BATCH :	BATCH 48
LESSON :	**BATCH SCRIPTING**
DATE :	31.12.2021
SUBJECT :	**BATCH SCRIPTING**

techproeducation

techproeducation

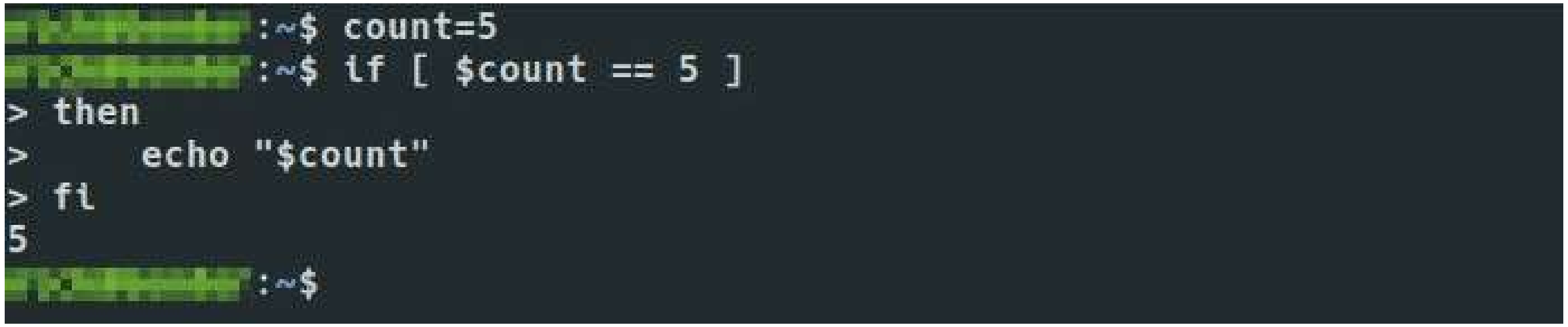
techproeducation

techproeducation

techproedu

IF STATEMENT

* A simple if statement essentially states, if a particular test is true, then perform a specified set of actions. If it's not true, don't take those acts.

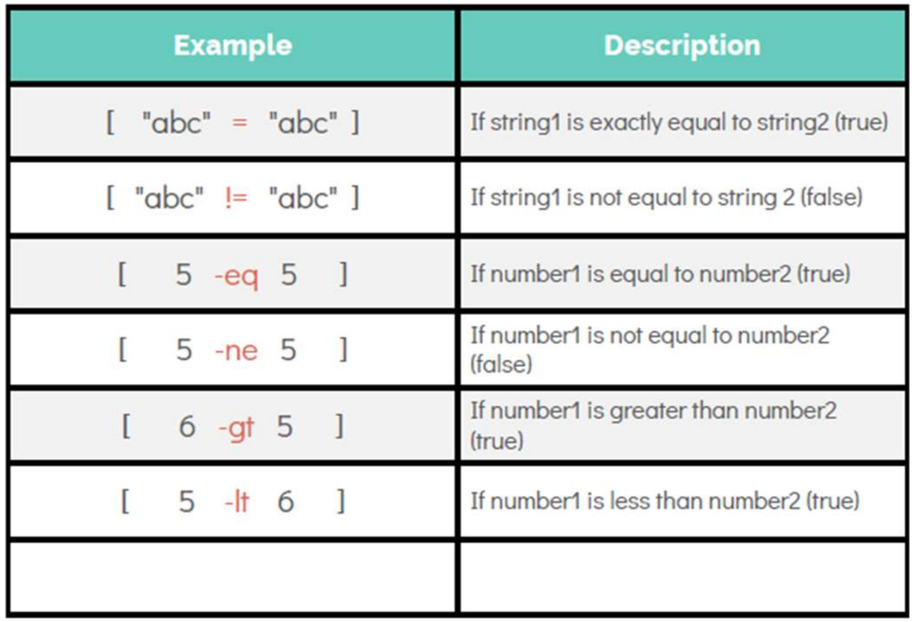


```
[redacted]:~$ count=5
[redacted]:~$ if [ $count == 5 ]
> then
>     echo "$count"
> fi
5
[redacted]:~$
```

