BASH CASE

1)

```
Select prab@8f7b90438d62516: ~/case
  GNU nano 7.2                    script1.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
```

```
prab@8f7b90438d62516: ~/case
prab@8f7b90438d62516:~/case$ touch script1.sh
prab@8f7b90438d62516:~/case$ chmod +x script1.sh
prab@8f7b90438d62516:~/case$ nano script1.sh
prab@8f7b90438d62516:~/case$ ./script1.sh
Do you know Java Programming?
Yes/No? :yes
That's amazing.
```

2)

```
prab@8f7b90438d62516: ~/case
  GNU nano 7.2                    script2.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
```

```
prab@8f7b90438d62516: ~/case
prab@8f7b90438d62516:~/case$ touch script2.sh
prab@8f7b90438d62516:~/case$ chmod +x script2.sh
prab@8f7b90438d62516:~/case$ nano script2.sh
```

```
prab@8f7b90438d62516:~/case$ ./script2.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.

prab@8f7b90438d62516:~/case$ ./script2.sh
Which Operating System are you using?
Windows, Android, Chrome, Linux, Others?
Type your OS Name:linux
You might be serious about security!!
```

BASH FOR LOOP

1) Basic For Loop

```
prab@8f7b90438d62516: ~
  GNU nano 7.2                    basic_example.sh *
#!/bin/bash
learn="start learning from javatpoint"
for learn in $learn
do
echo $learn
done
echo "Thank You"
```

```
prab@8f7b90438d62516: ~
prab@8f7b90438d62516:~$ touch basic_example.sh
prab@8f7b90438d62516:~$ chmod +x basic_example.sh
prab@8f7b90438d62516:~$ nano basic_example.sh
prab@8f7b90438d62516:~$ ./basic_example.sh
start
learning
from
javatpoint
Thank You
prab@8f7b90438d62516:~$
```

2) For loop to read a range

```
prab@8f7b90438d62516: ~
  GNU nano 7.2                    range_example.sh *
#!/bin/bash
for num in {1..10}
do
echo $num
done
echo "series of numbers from 1 to 10"
```

```
prab@8f7b90438d62516: ~
prab@8f7b90438d62516:~$ touch range_example.sh
prab@8f7b90438d62516:~$ chmod a+x range_example.sh
prab@8f7b90438d62516:~$ nano range_example.sh
prab@8f7b90438d62516:~$ ./range_example.sh
1
2
3
4
5
6
7
8
9
10
series of numbers from 1 to 10
```

3) For loop to read a range with increment and decrement

```
prab@8f7b90438d62516: ~
  GNU nano 7.2                    range_example2.sh *
#!/bin/bash
for num in {1..10..1}
do
echo $num
done
```

```
prab@8f7b90438d62516: ~
prab@8f7b90438d62516:~$ touch range_example2.sh
prab@8f7b90438d62516:~$ chmod a+x range_example2.sh
prab@8f7b90438d62516:~$ nano range_example2.sh
prab@8f7b90438d62516:~$ ./range_example2.sh
1
2
3
4
5
6
7
8
9
10
prab@8f7b90438d62516:~$
```

```
prab@8f7b90438d62516: ~
  GNU nano 7.2                    range_example2.2.sh *
#!/bin/bash
for num in {10..0..1}
do
echo $num
done
```

```
prab@8f7b90438d62516: ~
-bash: ./range_example2.2.sh: No such file or directory
prab@8f7b90438d62516:~$ touch range_example2.2.sh
prab@8f7b90438d62516:~$ chmod a+x range_example2.2.sh
prab@8f7b90438d62516:~$ nano range_example2.2.sh
prab@8f7b90438d62516:~$ ./range_example2.2.sh
10
9
8
7
6
5
4
3
2
1
0
```

4) For loop to read array variables

```
prab@8f7b90438d62516: ~
  GNU nano 7.2                    arr_example.sh *
#!/bin/bash
arr=("welcome""to""javatpoint")
for i in "${arr[@]}"
do
echo $i
done
```

```
prab@8f7b90438d62516: ~
prab@8f7b90438d62516:~$ touch arr_example.sh
prab@8f7b90438d62516:~$ chmod +x arr_example.sh
prab@8f7b90438d62516:~$ nano arr_example.sh
prab@8f7b90438d62516:~$ ./arr_example.sh
welcometojavatpoint
```

5) For loop to read white spaces in string as word separator

```
prab@8f7b90438d62516: ~
  GNU nano 7.2                    string_example.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
```

```
prab@8f7b90438d62516: ~
prab@8f7b90438d62516:~$ touch string_example.sh
prab@8f7b90438d62516:~$ chmod +x string_example.sh
prab@8f7b90438d62516:~$ nano string_example.sh
prab@8f7b90438d62516:~$ ./string_example.sh
Let's
start
learning
from
Javatpoint.
```

