print all keys and values using loop through the array.



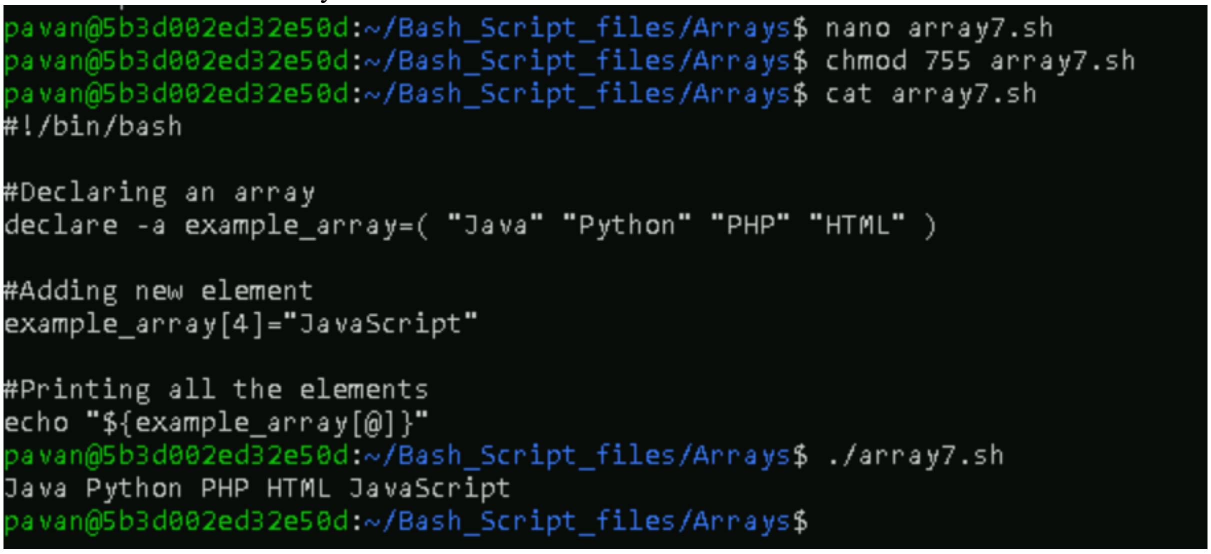
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ nano array5.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ chmod 755 array5.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ cat array5.sh
#!/bin/bash
#Script to print all keys and values using loop through the array

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
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ ./array5.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
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$
```

5. To loop through an array in C-style.



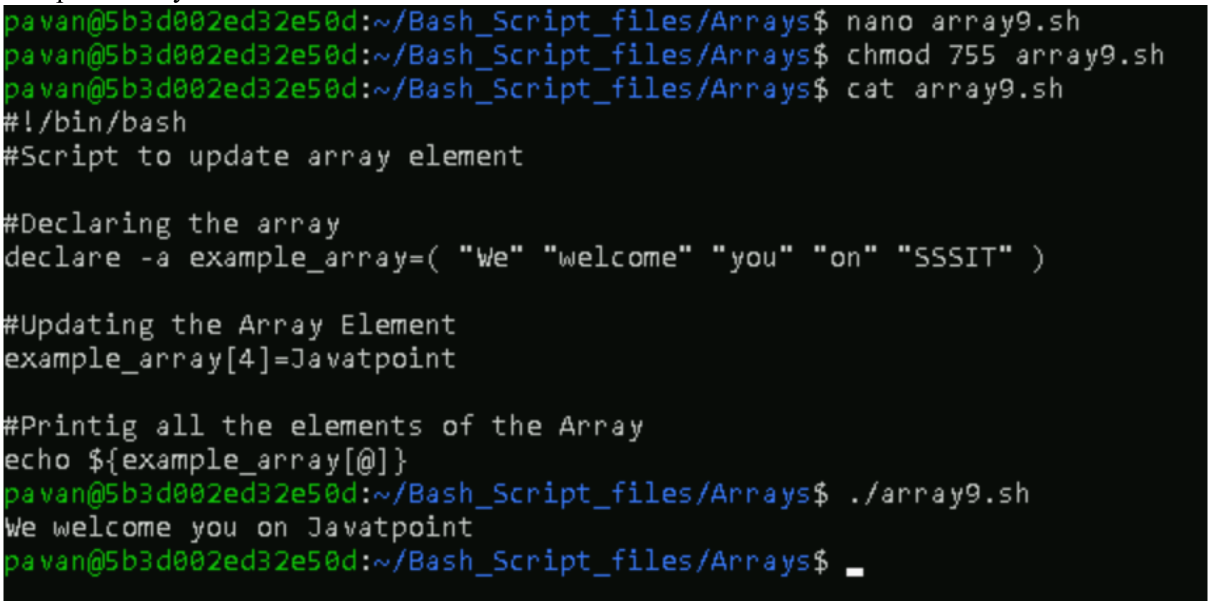
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ nano array6.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ chmod a+x array6.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ cat array6.sh
#!/bin/bash
#Script to loop through an array in C-style

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
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ ./array6.sh
0 Welcome
1 To
2 Javatpoint
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$
```

