# Shell Script Examples

## Example 01: Variable

```
#!/bin/bash

var=1234;        echo "$var"           #1234
var=ab cdd;      echo "$var"           #Error
var="abcd";      echo "$var"           #abcd
var="ab cd";     echo "$var"           #ab cd
var=1234;        echo "$var"           #1234
var="1234";      echo "$var"           #1234
var="\"abcd\""; echo "$var"           #"abcd"
var=\"abcd\";    echo "$var"           #"abcd"

echo "Enter any value: "
read val
echo "Entered value: $val"

read -p "Enter any value: " val
echo "Entered value: $val"
```

## Example 02: Operators

```
#!/bin/bash

read -p "Enter Num1: " num1
read -p "Enter Num2: " num2

# Arithmetic operations
echo "Addition: $(expr $num1 + $num2)"
echo "Subtraction: $(expr $num1 - $num2)"
echo "Division: $(expr $num1 / $num2)"
echo "Multiplication: $(expr $num1 \* $num2)"
echo "Reminder: $(expr $num1 % $num2)"

# Numeric Comparisons
expr $num1 = $num2
expr $num1 != $num2
expr $num1 \< $num2
expr $num1 \> $num2
expr $num1 \<= $num2
expr $num1 \>= $num2

#String Comparisons
expr Linux : Lin
expr Linux : Linx
expr Linux : Linux
```

**Example 03: String comparisons**

```
#!/bin/bash

# String comparison

test abc = abc;   echo "$?" #0:Success
test abd = abc;   echo "$?" #1:Failed
test abc != abc; echo "$?"  #1:Failed
test abd != abc; echo "$?"  #0:Success

# Numeric Comparions
test 10 -lt 100; echo "$?"  #0:Success
test 10 -le 100; echo "$?"  #0:Success
test 10 -gt 100; echo "$?"  #1:Failed
test 10 -ge 100; echo "$?"  #1:Failed
test 10 -eq 100; echo "$?"  #1:Failed
test 10 -ne 100; echo "$?"  #0:Success
```

**Example 04: if statement**

```
#!/bin/bash
# Identify even or Odd numbers

read -p "Enter number: " num
if [ $(( $num % 2 )) -eq 0 ]
then
      echo "Even Number"
else
      echo "Odd Number"
fi
```

**Example 05: case statement**

```
#!/bin/bash

case $1 in
"red"|"RED"|???)
      echo "red color"
;;
"green")
      echo "green color"
```

```
;;
"blue")
     echo "blue color"
;;
*)
     echo "other color"
esac
```

**Example 06: Logical operations**

```
#!/bin/bash

if test $1 = "red" -o $1 = "RED"
then
     echo "red color"
elif test $1 = "yellow"
then
     echo "yellow color"
elif test $1 = "green"
then
     echo "green color"
else
     echo "unknown color"
fi
```

**Example 07: Checking exit status of command**

```
#!/bin/bash

#if  command
#then
#    stmt
#fi

#exit status of a command can be measured with $?

pwd
if [ $? -eq 0 ]
then
     echo "success0"
fi

if pwd
then
     echo "success1"
fi
###############################
```

```
pwd > /dev/null
if [ $? -eq 0 ]
then
     echo "success0"
fi

if pwd > /dev/null
then
     echo "success2"
fi

###############################

pwd > /dev/null
if [ $? -eq 0 ]
then
     echo "success0"
fi > /dev/null

if pwd
then
     echo "success3"
fi > /dev/null
```

**Example 08: Redirection concept with conditional statement**

```
#!/bin/bash

if pwd
then
     echo "success0" > /dev/null
     echo "success1"
fi


if pwd
then
     echo "success10"
     echo "success11"
fi > /dev/null
```

**Example 09: Case statement**

```
#!/bin/bash
```

```
#echo "${#1}"
if [ ${#1} -ne 3 ]
then
     echo "Error: Total chars are not 3"
     exit 1;
fi

case $1 in
[a-zA-Z][A-Za-z][a-zA-Z])
     echo "all alphabets"
     ;;
[0-9][0-9][0-9])
     echo "all digits"
     ;;
*)
     echo "Mixture"
esac
```

**Example 10: While loop to print 5-1 in reverse order**

```
#!/bin/bash

#while command
#do
#    stmt
#done
cnt=5
while test $cnt -ge 0
do
     echo "$cnt"
     cnt=`expr $cnt - 1`
done
```

**Example 11: While Loop: script similar to cat command**

```
#!/bin/bash

while read line
do
     echo "$line"
done
```

**Example 12: Reading file with While loop**

```
#!/bin/bash

while read line
do
	echo "$line"
#	read line
done < cricket
```