6) For Loop to read each line in string as word

```
prab@8f7b90438d62516: ~
  GNU nano 7.2                    string_example2.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

                              [ Read 11 lines ]
^G Help        ^O Write Out   ^W Where Is    ^K Cut         ^T Execute
^X Exit        ^R Read File   ^\ Replace     ^U Paste       ^J Justify
```

```
prab@8f7b90438d62516: ~
prab@8f7b90438d62516:~$ touch string_example2.sh
prab@8f7b90438d62516:~$ chmod +x string_example2.sh
prab@8f7b90438d62516:~$ nano string_example2.sh
```

```
prab@8f7b90438d62516:~$ ./string_example2.sh
Let's start
learning from
Javatpoint.
```

7) For loop to Read three-expression

```
prab@8f7b90438d62516: ~/for_loop
  GNU nano 7.2                    example.sh *
#!/bin/bash
for((i=1;i<=10;i++))
do
echo "$i"
done
```

```
prab@8f7b90438d62516: ~/for_loop
prab@8f7b90438d62516:~/for_loop$ touch example.sh
prab@8f7b90438d62516:~/for_loop$ chmod +x example.sh
prab@8f7b90438d62516:~/for_loop$ nano example.sh
prab@8f7b90438d62516:~/for_loop$ ./example.sh
1
2
3
4
5
6
7
8
9
10
prab@8f7b90438d62516:~/for_loop$
```

8) For loop with a break statement

```
prab@8f7b90438d62516: ~/for_loop
  GNU nano 7.2                    break_example.sh
#!/bin/bash
#Table of 2
for table in {2..100..2}
do
echo $table
if [ $table == 20 ]; then
break
fi
done
```

```
prab@8f7b90438d62516: ~/for_loop
prab@8f7b90438d62516:~/for_loop$ touch break_example.sh
prab@8f7b90438d62516:~/for_loop$ chmod +x break_example.sh
prab@8f7b90438d62516:~/for_loop$ nano break_example.sh
prab@8f7b90438d62516:~/for_loop$ ./break_example.sh
2
4
6
8
10
12
14
16
18
20
```

9) For loop with continue statement

```
prab@8f7b90438d62516: ~/for_loop
  GNU nano 7.2                    continue_ex.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
```

```
prab@8f7b90438d62516: ~/for_loop
prab@8f7b90438d62516:~/for_loop$ touch continue_ex.sh
prab@8f7b90438d62516:~/for_loop$ chmod +x continue_ex.sh
prab@8f7b90438d62516:~/for_loop$ nano continue_ex.sh
prab@8f7b90438d62516:~/for_loop$ ./continue_ex.sh
1
2
3
4
5
16
17
18
19
20
prab@8f7b90438d62516:~/for_loop$
```

10) Infinite BASH for loop

```
prab@8f7b90438d62516: ~/for_loop
  GNU nano 7.2                    infinite.sh *
#!/bin/bash

i=1;
for (( ; ; ))
do
sleep 1s
echo "Current Number: $((i++))"
done
```

```
prab@8f7b90438d62516: ~/for_loop
prab@8f7b90438d62516:~/for_loop$ touch infinite.sh
prab@8f7b90438d62516:~/for_loop$ chmod +x infinite.sh
prab@8f7b90438d62516:~/for_loop$ nano infinite.sh
prab@8f7b90438d62516:~/for_loop$ ./infinite.sh
Current Number: 1
Current Number: 2
Current Number: 3
Current Number: 4
Current Number: 5
Current Number: 6
Current Number: 7
Current Number: 8
Current Number: 9
Current Number: 10
Current Number: 11
Current Number: 12
Current Number: 13
Current Number: 14
Current Number: 15
Current Number: 16
Current Number: 17
Current Number: 18
Current Number: 19
Current Number: 20
Current Number: 21
Current Number: 22
^C
prab@8f7b90438d62516:~/for_loop$
```

BASH WHILE

1.Simple while loop

```
prab@8f7b90438d62516: ~/while
  GNU nano 7.2                    script1.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."
```

```
prab@8f7b90438d62516: ~/while
prab@8f7b90438d62516:~/while$ touch script1.sh
prab@8f7b90438d62516:~/while$ chmod +x script1.sh
prab@8f7b90438d62516:~/while$ nano script1.sh
```

```
prab@8f7b90438d62516:~/while$ ./script1.sh
Enter starting number: 4
Enter ending number: 12
4
5
6
7
8
9
10
11
12
This is the sequence that you wanted.
```

2) Multiple condition while loop

```
prab@8f7b90438d62516: ~/while
  GNU nano 7.2                                   script2.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."
```

```
prab@8f7b90438d62516: ~/while
prab@8f7b90438d62516:~/while$ touch script2.sh
prab@8f7b90438d62516:~/while$ chmod +x script2.sh
prab@8f7b90438d62516:~/while$ nano script2.sh
prab@8f7b90438d62516:~/while$ ./script2.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.
prab@8f7b90438d62516:~/while$
```

