Topic Name:

The main aim of this lab session is to provide hands-on experience on

- Explore file structure
- File management commands
- Absolute path and Relative path
- Globbing
- Scripting

File Structure

1. Under the root directory there are many files like
   /bin , /boot , /dev , /etc , ...
   Find out the importance of those files
   Example : /etc is for user account details

S.No	Directory	Usage
1	/	Root directory
2	/bin	Binary files
3	/boot	It contains all the boot-related information files and folders such as conf, grub, etc
4	/dev	It is the location of the device files such as dev/sda1, dev/sda2 etc
5	/etc	System configuration files
6	/home	Home directory. It is the default current directory.
7	/lib	It contains kernel modules and a shared library.
8	/proc	It is a virtual and pseudo file system to contain information about the running processes with a specific process ID or PID.
9	/sbin	Binary executable programs for an administrator.
10	/tmp	Temporary space typically cleared on reboot.
11	/var	Log files
12	/opt	Optional or third-party software
13	/Usr	User related programs.

2. In Linux, there are three different files
   Regular file
   Directory
   Special file
   Block file
   Character file
   Socket file
   Pipe file

Fill the below table:

File Type	Represented by (Hint ls)	Role	How to create	How to check	Location	Screen shot
Regular file	-	Stores data such as text,images,etc		NA		
- Text file	-	Contains plain text		NA		
- Compressed file	-	Stores data in a compressed format		NA		
- Image	-	Stores image data	NA	NA		
Directory	d			NA		
Block file	b		NA	NA		
Character file	c		NA	NA		
Socket file	s		NA	NA		
pipe file	p		NA	NA		

3. Globbing

a. Go back to CYS

b. Create multiple subdirectories using single command

LS

Unit1

command

glob

Unit2

command

grep

Unit3

constructs

```
┌──(ceyona㉿kali)-[~/cys]
└─$ mkdir -p cys cys/LS/Unit1/{Command,glob} cys/LS/Unit2/{command,grep} cys/LS/Unit3/Constructors

┌──(ceyona㉿kali)-[~/cys]
└─$ tree cys
cys
└── LS
    ├── Unit1
    │   ├── Command
    │   └── glob
    ├── Unit2
    │   ├── command
    │   └── grep
    └── Unit3
        └── Constructors

10 directories, 0 files
```

c. Navigate to unit1/glob

```
┌──(ceyona㉿kali)-[~/cys]
└─$ cd Unit1/glob
```

d. Create the following files :

Commands.txt
Commands1.txt
Commands2.txt
page1.html
page2.html
page3.html
file1
file10
file11
file2
File2
File3
file33
fileAB
filea
fileA
fileAAA
file(
file 2

```
┌──(ceyona㉿kali)-[~/cys]
└─$ touch Commands.txt Commands1.txt Commands2.txt page1.html page2.html page3.html file1 file10 file11 file2 File2 File3 file33 fileAB filea fileA fileAAA file\( file\ 2
```

i. List all files starting with file

```
┌──(ceyona㉿kali)-[~/cys]
└─$ ls file*
'file 2'  'file('   file1   file10   file11   file2   file33   fileA   fileAAA   fileAB   filea
```

ii. List all files starting with File

```
┌──(ceyona㉿kali)-[~/cys]
└─$ ls File*
File2  File3
```

iii. List all files starting with file and ending in a number.

```
┌──(ceyona㉿kali)-[~/cys]
└─$ ls file*[0-9]
'file 2'   file1   file10   file11   file2   file33
```

iv. List all files starting with file and ending with a letter

```
┌──(ceyona㉿kali)-[~/cys]
└─$ ls file*[a-zA-Z]
fileA  fileAAA  fileAB  filea
```

v. List all files starting with File and having a digit as fifth character.

```
┌──(ceyona㉿kali)-[~/cys]
└─$ ls File?
File2  File3
```

vi. List all files starting with File and having a digit as fifth character and nothing else.

```
┌──(ceyona㉿kali)-[~/cys]
└─$ ls File?
File2  File3
```

vii. List (with ls) all files starting with a letter and ending in a number.

```
┌──(ceyona㉿kali)-[~/cys]
└─$ ls [a-zA-Z]*[0-9]
 File2   File3  'file 2'   file1   file10   file11   file2   file33
```

viii. List (with ls) all files that have exactly five characters.

```
┌──(ceyona㉿kali)-[~/cys]
└─$ ls ?????
 File2   File3  'file('   file1   file2   fileA   filea
```

ix. List (with ls) all files that start with f or F and end with 3 or A.

```
┌──(ceyona㉿kali)-[~/cys]
└─$ ls [fF]*[3A]
File3  file33  fileA  fileAAA
```