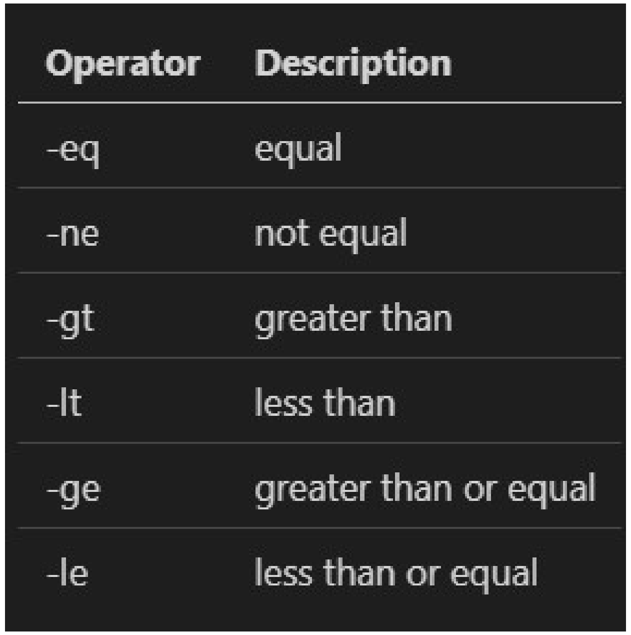
STATEMENTS

Comparing statement(s)

- Comparing statement are for comparing two variables.



Example	Description
["abc" = "abc"]	If string1 is exactly equal to string2 (true)
["abc" != "abc"]	If string1 is not equal to string 2 (false)
[5 -eq 5]	If number1 is equal to number2 (true)
[5 -ne 5]	If number1 is not equal to number2 (false)
[6 -gt 5]	If number1 is greater than number2 (true)
[5 -lt 6]	If number1 is less than number2 (true)

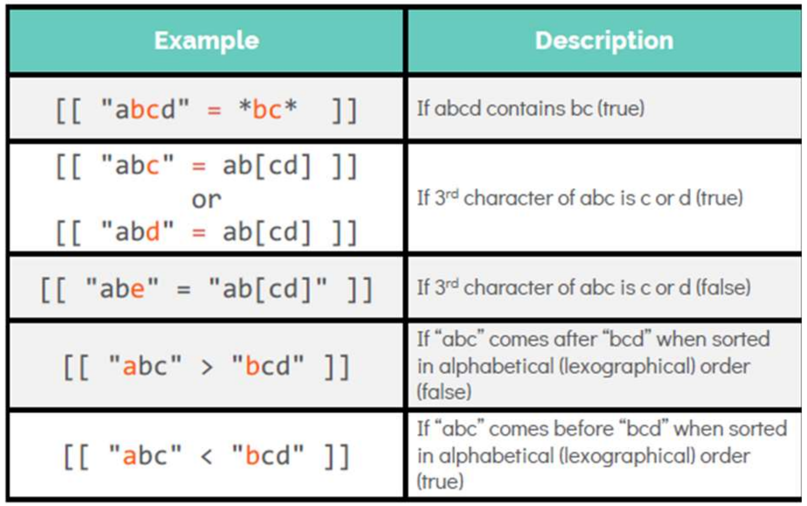


Operator	Description
-eq	equal
-ne	not equal
-gt	greater than
-lt	less than
-ge	greater than or equal
-le	less than or equal

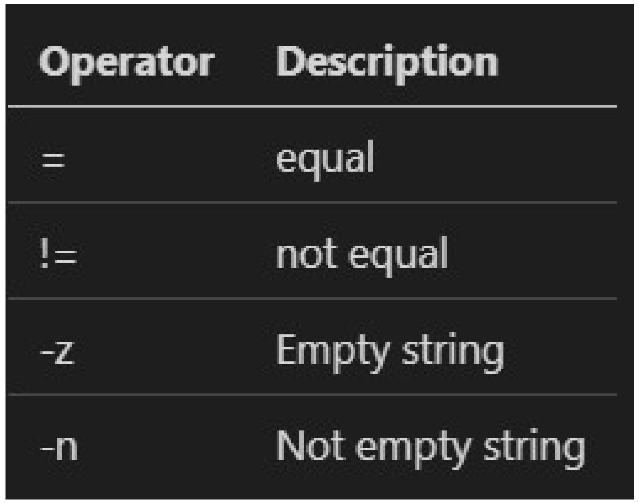
STATEMENTS

STRING OPERATORS

- String operations are for making operations with strings broadly.