3) Infinite While Loop

```
prab@8f7b90438d62516: ~/while
  GNU nano 7.2
#!/bin/bash
#An infinite while loop

while :
do
echo "Welcome to Javatpoint."
done
```

```
prab@8f7b90438d62516:~/while$ touch script3.sh
prab@8f7b90438d62516:~/while$ chmod +x script3.sh
prab@8f7b90438d62516:~/while$ nano script3.sh
prab@8f7b90438d62516:~/while$ ./script3.sh
```

```
Welcome to Javatpoint.
Welcome to Javatpoint.
Welcome to Javatpoint.
Welcome to Javatpoint.
Welcome to Javatpoint.
Welcome to Javatpoint.
Welcome to Javatpoint.^C
```

4) Break statement

prab@8f7b90438d62516: ~/while

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

prab@8f7b90438d62516: ~/while

```
prab@8f7b90438d62516:~/while$ touch script4.sh
prab@8f7b90438d62516:~/while$ chmod +x script4.sh
prab@8f7b90438d62516:~/while$ nano script4.sh
prab@8f7b90438d62516:~/while$ ./script4.sh
Countdown for Website Launching...
10
9
8
7
6
5
4
3
Mission Aborted, Some Technical Error Found.
prab@8f7b90438d62516:~/while$
```

5) While loop continue statement

prab@8f7b90438d62516: ~/while

```
  GNU nano 7.2
#!/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."
```

prab@8f7b90438d62516: ~/while

```
prab@8f7b90438d62516:~/while$ touch script5.sh
prab@8f7b90438d62516:~/while$ chmod +x script5.sh
prab@8f7b90438d62516:~/while$ nano script5.sh
prab@8f7b90438d62516:~/while$ ./script5.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.
```

6) While loop with C style

```
prab@8f7b90438d62516: ~/while
  GNU nano 7.2                                                    sc
#!/bin/bash
#While loop example in C style

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

```
prab@8f7b90438d62516:~/while$ touch script6.sh
prab@8f7b90438d62516:~/while$ chmod +x script6.sh
prab@8f7b90438d62516:~/while$ nano script6.sh
prab@8f7b90438d62516:~/while$ ./script6.sh
1
2
3
4
5
6
7
8
9
10
```

UNTIL

1) Until loop with single condition

prab@8f7b90438d62516: ~/until

```
  GNU nano 7.2
#!/bin/bash
#Bash Until Loop example with a single condition

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

prab@8f7b90438d62516: ~/until

```
prab@8f7b90438d62516:~/until$ touch script.sh
prab@8f7b90438d62516:~/until$ chmod +x script.sh
prab@8f7b90438d62516:~/until$ nano script.sh
prab@8f7b90438d62516:~/until$ ./script.sh
1
2
3
4
5
6
7
8
9
10
prab@8f7b90438d62516:~/until$
```

2) until loop with multiple conditions

prab@8f7b90438d62516: ~/until

```
  GNU nano 7.2
#!/bin/bash
#Bash Until Loop example with multiple conditions

max=5
a=1
b=0

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

prab@8f7b90438d62516: ~/until

```
prab@8f7b90438d62516:~/until$ touch script2.sh
prab@8f7b90438d62516:~/until$ chmod +x script2.sh
prab@8f7b90438d62516:~/until$ nano script2.sh
prab@8f7b90438d62516:~/until$ ./script2.sh
a = 1 & b = 0.
a = 2 & b = 1.
a = 3 & b = 2.
a = 4 & b = 3.
a = 5 & b = 4.
prab@8f7b90438d62516:~/until$
```

BASH STRING

1)Equal Operator

```
prab@8f7b90438d62516: ~/string
  GNU nano 7.2
#!/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
```

```
prab@8f7b90438d62516: ~/string
prab@8f7b90438d62516:~/string$ touch script.sh
prab@8f7b90438d62516:~/string$ chmod a+x script.sh
prab@8f7b90438d62516:~/string$ nano script.sh
prab@8f7b90438d62516:~/string$ ./script.sh
Strings are not equal.
```

2) Not equal operator

```
prab@8f7b90438d62516: ~/string
  GNU nano 7.2
#!/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
```

prab@8f7b90438d62516: ~/string

```
prab@8f7b90438d62516:~/string$ touch script2.sh
prab@8f7b90438d62516:~/string$ chmod +x script2.sh
prab@8f7b90438d62516:~/string$ nano script2.sh
prab@8f7b90438d62516:~/string$ ./script2.sh
Strings are not equal.
```

3) Less than operator

prab@8f7b90438d62516: ~/string

```
  GNU nano 7.2
#!/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
```

prab@8f7b90438d62516: ~/string

