Bash If Else

Let Us understand how to use if-else statements in Bash scripts to get our automated tasks completed.

Bash if-else statements are used to perform conditional tasks in the sequential flow of execution of statements. Sometimes, we want to process a specific set of statements if a condition is true, and another set of statements if it is false. To perform such type of actions, we can apply the if-else mechanism. We can apply the condition with the 'if statement'.

Bash If Else Syntax

A syntax of if-else statement in Bash Shell Scripting can be defined as below:

```
➢ if [ condition ];
➢ then
➢   <if block commands>
➢ else
➢   <else block commands>
➢ fi
```

Important Points to Remember

1. We can use a set of one or more conditions joined using conditional operators.
2. Else block commands includes a set of actions to perform when the condition is false.
3. The semi-colon (;) after the conditional expression is a must.

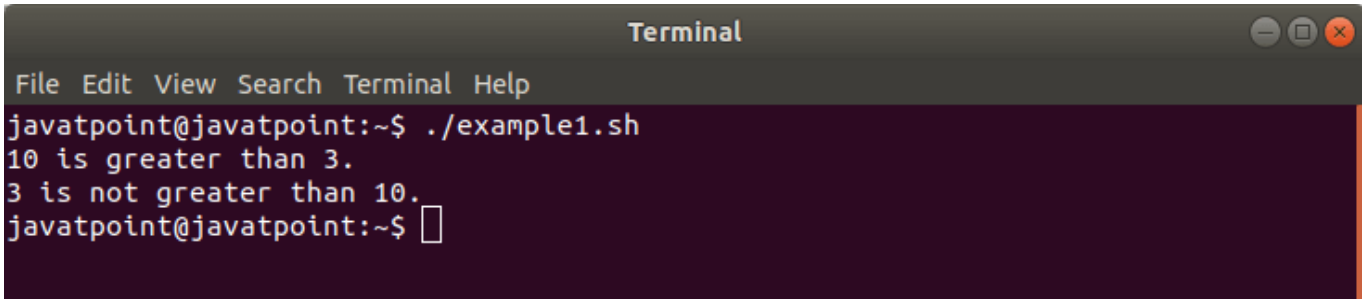
Check out the following examples demonstrating the use of the if-else statement in Bash Script:

Example 1

Following example consists of two different scenarios where in the first if-else statement, the condition is true, and in the second if-else statement, the condition is false.

```
#!/bin/bash
#when the condition is true
if [ 10 -gt 3 ];
then
  echo "10 is greater than 3."
else
  echo "10 is not greater than 3."
fi
#when the condition is false
if [ 3 -gt 10 ];
then
  echo "3 is greater than 10."
else
  echo "3 is not greater than 10."
fi
```

Output



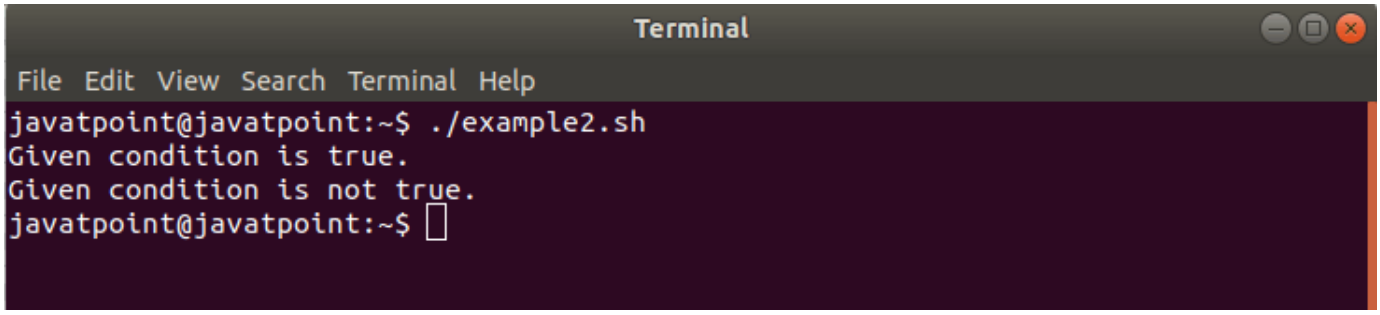
In the first if-else expression, the condition (10 -gt 3) is true and so the statement in the if block is executed. Whereas in the other if-else expression, the condition (3 -gt 10) is false and so the statement in the else block is executed.