Example	Description
`[[ "abcd" = *bc* ]]`	If abcd contains bc (true)
`[[ "abc" = ab[cd] ]]` or `[[ "abd" = ab[cd] ]]`	If 3rd character of abc is c or d (true)
`[[ "abe" = "ab[cd]" ]]`	If 3rd character of abc is c or d (false)
`[[ "abc" > "bcd" ]]`	If "abc" comes after "bcd" when sorted in alphabetical (lexographical) order (false)
`[[ "abc" < "bcd" ]]`	If "abc" comes before "bcd" when sorted in alphabetical (lexographical) order (true)

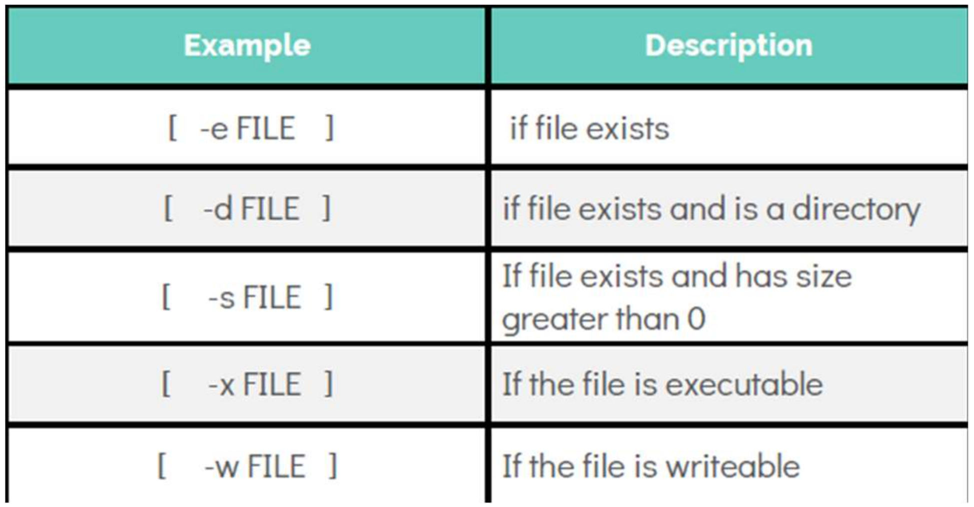


Operator	Description
=	equal
!=	not equal
-z	Empty string
-n	Not empty string

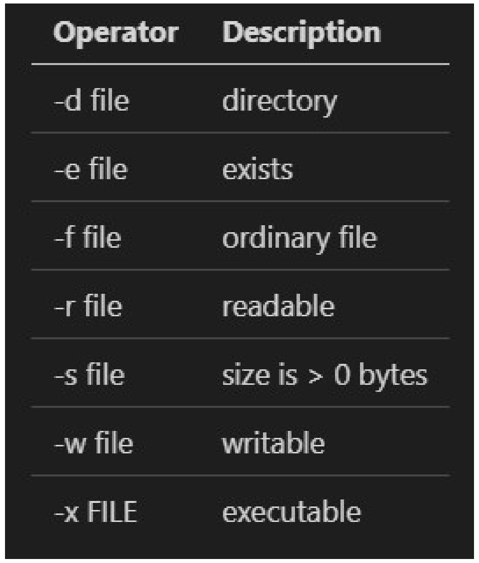
STATEMENTS

File Test Operators

- There are a few operators that can be used to test various properties associated with a Linux file.



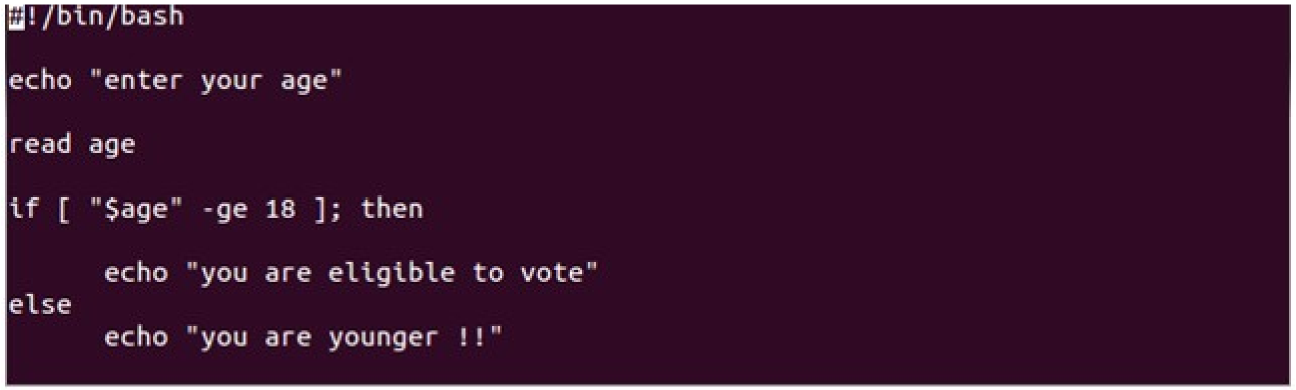
Example	Description
[-e FILE]	if file exists
[-d FILE]	if file exists and is a directory
[-s FILE]	If file exists and has size greater than 0
[-x FILE]	If the file is executable
[-w FILE]	If the file is writeable