```
prab@8f7b90438d62516:~/string$ touch script3.sh
prab@8f7b90438d62516:~/string$ chmod +x script3.sh
prab@8f7b90438d62516:~/string$ nano script3.sh
prab@8f7b90438d62516:~/string$ ./script3.sh
WelcometoJavatpoint is not less then Javatpoint
```

4) Greater than operator

prab@8f7b90438d62516: ~/string

```
  GNU nano 7.2
#!/bin/sh
str1="WelcometoJavatpoint"
str2="Javatpoint"
if [ $str1 \> $str2 ];

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

prab@8f7b90438d62516: ~/string

```
prab@8f7b90438d62516:~/string$ touch script4.sh
prab@8f7b90438d62516:~/string$ chmod +x script4.sh
prab@8f7b90438d62516:~/string$ nano script4.sh
prab@8f7b90438d62516:~/string$ ./script4.sh
WelcometoJavatpoint is greater then Javatpoint
```

5) To check if string length is greater than zero

```
prab@8f7b90438d62516: ~/string
  GNU nano 7.2
#!/bin/sh
str="WelcometoJavatpoint"
if [ -n $str ];
then
echo "String is not empty"
else
echo "String is empty"
fi
```

```
prab@8f7b90438d62516: ~/string
prab@8f7b90438d62516:~/string$ touch script5.sh
prab@8f7b90438d62516:~/string$ chmod +x script5.sh
prab@8f7b90438d62516:~/string$ nano script5.sh
prab@8f7b90438d62516:~/string$ ./script5.sh
String is not empty
```

6) To check if the string length is equal to zero

prab@8f7b90438d62516: ~/string

```
GNU nano 7.2
#!/bin/sh
str=""
if [ -z $str ];
then
echo "String is empty."
else
echo "String is non-empty."
fi
```

prab@8f7b90438d62516: ~/string

```
prab@8f7b90438d62516:~/string$ touch script6.sh
prab@8f7b90438d62516:~/string$ chmod +x script6.sh
prab@8f7b90438d62516:~/string$ nano script6.sh
prab@8f7b90438d62516:~/string$ ./script6.sh
String is empty.
```

BASH FIND

1) ${#string}

prab@8f7b90438d62516: ~/string_find

```
  GNU nano 7.2
#!/bin/bash
#Bash program to find the length of a string

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

echo "Length of '$str' is $length"
```

prab@8f7b90438d62516: ~/string_find

```
prab@8f7b90438d62516:~/string_find$ touch example1.sh
prab@8f7b90438d62516:~/string_find$ chmod +x example1.sh
prab@8f7b90438d62516:~/string_find$ nano example1.sh
prab@8f7b90438d62516:~/string_find$ ./example1.sh
Length of 'Welcome to Javatpoint' is 21
```

2) ` expr length “$str”`

prab@8f7b90438d62516: ~/string_find

```
  GNU nano 7.2
#!/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"
```

prab@8f7b90438d62516: ~/string_find

```
prab@8f7b90438d62516:~/string_find$ touch example2.sh
prab@8f7b90438d62516:~/string_find$ chmod +x example2.sh
prab@8f7b90438d62516:~/string_find$ nano example2.sh
prab@8f7b90438d62516:~/string_find$ ./example2.sh
Length of 'Welcome to Javatpoint' is 21
```

3) `expr "$str" : '.*'`

prab@8f7b90438d62516: ~/string_find

```
  GNU nano 7.2
#!/bin/bash
#Bash script to find the length of a string

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

echo "Length of '$str' is $length"
```

prab@8f7b90438d62516: ~/string_find

```
prab@8f7b90438d62516:~/string_find$ touch example3.sh
prab@8f7b90438d62516:~/string_find$ chmod +x example3.sh
prab@8f7b90438d62516:~/string_find$ nano example3.sh
```

```
prab@8f7b90438d62516:~/string_find$ ./example3.sh
Length of 'Welcome to Javatpoint' is 21
```

4) ‘wc commad’

```
prab@8f7b90438d62516: ~/string_find
  GNU nano 7.2                                          example4.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"
```

```
prab@8f7b90438d62516: ~/string_find
prab@8f7b90438d62516:~/string_find$ touch example4.sh
prab@8f7b90438d62516:~/string_find$ chmod +x example4.sh
prab@8f7b90438d62516:~/string_find$ nano example4.sh
prab@8f7b90438d62516:~/string_find$ ./example4.sh
Length of 'Welcome to Javatpoint' is 22
```

5) ‘awk’ command

```
prab@8f7b90438d62516: ~/string_find
  GNU nano 7.2
#!/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"
```

Select prab@8f7b90438d62516: ~/string_find

```
prab@8f7b90438d62516:~/string_find$ touch example6.sh
prab@8f7b90438d62516:~/string_find$ chmod +x example6.sh
prab@8f7b90438d62516:~/string_find$ nano example6.sh
prab@8f7b90438d62516:~/string_find$ ./example6.sh
Length of 'Welcome to Javatpoint' is 21
prab@8f7b90438d62516:~/string_find$
```

