Control structure

The unix support two types of control structure.

1. Decision making control structure and
2. Loop control structure.

Decision making control structure allow the computer to take the decision depending upon the condition and executes the statements or skip some statement.

The Loop control structure helps computer to execute a group of statements repeatedly for specified number of time.

1. Decision making control structure

Unix Shell supports two Decision making control structure

1. **if** statement
2. **case...esac** statement
3. **if statement**

it take decisions, depending on the fulfillment of a certain condition. There are three types of if statement.

1. [**if...fi statement**](https://www.tutorialspoint.com/unix/if-fi-statement.htm)
2. [**if...else...fi statement**](https://www.tutorialspoint.com/unix/if-else-statement.htm)
3. [**if...elif...else...fi statement**](https://www.tutorialspoint.com/unix/if-elif-statement.htm)
4. **if...fi** statement is the fundamental control statement that allows Shell to make decisions and execute statements conditionally.

## Syntax

if [ expression ]

then

Statement(s) to be executed if expression is true

fi

The *Shell expression* is evaluated in the above syntax. If the resulting value is *true*, given *statement(s)* are executed. If the *expression* is *false* then no statement would be executed. Most of the times, comparison operators are used for making decisions.

It is recommended to be careful with the spaces between braces and expression. No space produces a syntax error.

If **expression** is a shell command, then it will be assumed true if it returns **0** after execution. If it is a Boolean expression, then it would be true if it returns true.

## Example

a=10

b=20

if [ $a == $b ]

then

echo "a is equal to b"

fi

if [ $a != $b ]

then

echo "a is not equal to b"

fi

The above script will generate the following result −

a is not equal to b

The **if...else...fi** statement is the next form of control statement that allows Shell to execute statements in a controlled way and make the right choice.

## Syntax

if [ expression ]

then

Statement(s) to be executed if expression is true

else

Statement(s) to be executed if expression is not true

fi

The Shell *expression* is evaluated in the above syntax. If the resulting value is *true*, given *statement(s)* are executed. If the *expression* is *false*, then no statement will be executed.

## Example

The above example can also be written using the *if...else* statement as follows −

[Live Demo](http://tpcg.io/FqxOwQ)

#!/bin/sh

a=10

b=20

if [ $a == $b ]

then

echo "a is equal to b"

else

echo "a is not equal to b"

fi

Upon execution, you will receive the following result −

a is not equal to b

The **if...elif...fi** statement is the one level advance form of control statement that allows Shell to make correct decision out of several conditions.

## Syntax

if [ expression 1 ]

then

Statement(s) to be executed if expression 1 is true

elif [ expression 2 ]

then

Statement(s) to be executed if expression 2 is true

elif [ expression 3 ]

then

Statement(s) to be executed if expression 3 is true

else

Statement(s) to be executed if no expression is true

fi

This code is just a series of *if* statements, where each *if* is part of the *else* clause of the previous statement. Here statement(s) are executed based on the true condition, if none of the condition is true then *else* block is executed.

## Example

[Live Demo](http://tpcg.io/i5Hw31)

#!/bin/sh

a=10

b=20

if [ $a == $b ]

then

echo "a is equal to b"

elif [ $a -gt $b ]

then

echo "a is greater than b"

elif [ $a -lt $b ]

then

echo "a is less than b"

else

echo "None of the condition met"

fi

Upon execution, you will receive the following result −

a is less than b

1. Loop control structure.