Example 2

In this example, we explained how to use multiple conditions with the if-else statement in Bash. We use bash logical operators to join multiple conditions.

```
#!/bin/bash
# When condition is true
# TRUE && FALSE || FALSE || TRUE
if [[ 10 -gt 9 && 10 == 9 || 2 -lt 1 || 25 -gt 20 ]];
then
  echo "Given condition is true."
else
  echo "Given condition is false."
fi
# When condition is false
#TRUE && FALSE || FALSE || TRUE
if [[ 10 -gt 9 && 10 == 8 || 3 -gt 4 || 8 -gt 8 ]];
then
  echo "Given condition is true."
else
  echo "Given condition is not true."
fi
```

Output



Bash If Else Statement in a Single Line

We can write complete 'if-else statement' along with the commands in a single line. You need to follow the given rules to use if-else statement in a single line:

- Use a semi-colon (;) at the end of statements in if and else blocks.
- Use spaces as a delimiter to append all the statements.

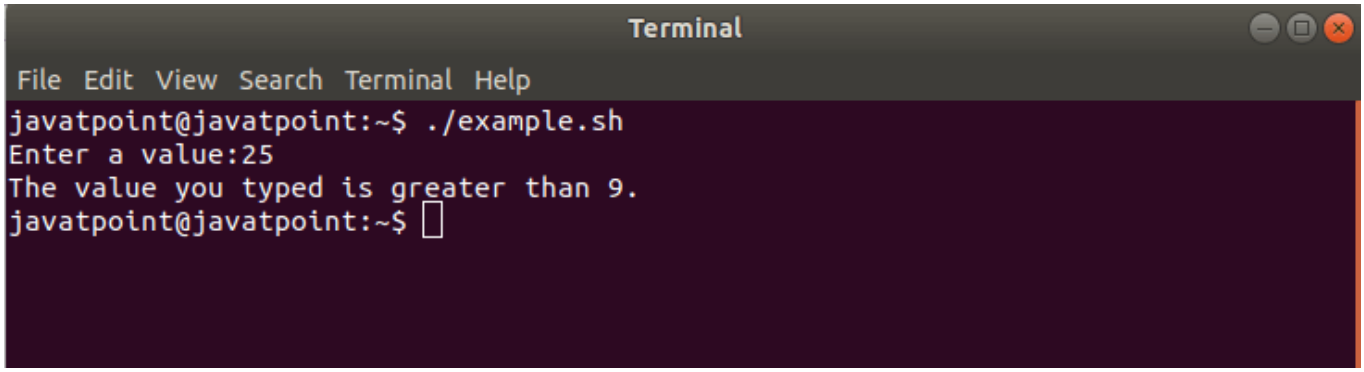
An example is given below demonstrating how to use if-else statement in a single line:

Example 3

```
#!/bin/bash
read -p "Enter a value:" value
if [ $value -gt 9 ]; then echo "The value you typed is greater than 9."; else echo "The value you typed is not greater than 9."; fi
```

Output

When we enter a value as 25, then the output will look like:



Bash Nested If Else

Just like nested if statement, the if-else statement can also be used inside another if-else statement. It is called nested if-else in Bash scripting.