STRING SPLIT

1) BASH split string by space

prab@8f7b90438d62516: ~/split_string

```
GNU nano 7.2                                                    scr
#!/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
```

prab@8f7b90438d62516: ~/split_string

```
prab@8f7b90438d62516:~/split_string$ touch script1.sh
prab@8f7b90438d62516:~/split_string$ chmod a+x script1.sh
prab@8f7b90438d62516:~/split_string$ nano script1.sh
```

```
prab@8f7b90438d62516:~/split_string$ ./script1.sh
Enter any string separated by space: my name is prabhakar
my
name
is
prabhakar
prab@8f7b90438d62516:~/split_string$ nano script1.sh
prab@8f7b90438d62516:~/split_string$
```

2) BASH split string by symbol

```
#!/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]}"
```

prab@8f7b90438d62516: ~/split_string

```
prab@8f7b90438d62516:~/split_string$ touch script2.sh
prab@8f7b90438d62516:~/split_string$ chmod +x script2.sh
prab@8f7b90438d62516:~/split_string$ nano script2.sh
```

```
prab@8f7b90438d62516:~/split_string$ nano script2.sh
prab@8f7b90438d62516:~/split_string$ ./script2.sh
Enter Name, State and Age separated by a comma: Prabhakar, Bihar, 22
Name : Prabhakar
State :  Bihar
Age :  22
```

SPLIT Without $IFS variable

1) BASH split string by symbol

```
  GNU nano 7.2                                                    script3.sl
#!/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
```

```
prab@8f7b90438d62516: ~/split_string
prab@8f7b90438d62516:~/split_string$ touch script3.sh
prab@8f7b90438d62516:~/split_string$ chmod +x script3.sh
prab@8f7b90438d62516:~/split_string$ nano script3.sh
prab@8f7b90438d62516:~/split_string$ ./script3.sh
Enter any string separated by colon(:) we:welcome:you:on:javatpoint

we
welcome
you
on
javatpoint
```

2) BASH split string by another string

```
prab@8f7b90438d62516: ~/split_string
  GNU nano 7.2                    script4.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
```

```
prab@8f7b90438d62516: ~/split_string
prab@8f7b90438d62516:~/split_string$ touch script4.sh
prab@8f7b90438d62516:~/split_string$ chmod +x script4.sh
prab@8f7b90438d62516:~/split_string$ nano script4.sh
prab@8f7b90438d62516:~/split_string$ ./script4.sh
declare -a array=([0]="We" [1]="Welcome" [2]="You" [3]="On" [4]="Javatpoint"
)
```

3) BASH split trim using Trim command

```
prab@8f7b90438d62516: ~/split_string
  GNU nano 7.2                    script5.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
```

```
prab@8f7b90438d62516: ~/split_string
prab@8f7b90438d62516:~/split_string$ touch script5.sh
prab@8f7b90438d62516:~/split_string$ chmod a+x script5.sh
prab@8f7b90438d62516:~/split_string$ nano script5.sh
prab@8f7b90438d62516:~/split_string$ ./script5.sh
tr: missing operand after ';\\n'
Two strings must be given when translating.
Try 'tr --help' for more information.
prab@8f7b90438d62516:~/split_string$ nano script5.sh
prab@8f7b90438d62516:~/split_string$ ./script5.sh
We
welcome
you
on
javatpoint.
```

BASH Substring

1) extract till Specific characters from String

```
prab@8f7b90438d62516: ~/substring
  GNU nano 7.2                    script1.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} "
```

```
prab@8f7b90438d62516: ~/substring
prab@8f7b90438d62516:~/substring$ touch script1.sh
prab@8f7b90438d62516:~/substring$ chmod +x script1.sh
prab@8f7b90438d62516:~/substring$ nano script1.sh
prab@8f7b90438d62516:~/substring$ ./script1.sh
String: We welcome you on Javatpoint.
Total characters in a String: 29
Substring: We welcome
Total characters in Substring: 10
prab@8f7b90438d62516:~/substring$
```

2) To extract from Specific Character onwards

```
prab@8f7b90438d62516: ~/substring
  GNU nano 7.2                    script2.sh *
#!/bin/bash
#Script to print from 11th character onwards
str="We welcome you on Javatpoint."
substr="${str:11}"
echo "$substr"
```

```
prab@8f7b90438d62516: ~/substring
prab@8f7b90438d62516:~/substring$ touch script2.sh
prab@8f7b90438d62516:~/substring$ chmod +x script2.sh
prab@8f7b90438d62516:~/substring$ nano script2.sh
prab@8f7b90438d62516:~/substring$ ./script2.sh
you on Javatpoint.
prab@8f7b90438d62516:~/substring$
```