**Example 13: Until Loop to print 0-10**

```
#!/bin/bash

#until command
#do
#	stmt
#done


cnt=0
until test $cnt -eq 11
do
	echo "$cnt"
	cnt=`expr $cnt + 1`
done
```

**Example 14: For Loop**

```
#!/bin/bash

#for i in list
#do
#	stmt
#done

for i in {1..5}
do
	echo "$i"
done

echo "*******************************";

for i in 10 3 50 13 50 a b abc
do
```

```
     echo "$i"
done;
```

**Example 15: For loop**

```
#!/bin/bash

#for i in list
#do
#    stmt
#done

for i in {1..}
do
     echo "Processed element : $i"
done

for i in 1{a..f}{p..q}
do
     echo "$i"
done
```

**Example 16: continue statement**

```
#!/bin/bash

for i
do
     if [ $i = 10 ]
     then
          continue
     fi
     echo "$i"
done
echo "out of loop"
```

**Example 17: break statement**

```
#!/bin/bash

for i
do
     echo "$i"
     if [ $i = 10 ]
     then
          break;
     fi
done
echo "out of loop"
```

**Example 18: Array**

```
#!/bin/bash

arr=(sanjay ajay vijay);

echo "arr[0] : ${arr[0]}";
echo "arr[1] : ${arr[1]}";
echo "arr[2] : ${arr[2]}";
IFS=?
echo "All elements arr[*] : ${arr[*]}";
echo "All elements arr[@] : ${arr[@]}";

echo "Total elements [@] : ${#arr[@]}";
echo "Total elements : ${#arr}";
```

**Example 19: Array**

```
#!/bin/bash

var=sample
echo "${var[0]}"
var[5]="somedata"
echo "${var[@]}"

fruits=("apple" "banana" "mango")
echo "$fruits"
```

**Example 20: Array**

```
#!/bin/bash
arr=("ash" "ksh" "b ash")
arr[10]="banana"
arr[5]="mango"
arr[8]="apple"

for i in {0..10}
```

```
do
     echo "arr[$i]: ${arr[$i]}"
done

IFS=?

echo "${arr[*]}"
echo "${arr[@]}"

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

echo "Total array elements: ${#arr[@]}"
IFS=","
echo -e "Array elements:\narr[@]: ${arr[@]}\narr[*]: ${arr[*]} "
```

**Example 21: Count total number of character in String**

```
#!/bin/bash

read line;
echo "${#line}"
```

**Example 22: Command line Arguments**

```
#!/bin/bash

IFS="?"

echo "Total parameters: $#"
echo "All parameters\(IFS\): $*"
echo "All parameters: $@"

echo "Script Name: $0"
echo "First parameters: $1"
echo "Second parameters: $2"
```

**Example 23: Command line Arguments: Access 10th Argument**

```
#!/bin/bash
```

```
echo "Total parameters: $#"
echo "All parameters\(IFS\): $*"
echo "All parameters: $@"

echo "Script Name: $0"
echo "First parameters: $1"
echo "Second parameters: $2"
echo "Tenth parameters: ${10}"
```

**Example 24: Accessing Command line Arguments with shift command**

```
#!/bin/bash

echo Total parameters: $#
echo All parameters\(IFS\): $@
echo All parameters: $*

echo Script Name: $0
echo First parameters: $1
echo Second parameters: $2

# General errors
cnt=1; echo "First parameters: $$cnt"
cnt=2; echo "Second parameters: $$cnt"
cnt=10; echo "Tenth parameters: ${$cnt}"

shift 1
echo "Total parameters: $#"
echo "All parameters: $*"

shift 2
echo "Total parameters: $#"
echo "All parameters: $*"
```

**Example 25: Command Line Arguments**

```
#!/bin/bash

if test $1 = 1
then
     echo "one"
elif test $1 = 2
then
     echo "two"
```

```
fi

if test $1 = 1
then
     echo "one"
else
     echo "else part"
     if test $1 = 2
     then
          echo "two"
     fi
fi
```

**Example 26: Restricting Command line Arguments**

```
#!/bin/bash
#Restrict number of arguments supplied to script

if [ $# -ne 2 ]
then
     echo -e "Error.\nCorrect Usage: bash $0 <num1> <num2>"
     exit 127
fi

echo "Result : `expr $1 + $2` "
```

**Example 27: Processing Command line argument while loop**

```

#!/bin/bash

while test ! -z $1
do
     echo "$@"
     shift 1
done
```

**Example 28: Processing Command line argument for loop**

```

#!/bin/bash
IFS=,
for i in $*
do
     echo "$i";
```

```
done
```