Operator	Description
-d file	directory
-e file	exists
-f file	ordinary file
-r file	readable
-s file	size is > 0 bytes
-w file	writable
-x FILE	executable

IF-ELSE STATEMENT

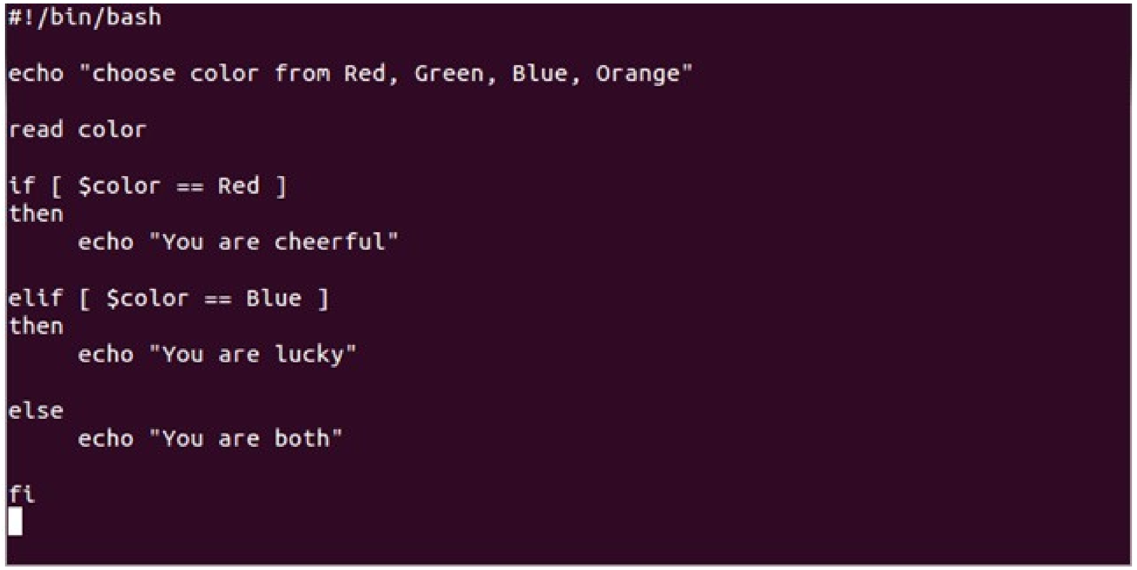
- Sometimes we want to execute a block of code if a statement is true, and another block of code if it is false. In that case, we use if-else statements.



```
#!/bin/bash

echo "enter your age"

read age

if [ "$age" -ge 18 ]; then

      echo "you are eligible to vote"
else
      echo "you are younger !!"
```

ELIF STATEMENT

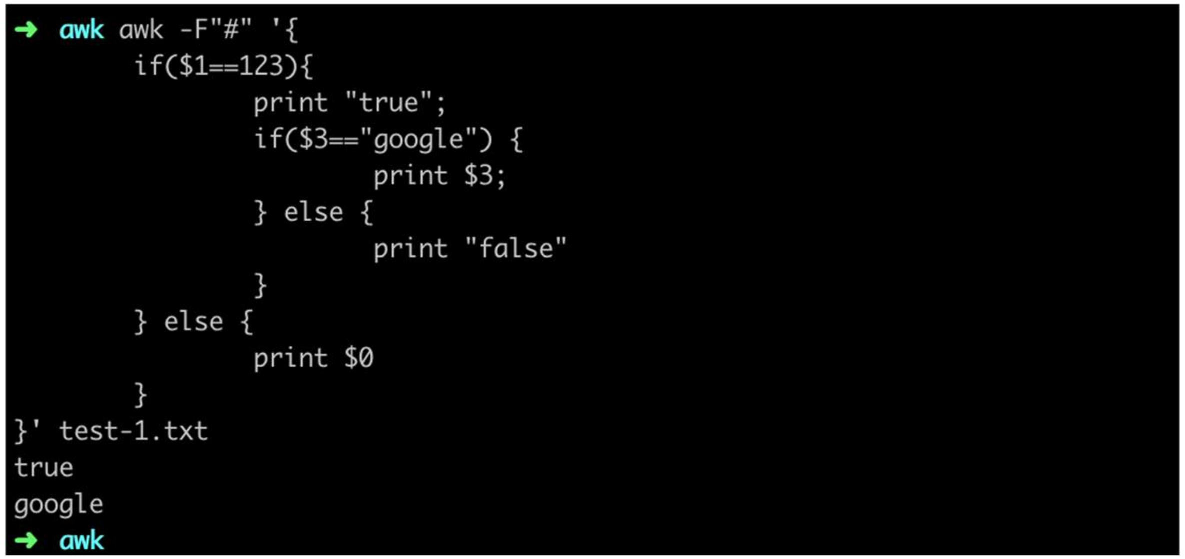
- The elif statement is used when it requires to specify several conditions in our program.



```
#!/bin/bash

echo "choose color from Red, Green, Blue, Orange"

read color

if [ $color == Red ]
then
     echo "You are cheerful"

elif [ $color == Blue ]
then
     echo "You are lucky"

else
     echo "You are both"

fi
```