3) To extract a single character

```
prab@8f7b90438d62516: ~/substring
  GNU nano 7.2                    script3.sh *
#!/bin/bash
#Script to print 11th character of a String
str="We welcome you on Javatpoint."
substr="${str:11:1}"
echo "$substr"
```

```
prab@8f7b90438d62516: ~/substring
prab@8f7b90438d62516:~/substring$ touch script3.sh
prab@8f7b90438d62516:~/substring$ chmod +x script3.sh
prab@8f7b90438d62516:~/substring$ nano script3.sh
prab@8f7b90438d62516:~/substring$ ./script3.sh
y
prab@8f7b90438d62516:~/substring$
```

4) To extract specific characters from last

```
prab@8f7b90438d62516: ~/substring
  GNU nano 7.2                    script4.sh *
#!/bin/bash
#Script to extract 11 characters from last
str="We welcome you on Javatpoint."
substr="${str:(-11)}"
echo "$substr"
```

```
prab@8f7b90438d62516: ~/substring
prab@8f7b90438d62516:~/substring$ touch script4.sh
prab@8f7b90438d62516:~/substring$ chmod +x script4.sh
prab@8f7b90438d62516:~/substring$ nano script4.sh
prab@8f7b90438d62516:~/substring$ ./script4.sh
Javatpoint.
prab@8f7b90438d62516:~/substring$
```

Bash Concatenate String

1) Write variables side by side

```
prab@8f7b90438d62516: ~/concatenate
  GNU nano 7.2                    script1.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
```

```
prab@8f7b90438d62516: ~/concatenate
prab@8f7b90438d62516:~/concatenate$ touch script1.sh
prab@8f7b90438d62516:~/concatenate$ chmod +x script1.sh
prab@8f7b90438d62516:~/concatenate$ nano script1.sh
prab@8f7b90438d62516:~/concatenate$ ./script1.sh
We welcome you on Javatpoint.
```

2) Using Double Quotes

```
prab@8f7b90438d62516: ~/concatenate
  GNU nano 7.2                    script2.sh *
#!/bin/bash
#Script to Concatenate Strings
#Declaring String Variable
str="We welcome you"
#Add the variable within the string
echo "$str on Javatpoint."
```

```
prab@8f7b90438d62516: ~/concatenate
prab@8f7b90438d62516:~/concatenate$ touch script2.sh
prab@8f7b90438d62516:~/concatenate$ chmod +x script2.sh
prab@8f7b90438d62516:~/concatenate$ nano script2.sh
prab@8f7b90438d62516:~/concatenate$ ./script2.sh
We welcome you on Javatpoint.
prab@8f7b90438d62516:~/concatenate$
```

3) Using append operator with loop

```
prab@8f7b90438d62516: ~/concatenate
  GNU nano 7.2                    script3.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"
```

```
prab@8f7b90438d62516: ~/concatenate
prab@8f7b90438d62516:~/concatenate$ touch script3.sh
prab@8f7b90438d62516:~/concatenate$ chmod +x script3.sh
prab@8f7b90438d62516:~/concatenate$ nano script3.sh
prab@8f7b90438d62516:~/concatenate$ ./script3.sh
Printing the name of the programming languages
javapythonCC++
prab@8f7b90438d62516:~/concatenate$
```

4) Using the printf function

```
prab@8f7b90438d62516: ~/concatenate
  GNU nano 7.2                    script4.sh *
#!/bin/bash
str="Welcome"
printf -v new_str "$str to Javatpoint."
echo $new_str
```

```
prab@8f7b90438d62516: ~/concatenate
prab@8f7b90438d62516:~/concatenate$ touch script4.sh
prab@8f7b90438d62516:~/concatenate$ chmod +x script4.sh
prab@8f7b90438d62516:~/concatenate$ nano script4.sh
prab@8f7b90438d62516:~/concatenate$ ./script4.sh
Welcome to Javatpoint.
prab@8f7b90438d62516:~/concatenate$
```

5) Using Literal Strings

```
prab@8f7b90438d62516: ~/concatenate
  GNU nano 7.2                    script5.sh *
#!/bin/bash
str="Welcome to"
newstr="${str} Javatpoint."
echo "$newstr"
```

```
Select prab@8f7b90438d62516: ~/concatenate
prab@8f7b90438d62516:~/concatenate$ touch script5.sh
prab@8f7b90438d62516:~/concatenate$ chmod +x script5.sh
prab@8f7b90438d62516:~/concatenate$ nano script5.sh
prab@8f7b90438d62516:~/concatenate$ ./script5.sh
Welcome to Javatpoint.
prab@8f7b90438d62516:~/concatenate$
```