Following is an example explaining how to make use of the nested if-else statement in Bash:

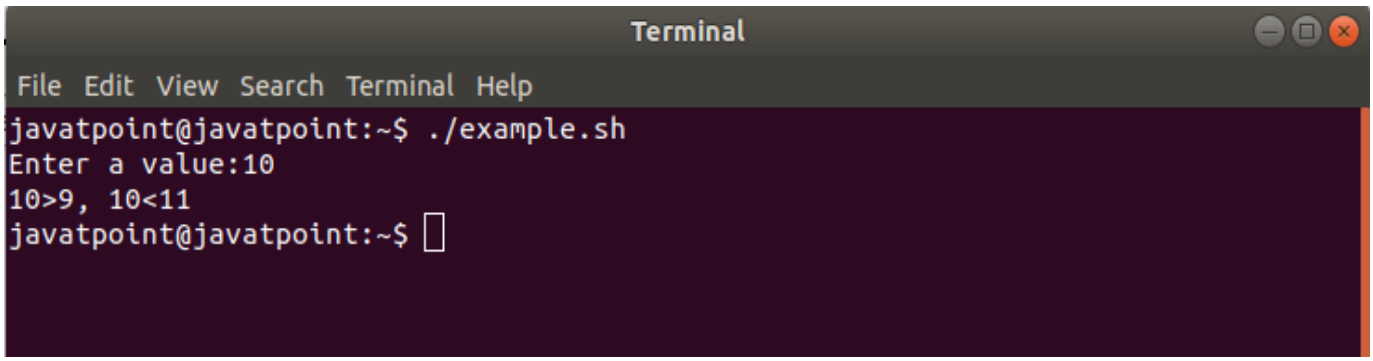
Example 4

```
#!/bin/bash
read -p "Enter a value:" value
```

```
if [ $value -gt 9 ];
then
  if [ $value -lt 11 ];
  then
    echo "$value>9, $value<11"
  else
    echo "The value you typed is greater than 9."
  fi
else echo "The value you typed is not greater than 9."
fi
```

Output

If we enter 10 as value, then the output will look like this:



Bash Else If

In this topic, we will understand how to use else-if (elif) statements in Bash scripts to get our automated tasks completed.

Bash else-if statement is used for multiple conditions. It is just like an addition to Bash if-else statement. In Bash elif, there can be several elif blocks with a boolean expression for each one of them. In the case of the first 'if statement', if a condition goes false, then the second 'if condition' is checked.

Syntax of Bash Else If (elif)

The syntax of else-if statement in Bash shell scripting can be defined as:

- if [condition];
- then
- <commands>
- elif [condition];
- then
- <commands>
- else
- <commands>
- fi

Just like if-else, we can use a set of one or more conditions joined using conditional operators. The set of commands are executed when the condition is true. If there is no true condition, then the block of commands inside the 'else statement' is executed.

Following are some examples demonstrating the usage of the else-if statement:

Example 1

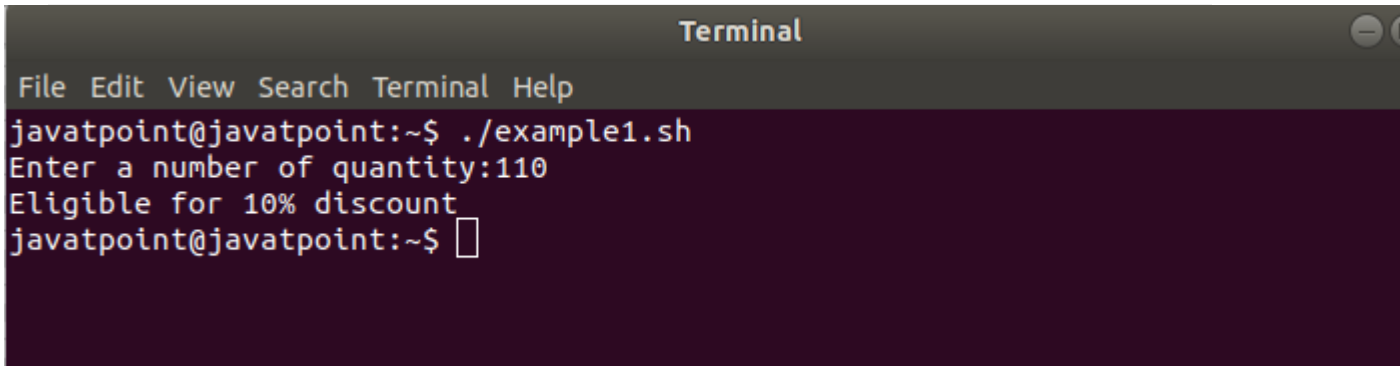
Following example consists of two different scenarios wherein the first else-if statement, the condition is true, and in the second else-if statement, the condition is false.