NESTED IF STATEMENT

- If statements can be nested.



```
➜  awk awk -F"#" '{
        if($1==123){
                print "true";
                if($3=="google") {
                        print $3;
                } else {
                        print "false"
                }
        } else {
                print $0
        }
}' test-1.txt
true
google
➜  awk
```

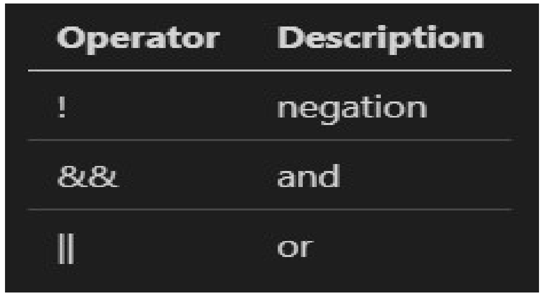
• ! inverts a true condition into false and vice versa.

• ! inverts a true condition into false and vice versa.

• ! inverts a true condition into false and vice versa.

BOOLEAN OPERATIONS

- The Boolean operators below are supported by the Bourne Shell.

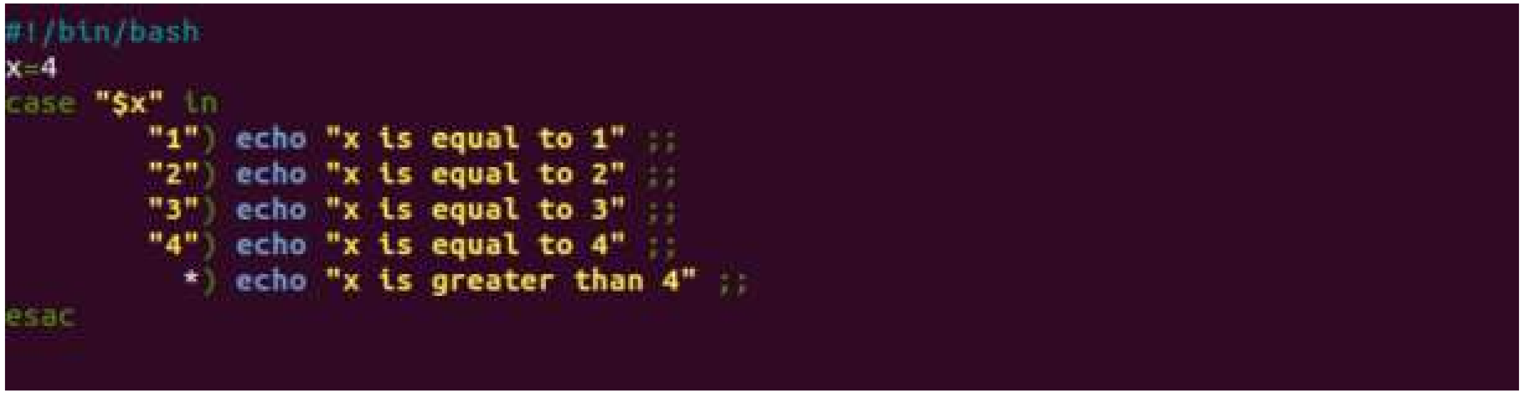


Operator	Description
!	negation
&&	and
||	or

- ! inverts a true condition into false and vice versa.
- && is logical AND. If both the operands are true, then the condition becomes true otherwise false.
- || is logical OR. If one of the operands is true, then the condition becomes true.

CASE STATEMENT

* To execute a multiway branch, we can use several if-elif statements but that would soon become complicated. Bash case statements are similar to if-else statements but are easier and simpler. It helps to match one variable against several values.



```
#!/bin/bash
x=4
case "$x" in
        "1") echo "x is equal to 1" ;;
        "2") echo "x is equal to 2" ;;
        "3") echo "x is equal to 3" ;;
        "4") echo "x is equal to 4" ;;
          *) echo "x is greater than 4" ;;
esac
```