Shell Scripting

- Source code vs machine code
  - Source code is the human readable code written in a programing language like python
  - Machine code consists of binary 1s and 0s and is the language a computers CPU understands
- Compiler: a program used for converting a complete source code into machine code
- Interpreter: a program that converts source code into machine code line by line
- Compiled program vs Interpreted program
  - a compiled is a binary file produced by a compiler
  - an interpreted program is a program that requires an interpreter to interpret and run each instruction
- Shell scripts are examples of interpreted programs, and the interpreter used is the BASH shell

Creating a basic script

- Start vim, enable line numbers, and enter insert mode
- type:
  - #!/bin/bash
  - echo "this is a script that displays information about your linux system"
  - uname -a
- Save the file and name it script1.sh
- Type chmod u+x script1.sh to make the file executable
- to run the script type : ./script1.sh

Displaying text

- To display a line of text use the echo command
  - Ex. echo "this is a message"
- echo -n does not output a new line

Working with variables

- Variable: placeholder for data
- Environment variable: is the placeholder for data that can change: typically, it gets its value automatically from the OS startup or the shell being used.
- The HOME environment variable stores the absolute pathname to a user's home directory, so it varies for each user
- The env command allows you to see all environment variables
- You can also used the echo command to see the calue of each environment variable
  - echo $HOME
  - echo $HOST

Exit Status Codes

- Exit Status code: a number sent to the shell when you run a command
- Successful commands usually return the code 0, and failures return a value greater than 0

Using Structured Commands

Conditions

- The if statement is used to carry out certain commands based on testing a condition and the exit status of the command
- if-statement- Starts the condition being tested
- then statement- Starts the portion of code specifying what to do if the condition evaluates to true
- else statement- Starts the portion of code specifying what to do if the condition evaluates to false.
- The fi statement indicates the end of the if statement