6. To add elements to an Array.



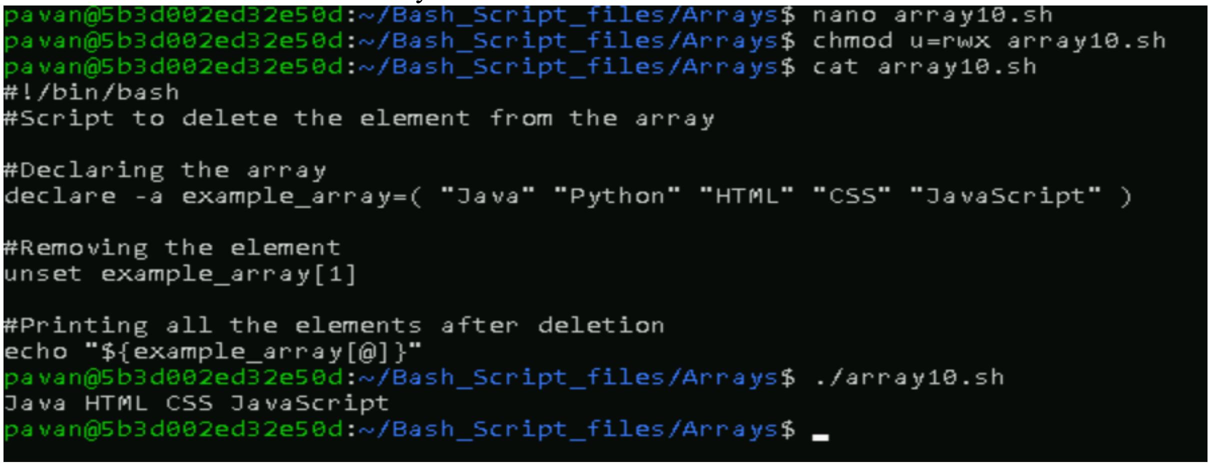
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ nano array7.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ chmod 755 array7.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ cat array7.sh
#!/bin/bash

#Declaring an array
declare -a example_array=( "Java" "Python" "PHP" "HTML" )

#Adding new element
example_array[4]="JavaScript"

#Printing all the elements
echo "${example_array[@]}"
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ ./array7.sh
Java Python PHP HTML JavaScript
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$
```

7. To update array element.



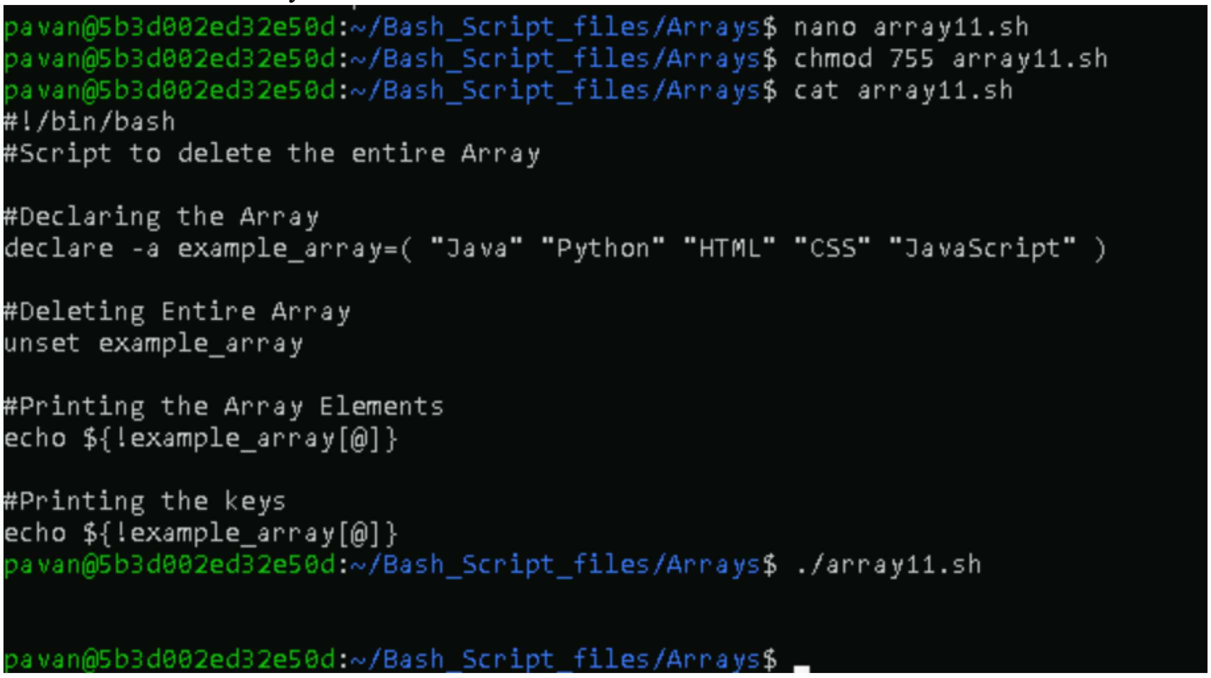
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ nano array9.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ chmod 755 array9.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ cat array9.sh
#!/bin/bash
#Script to update array element

#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 ${example_array[@]}
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ ./array9.sh
We welcome you on Javatpoint
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$
```

