CHMOD COMMAND

OPERATING SYSTEMS LABS

ASSIGNMENT # 01

Submitted by

MUHAMMAD USMAN

(19P-0116)

Submitted to

MR. MUHAMMAD ABDULLAH

(INSTRUCTOR CS)

DEPARTMENT OF COMPUTER SCIENCE

FAST NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCINCES, PESHAWAR

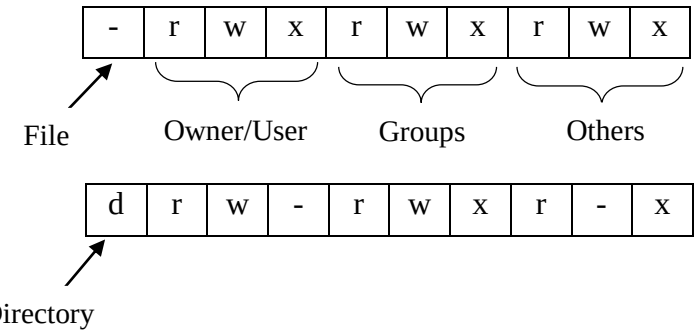
Session 2019-2023

CHMOD Command:

Chmod command in Linux has an important role as it can give some file/directory permissions such as read, write, and execute. There are ten-characters that show the type and permissions that a particular file or directory has. These permissions are defined as follow:

Modes	Description
-	No permission given.
r	Permission to read from file or directory.
w	Permission to write on file or directory.
x	Permission to execute file or directory.

The following diagrams show how these characters look and what they represent:



There are two different ways to set these permissions of a file or directory and they are the following:

1. **Absolute Mode:**
   Absolute mode is a method that sets permissions in a numeric way and the following table show how to set permissions for user, groups, and others:

Number	Description	Symbols
0	No permission	- - -
1	Execute	- - x
2	Write	- w -
3	Write + Execute	- w r
4	Read	r - -
5	Read + Execute	r - x
6	Read + Write	r w -
7	Read + Write + Execute	r w x

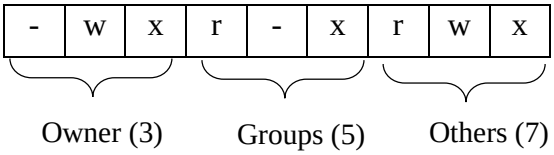
Syntax:

```
chmod [numeric value for U G O] [file name]
```

Examples:

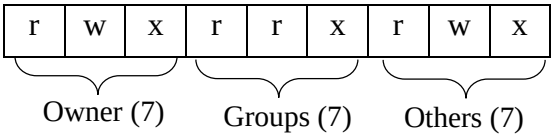
i. `Chmod 357 file1.txt`

In the above example the permissions will be set:



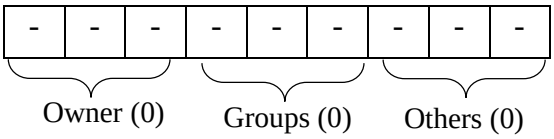
ii. `chmod 777 file1.txt`

Now, for this example the following permission will be set:



iii. `chmod 000 file1.txt`

This command will remove all the given permissions to all the classes.



2. Symbolic Mode:

In symbolic mode, the user can add/remove permissions from one or more classes (owner, groups, and others). The following tables show the symbols that are used in Symbolic mode:

Reference	Class	Description
u	Owner/User	Owner/Users files
g	Group	Group files
o	Others	Other files
a	All	Appling permissions on all the above

Operator	Description
+	Add permission/s
–	Removes permission/s
=	Only add specific permission/s to the class

Syntax:

```
chmod [reference/s] [operator] [permission/s] [file name]
```

Examples:

i. `chmod u + r file2.txt`

The above example will allow the owner/user to read file/directory.

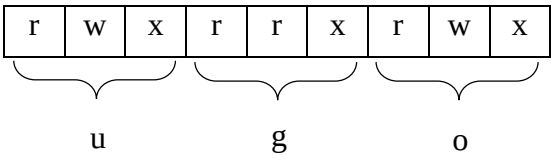
ii. `chmod g + rwx file2.txt`

In this example, group class will get all the permissions (read, write, and execute).

iii. `chmod u = r file2.txt`

The above example will add the permission to read but remove or nullify the other permissions given to the user class.