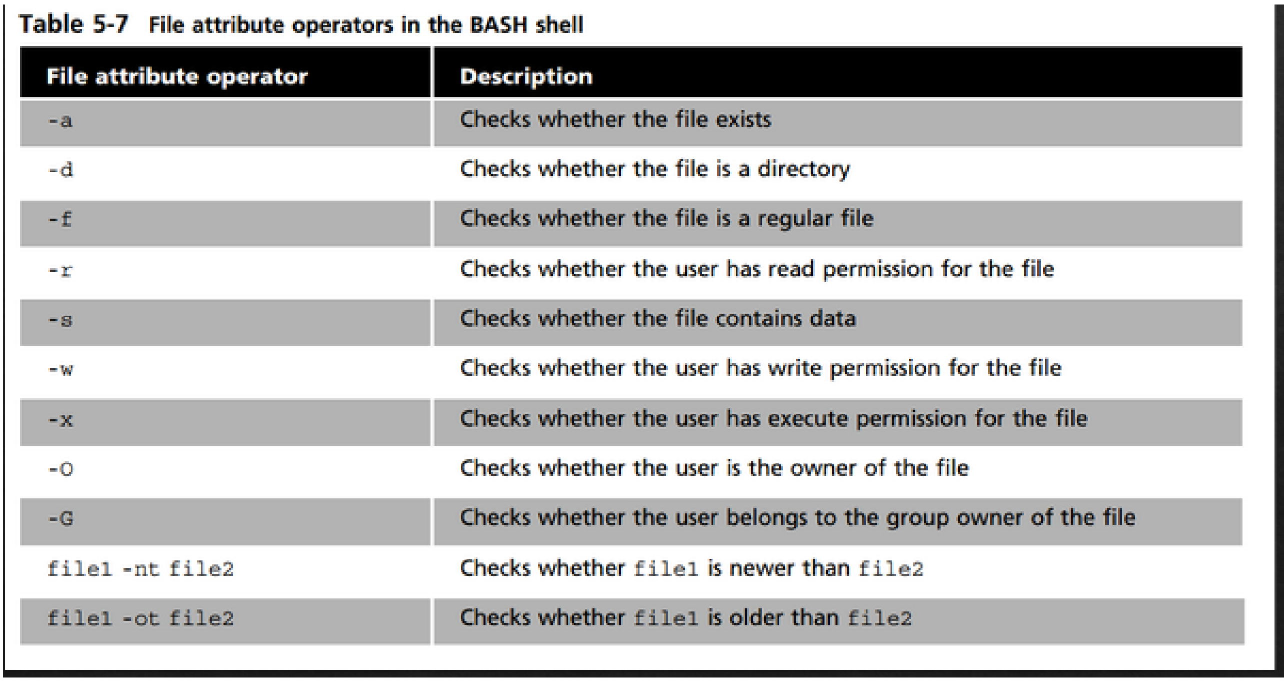
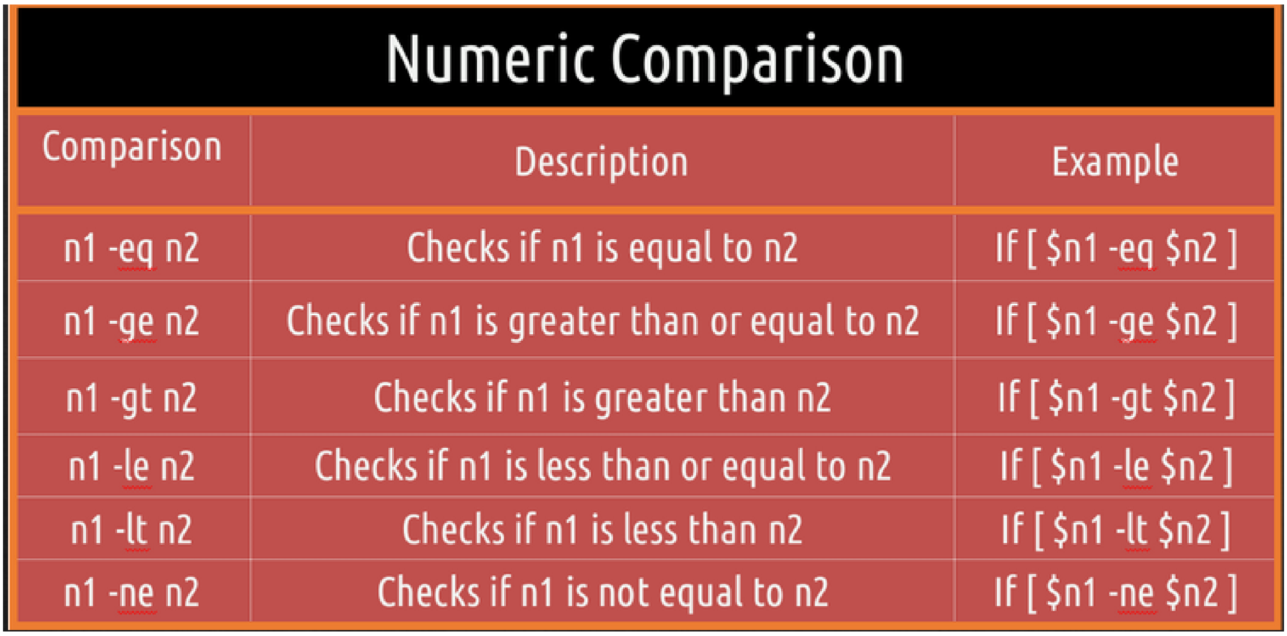