6) Using Underscore

```
prab@8f7b90438d62516: ~/concatenate
  GNU nano 7.2                    script6.sh *
#!/bin/bash
str1="Hello"
str2="World!"
echo "${str1}_${str2}"
```

```
prab@8f7b90438d62516: ~/concatenate
prab@8f7b90438d62516:~/concatenate$ touch script6.sh
prab@8f7b90438d62516:~/concatenate$ chmod +x script6.sh
prab@8f7b90438d62516:~/concatenate$ nano script6.sh
prab@8f7b90438d62516:~/concatenate$ ./script6.sh
Hello_World!
prab@8f7b90438d62516:~/concatenate$
```

7) Using any character

```
prab@8f7b90438d62516: ~/concatenate
  GNU nano 7.2                    script7.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"
```

```
prab@8f7b90438d62516: ~/concatenate
prab@8f7b90438d62516:~/concatenate$ touch script7.sh
prab@8f7b90438d62516:~/concatenate$ chmod +x script7.sh
prab@8f7b90438d62516:~/concatenate$ nano script7.sh
```

```
prab@8f7b90438d62516:~/concatenate$ ./script7.sh
Enter First Name: prabhakar
Enter State: bihar
Enter Age: 22
Name, State, Age: prabhakar,bihar,22
```

FUNCTION

1) Function type 1

```
prab@8f7b90438d62516: ~/function
  GNU nano 7.2                          script1.sh *
#!/bin/bash
JTP () {
echo 'Welcome to Javapoint'
}

JTP
```

```
prab@8f7b90438d62516: ~/function
prab@8f7b90438d62516:~/function$ touch script1.sh
prab@8f7b90438d62516:~/function$ chmod +x script1.sh
prab@8f7b90438d62516:~/function$ nano script1.sh
prab@8f7b90438d62516:~/function$ ./script1.sh
Welcome to Javapoint
```

2) Function type 2

```
prab@8f7b90438d62516: ~/function
  GNU nano 7.2                          script2.sh *
#!/bin/bash

function JTP {
echo 'Welcome to Javatpoint'
}

JTP
```

```
prab@8f7b90438d62516: ~/function
prab@8f7b90438d62516:~/function$ touch script2.sh
prab@8f7b90438d62516:~/function$ chmod +x script2.sh
prab@8f7b90438d62516:~/function$ nano script2.sh
prab@8f7b90438d62516:~/function$ ./script2.sh
Welcome to Javatpoint
prab@8f7b90438d62516:~/function$
```

3) Passing Argument

```
prab@8f7b90438d62516: ~/function
  GNU nano 7.2                    script3.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."
```

```
prab@8f7b90438d62516: ~/function
prab@8f7b90438d62516:~/function$ ./script3.sh
prab@8f7b90438d62516:~/function$ nano script3.sh
```

```
prab@8f7b90438d62516:~/function$ ./script3.sh
We
welcome
you
on
Javatpoint.
```

4) Variable Scope

```
prab@8f7b90438d62516: ~/function
  GNU nano 7.2                    script4.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."
```

```
prab@8f7b90438d62516: ~/function
prab@8f7b90438d62516:~/function$ touch script4.sh
prab@8f7b90438d62516:~/function$ chmod +x script4.sh
prab@8f7b90438d62516:~/function$ nano script4.sh
prab@8f7b90438d62516:~/function$ ./script4.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.
prab@8f7b90438d62516:~/function$
```

5) Return Values

```
prab@8f7b90438d62516: ~/function
  GNU nano 7.2                    script6.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 $?
```

```
prab@8f7b90438d62516: ~/function
prab@8f7b90438d62516:~/function$ touch script6.sh
prab@8f7b90438d62516:~/function$ chmod +x script6.sh
prab@8f7b90438d62516:~/function$ nano script6.sh
prab@8f7b90438d62516:~/function$ ./script6.sh
Hello User
Hello Reader
The previous function returned a value of 5
prab@8f7b90438d62516:~/function$
```

6) Overriding Commands

```
prab@8f7b90438d62516: ~/function
  GNU nano 7.2                          script7.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."
```

```
prab@8f7b90438d62516: ~/function
prab@8f7b90438d62516:~/function$ touch script7.sh
prab@8f7b90438d62516:~/function$ chmod +x script7.sh
prab@8f7b90438d62516:~/function$ nano script7.sh
prab@8f7b90438d62516:~/function$ ./script7.sh
[01-30 06:36:39] : Welcome to Javatpoint.
prab@8f7b90438d62516:~/function$
```

ARRAY

1) Printing element with Index

```
prab@8f7b90438d62516: ~/array
  GNU nano 7.2                    script1.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]}
```

```
prab@8f7b90438d62516: ~/array
prab@8f7b90438d62516:~/array$ touch script1.sh
prab@8f7b90438d62516:~/array$ chmod +x script1.sh
prab@8f7b90438d62516:~/array$ nano script1.sh
prab@8f7b90438d62516:~/array$ ./script1.sh
Javatpoint
prab@8f7b90438d62516:~/array$
```