8. To delete the element from the array.



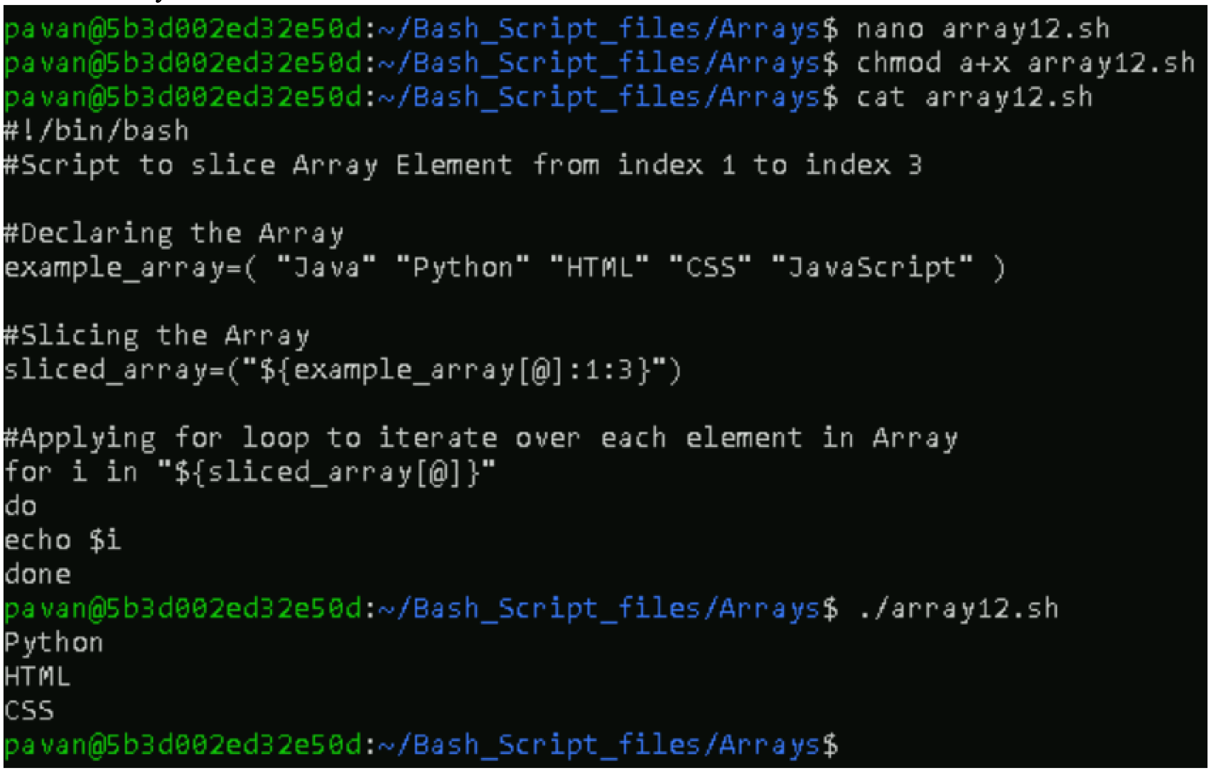
```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ nano array10.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ chmod u=rwx array10.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ cat array10.sh
#!/bin/bash
#Script to delete the element from the array

#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 "${example_array[@]}"
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ ./array10.sh
Java HTML CSS JavaScript
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$
```

9. To delete the entire Array.



```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ nano array11.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ chmod 755 array11.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ cat array11.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 ${!example_array[@]}
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ ./array11.sh


pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ _
```

10. To slice Array Element from index 1 to index 3.



```
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ nano array12.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ chmod a+x array12.sh
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ cat array12.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 $i
done
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$ ./array12.sh
Python
HTML
CSS
pavan@5b3d002ed32e50d:~/Bash_Script_files/Arrays$
```