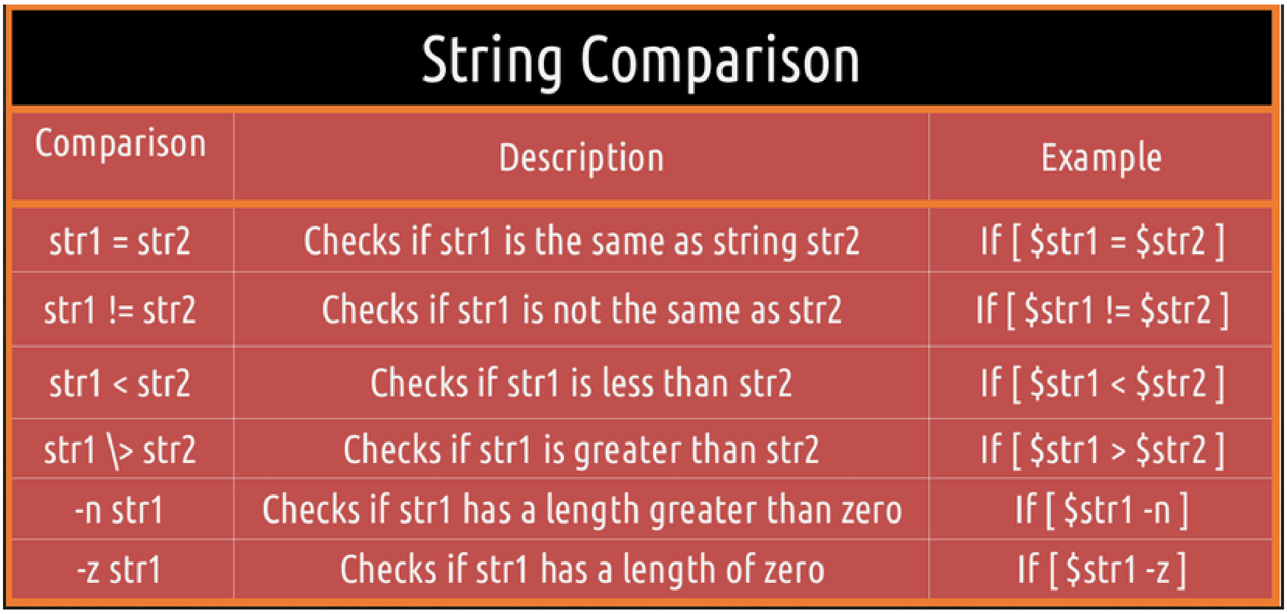
Table 5-7 File attribute operators in the BASH shell

File attribute operator	Description
`-a`	Checks whether the file exists
`-d`	Checks whether the file is a directory
`-f`	Checks whether the file is a regular file
`-r`	Checks whether the user has read permission for the file
`-s`	Checks whether the file contains data
`-w`	Checks whether the user has write permission for the file
`-x`	Checks whether the user has execute permission for the file
`-O`	Checks whether the user is the owner of the file
`-G`	Checks whether the user belongs to the group owner of the file
`file1 -nt file2`	Checks whether `file1` is newer than `file2`
`file1 -ot file2`	Checks whether `file1` is older than `file2`



Numeric Comparison

Comparison	Description	Example
n1 -eq n2	Checks if n1 is equal to n2	If [$n1 -eq $n2]
n1 -ge n2	Checks if n1 is greater than or equal to n2	If [$n1 -ge $n2]
n1 -gt n2	Checks if n1 is greater than n2	If [$n1 -gt $n2]
n1 -le n2	Checks if n1 is less than or equal to n2	If [$n1 -le $n2]
n1 -lt n2	Checks if n1 is less than n2	If [$n1 -lt $n2]
n1 -ne n2	Checks if n1 is not equal to n2	If [$n1 -ne $n2]



String Comparison

Comparison	Description	Example
str1 = str2	Checks if str1 is the same as string str2	If [$str1 = $str2]
str1 != str2	Checks if str1 is not the same as str2	If [$str1 != $str2]
str1 < str2	Checks if str1 is less than str2	If [$str1 < $str2]
str1 \> str2	Checks if str1 is greater than str2	If [$str1 > $str2]
-n str1	Checks if str1 has a length greater than zero	If [$str1 -n]
-z str1	Checks if str1 has a length of zero	If [$str1 -z]