2) Printing all elements of the array

```
prab@8f7b90438d62516: ~/array
  GNU nano 7.2                    script2.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[@]}"
```

```
prab@8f7b90438d62516: ~/array
prab@8f7b90438d62516:~/array$ touch script2.sh
prab@8f7b90438d62516:~/array$ chmod +x script2.sh
prab@8f7b90438d62516:~/array$ nano script2.sh
prab@8f7b90438d62516:~/array$ ./script2.sh
WelcomeToJavatpoint
```

3) Printing the keys of an Array

```
prab@8f7b90438d62516: ~/array
  GNU nano 7.2                    script3.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[@]}"
```

```
prab@8f7b90438d62516: ~/array
prab@8f7b90438d62516:~/array$ touch script3.sh
prab@8f7b90438d62516:~/array$ chmod +x script3.sh
prab@8f7b90438d62516:~/array$ nano script3.sh
prab@8f7b90438d62516:~/array$ ./script3.sh
0 1 2
prab@8f7b90438d62516:~/array$
```

4) Finding Array Length

```
Select prab@8f7b90438d62516: ~/array
  GNU nano 7.2                    script4.sh *
#!/bin/bash
#Declaring the Array
declare -a example_array=( "Welcome""To""Javatpoint" )
#Printing Array Length
echo "The array contains ${#example_array[@]} elements"
```

```
prab@8f7b90438d62516: ~/array
prab@8f7b90438d62516:~/array$ touch script4.sh
prab@8f7b90438d62516:~/array$ nano script4.sh
prab@8f7b90438d62516:~/array$ chmod +x script4.sh
prab@8f7b90438d62516:~/array$ ./script4.sh
The array contains 3 elements
prab@8f7b90438d62516:~/array$
```

5) Loop through the Array

```
prab@8f7b90438d62516: ~/array
  GNU nano 7.2                    script5.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
```

```
prab@8f7b90438d62516: ~/array
prab@8f7b90438d62516:~/array$ touch script5.sh
prab@8f7b90438d62516:~/array$ chmod +x script5.sh
prab@8f7b90438d62516:~/array$ nano script5.sh
prab@8f7b90438d62516:~/array$ ./script5.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
prab@8f7b90438d62516:~/array$
```

6) Adding elements to an Array

```
prab@8f7b90438d62516: ~/array
  GNU nano 7.2                    script6.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
```

```
prab@8f7b90438d62516: ~/array
prab@8f7b90438d62516:~/array$ touch script6.sh
prab@8f7b90438d62516:~/array$ chmod +x script6.sh
prab@8f7b90438d62516:~/array$ nano script6.sh
prab@8f7b90438d62516:~/array$ ./script6.sh
0 Welcome
1 To
2 Javatpoint
prab@8f7b90438d62516:~/array$
```

7) Updating Array Elements

```
prab@8f7b90438d62516: ~/array
  GNU nano 7.2                    script7.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[@]}"
```

```
prab@8f7b90438d62516: ~/array
prab@8f7b90438d62516:~/array$ touch script7.sh
prab@8f7b90438d62516:~/array$ chmod +x script7.sh
prab@8f7b90438d62516:~/array$ nano script7.sh
prab@8f7b90438d62516:~/array$ ./script7.sh
Java Python PHP HTML JavaScript
prab@8f7b90438d62516:~/array$
```

8) Deleting an element from an Array

```
prab@8f7b90438d62516: ~/array
  GNU nano 7.2                    script8.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[@]}
```

```
prab@8f7b90438d62516: ~/array
prab@8f7b90438d62516:~/array$ touch script8.sh
prab@8f7b90438d62516:~/array$ chmod +x script8.sh
prab@8f7b90438d62516:~/array$ nano script8.sh
prab@8f7b90438d62516:~/array$ ./script8.sh
We welcome you on Javatpoint
prab@8f7b90438d62516:~/array$
```

9) Delete the entire Array

```
prab@8f7b90438d62516: ~/array
  GNU nano 7.2                    script9.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[@]}"
```

```
prab@8f7b90438d62516: ~/array
prab@8f7b90438d62516:~/array$ touch script9.sh
prab@8f7b90438d62516:~/array$ chmod +x script9.sh
prab@8f7b90438d62516:~/array$ nano script9.sh
prab@8f7b90438d62516:~/array$ ./script9.sh
Java HTML CSS JavaScript
prab@8f7b90438d62516:~/array$
```

10) Slice Array Elements

```
prab@8f7b90438d62516: ~/array
  GNU nano 7.2                    script10.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[@]}
```

prab@8f7b90438d62516: ~/array

```
prab@8f7b90438d62516:~/array$ touch script10.sh
prab@8f7b90438d62516:~/array$ chmod +x script10.sh
prab@8f7b90438d62516:~/array$ nano script10.sh
prab@8f7b90438d62516:~/array$ ./script10.sh


prab@8f7b90438d62516:~/array$
```