Bash Script

```
#!/bin/bash

read -p "Enter a number of quantity:" num

if [ $num -gt 100 ];
then
echo "Eligible for 10% discount"
elif [ $num -lt 100 ];
then
echo "Eligible for 5% discount"
else
echo "Lucky Draw Winner"
echo "Eligible to get the item for free"
fi
```

Output

- If we enter the number of quantity as 110, then the condition of 'if statement' evaluates to true and the output looks like:



- If we enter the number of quantity as 90 then condition of 'elif statement' evaluates to true, and the output looks like:

```
Terminal
File  Edit  View  Search  Terminal  Help
javatpoint@javatpoint:~$ ./example1.sh
Enter a number of quantity:90
Eligible for 5% discount
javatpoint@javatpoint:~$
```

- If we enter the number of quantity as 100, then no condition will be true. In this case, the block of commands inside the 'else statement' is executed, and the output looks like:

```
Terminal
File  Edit  View  Search  Terminal  Help
javatpoint@javatpoint:~$ ./example1.sh
Enter a number of quantity:100
Lucky Draw Winner
Eligible to get the item for free
javatpoint@javatpoint:~$
```

This is how basic bash else-if works.

Example 2

This example is demonstrating how to use multiple conditions with the else-if statement in Bash. We use bash logical operators to join multiple conditions.

Bash Script

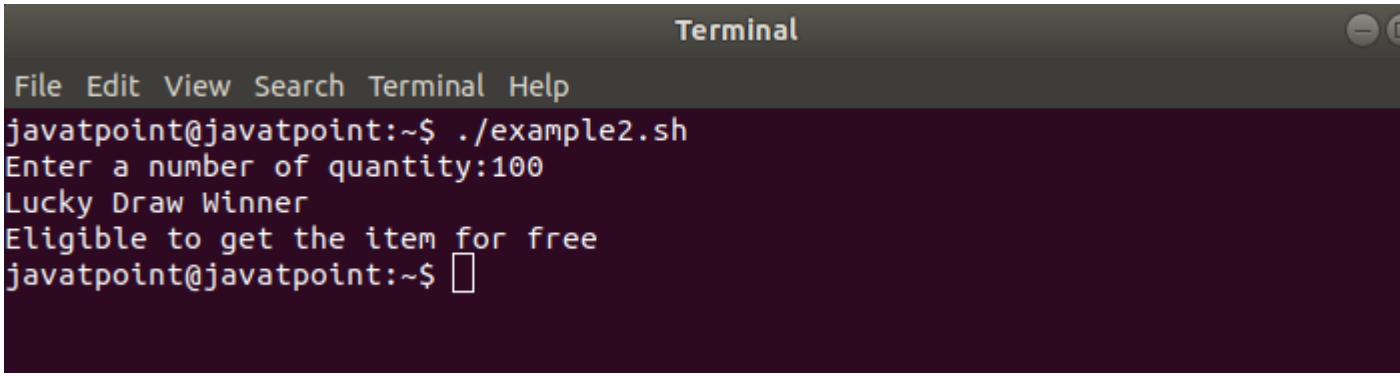
```
#!/bin/bash

read -p "Enter a number of quantity:" num

if [ $num -gt 200 ];
then
echo "Eligible for 20% discount"

elif [[ $num == 200 || $num == 100 ]];
```

```
then
echo "Lucky Draw Winner"
echo "Eligible to get the item for free"

elif [[ $num -gt 100 && $num -lt 200 ]];
then
echo "Eligible for 10% discount"

elif [ $num -lt 100 ];
then
echo "No discount"
fi
```

Note: It should be noted that else block is optional.

Output

If we enter the number of quantity as 100, then the output will look like:



Try this example by putting different values and check out the results.