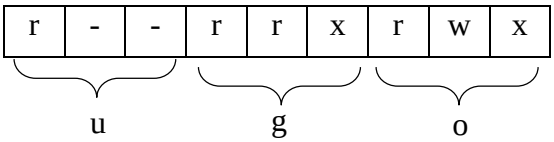
Before:



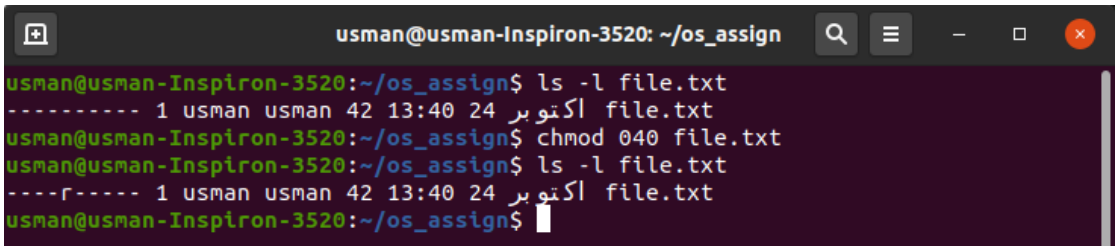
Command:

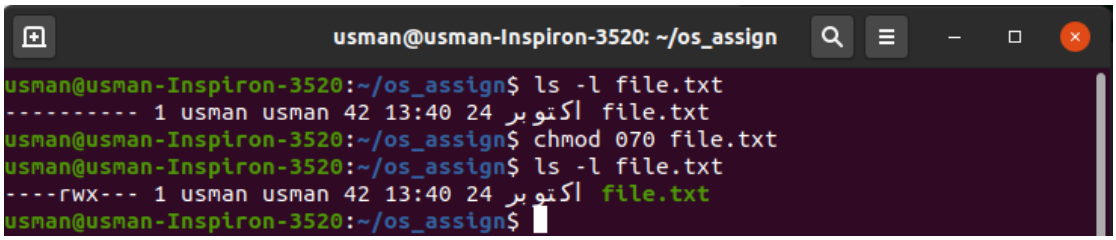
```
chmod u = r file2.txt
```

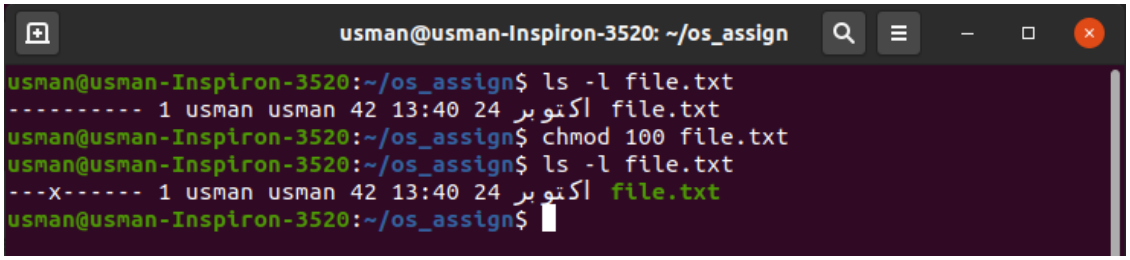
After:

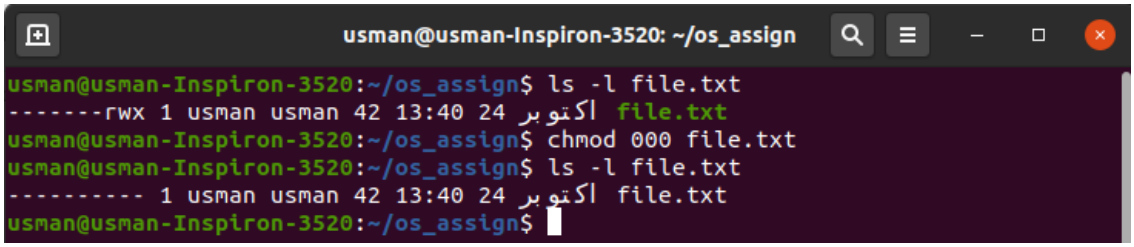


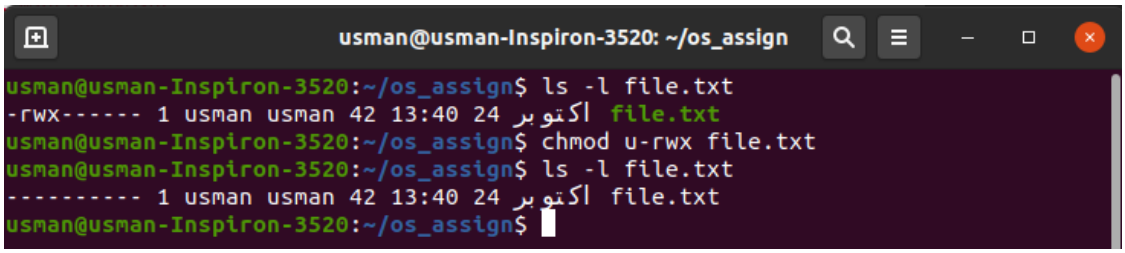
Some Screenshots of both modes:











Options:

Change the mode of each FILE to MODE. With --reference, change the mode of each FILE to that of RFILE.

-c, --changes

like verbose but report only when a change is made

-f, --silent, --quiet

suppress most error messages

-v, --verbose

output a diagnostic for every file processed

--no-preserve-root

do not treat '/' specially (the default)

--preserve-root

fail to operate recursively on '/'

--reference=RFILE

use RFILE's mode instead of MODE values

-R, --recursive

change files and directories recursively

--help

display this help and exit

--version

output version information and exit