**Example 29: Processing Command line argument for loop**

```
#!/bin/bash
for i in $@
do
     echo "$i"
done
```

**Example 30: Using IFS**

```
#!/bin/bash
# IFS: Internal Field Separator
IFS="+"
echo "[$*]"
sum=`echo "$*" | bc`
echo "$sum"
```

**Example 31: Storing Command line arguments in array**

```
#!/bin/bash
arr=($@)
echo -e "Array elements: \n ${arr[@]}"
```

**Example 32: Positional Parameters with set command**

```
#!/bin/bash

set `grep -v "^$" logindata.txt | sed 's/LOGIN    :  //g' | sed
's/PASSWORD :  //g' | tr '\n' '\t'`

#echo -e "$1\t$2\n";
#shift 2;
#echo -e "$1\t$2\n";


#until (shift 2)
#do
#    echo -e "$1\t$2\n";
```

```
#done;
```

**Example 33: File handling with While loop**

```
#!/bin/bash

while read line
do
     set `echo $line`;
     echo $1

done < data
```

**Example 34: File handling with While loop**

```
#!/bin/bash

while read line
do
     echo "$line" | fgrep "log"
done < $1

fgrep "log" $1 | while  read line; do echo "$line"; done;
```

**Example 35: Checking current shell pid**

```
#!/bin/bash

echo "Pid of current shell: $$"
```

**Example 36: Functions**

```
#!/bin/bash

function add()
{
     echo "I got total $# args..."
     echo "These are agrs: $@"
     echo `echo "$@"| tr " " "+" | bc`
}
```

```
add 1 3 5 6
add 1 3 5 6
add 3 3 5 6
add 1 53 5 6
add 1 53 5 6
```

**Example 37: Functions stored in other files. Function contains single return statement**

Function.sh

```
#!/bin/bash

function greet()
{
	echo "Hello...."
	return -1;

}

greet
echo $?

#greet;
```

Calling script

```
#!/bin/bash

. function.sh

echo "before Function call: "
greet
echo "After Function call: "
```

**Example 38: Functions stored in other files. Function contains multiple return statement**

isnumber.sh

```
#!/bin/bash

function isnumber()
{
	if echo "$1" | egrep "^[0-9]+$"
	then
		echo "Valid number"
```

```
            return 0;
      else
            echo "Not valid"
            return 1;
      fi
}
#result=`isnumber $1`
#echo "[$?]"
#echo "$result"
```

Calling script

```
#!/bin/bash

. isnumber.sh

result=0
for i in $@
do
      if isnumber $i
      then
            result=$(($i+$result))
      fi >> out
done
echo "For loop result: $result"

#IFS="+"
#sum=`echo "$*" | bc`
#echo "$sum"
#echo "[`echo "$@" | tr " " "+" | bc`]"
```

**Example 39: HERE doc**

```
#!/bin/bash

while read line
do
      set $line;
      echo "$1";
done << EOF
THIS is a row
IS
LINE 1
EOF
```

**Example 40: Mysql Database connectivity with Here DOC**

```
#!/bin/bash

mysql -u demo<<EOF
use test;
select * from player;
EOF
```

**Example 41: Database connectivity**

```
#!/bin/bash

#mysql -u demo -D test
#select * from player123;

echo "select * from player123;" | mysql -u demo -D test
```

**Example 42: File Handling**

Data file : emp

```
1000:amol j:20000:40

1001:sujit:20000:40

1002:sonali:10000:30

1003:sanjeev:3000:50

1004:sanjay j:40000:60

1005:vineeta:8000:30

2000:uday:40000:20
```

Script code:

```
clear screen
while true
do
clear
```

```
tput cup 5 10 ; echo "Emp No:"
tput cup 5 20 ; read eno
tno=`grep "^$eno:" emp | cut -d":" -f1`

if [ "$tno" = "$eno" ]
then
          tname=`grep "^$eno:" emp | cut -d":" -f2`
          tsal=`grep "^$eno:" emp | cut -d":" -f3`
          tdno=`grep "^$eno:" emp | cut -d":" -f4`


          tput cup 7 10 ; echo "Ename:"
          tput cup 7 20 ; echo $tname
          tput cup 9 10 ; echo "Salary:"
          tput cup 9 20 ; echo $tsal
          tput cup 11 10 ; echo "DeptNo:"
          tput cup 11 20 ; echo $tdno
          sleep 5
else
          tput cup 10 10 ; echo Record does not exist
          sleep 2
fi

          tput cup 12 10 ; echo "Want to continue?"
          tput cup 12 30 ; read ans

          case $ans in
                    y|Y) continue
                              ;;
                   n|N) exit
                              ;;
          esac
done
```