x. List (with ls) all files that start with f have i or R as second character and end in a number.

```
┌──(ceyona㉿kali)-[~/work]
└─$ ls | grep '^f'[ri]
file1
file1.txt
file2
file2.txt
```

xi. List all files that do not start with the letter F.

```
┌──(ceyona㉿kali)-[~/cys]
└─$ ls -l[!F]*
zsh: event not found: F]
```

xii. Remove all the *.html

```
┌──(ceyona㉿kali)-[~/cys]
└─$ rm *.html
```

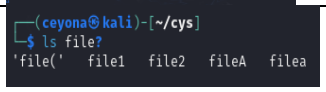
xiii. Rename *.txt to *.json

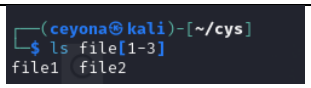
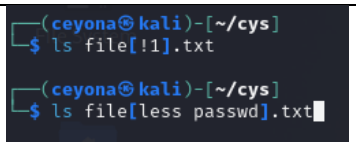
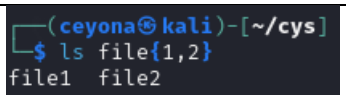
```
┌──(ceyona㉿kali)-[~/cys]
└─$ for file in *.txt; do mv "$file" "${file%.txt}.json"; done
```

4. Absolute path and relative path

Use rm, mv, cp, ls with absolute path and relative path as per your choice.

5. Wildcards

Notation	Use	Example	Screenshot
*	One or many	ls *	
?	Match only one character	ls file?	

[ ]	Used to match single character from a set of specified characters	ls file[1-3]	
[!]	Matches any character that is not a member of the set characters	ls file[!1].txt	
{ }	Used to generate multiple arguments by separating the values with commas	ls file{1,2}	

More on Character class

Notation	Use	Example	Screenshot
[:alnum:]	Matches any alphanumeric character	ls *[[:alnum:]].txt	
[:alpha:]	Matches any alphabetic character	ls *[[:alpha:]].txt	
[:digit:]	Matches any numeric digit (0-9).	ls *[[:digit:]].txt	
[:lower:]	Matches any lowercase alphabetic character	ls *[[:lower:]].txt	
[:upper:]	Matches any uppercase alphabetic character (A-Z)	ls *[[:upper:]].txt	

4. change permission

a) Change the permission set of /work/readme.txt so that only the user (owner) can read,write, and execute it. Use absolute mode.

```
┌──(ceyona㉿kali)-[~/work]
└─$ chmod 700 readme.txt

┌──(ceyona㉿kali)-[~/work]
└─$ ls -l
total 0
-rw-rw-r-- 1 ceyona ceyona 0 Aug 23 19:11 example.txt
-rw-rw-r-- 1 ceyona ceyona 0 Aug 23 19:11 file1
-rw-rw-r-- 1 ceyona ceyona 0 Aug 23 19:11 file1.txt
-rw-rw-r-- 1 ceyona ceyona 0 Aug 23 19:11 file2
-rw-rw-r-- 1 ceyona ceyona 0 Aug 23 19:11 file2.txt
-rwx------ 1 ceyona ceyona 0 Aug 23 19:11 readme.txt
```

b) Change the permission set of /work/readme.txt so that any user can read it, the group can read/write to it and the user (owner) can read/write/execute it. Use absolute mode.

```
┌──(ceyona㉿kali)-[~/work]
└─$ ls -l
total 0
-rw-rw-r-- 1 ceyona ceyona 0 Aug 23 19:11 example.txt
-rw-rw-r-- 1 ceyona ceyona 0 Aug 23 19:11 file1
-rw-rw-r-- 1 ceyona ceyona 0 Aug 23 19:11 file1.txt
-rw-rw-r-- 1 ceyona ceyona 0 Aug 23 19:11 file2
-rw-rw-r-- 1 ceyona ceyona 0 Aug 23 19:11 file2.txt
-rwxrw-r-- 1 ceyona ceyona 0 Aug 23 19:11 readme.txt
```

c) Change the permission set of /bin/bash so that only the user (owner) can read/write/ execute, group, and any user can execute it. However, whenever anyone executes it, it should run with the privileges of the owner user. Use absolute mode.

```
┌──(ceyona㉿kali)-[~/work]
└─$ chmod 711 /bin/bash
chmod: changing permissions of '/bin/bash': Operation not permitted
```

d) Change the permission set of /work/readme.txt so that only the user (owner) can read, write, and execute it. Use relative mode.

```
┌──(ceyona㉿kali)-[~/work]
└─$ chmod u+rwx,go-rwx readme.txt

┌──(ceyona㉿kali)-[~/work]
└─$ ls -l
total 0
-rw-rw-r-- 1 ceyona ceyona 0 Aug 23 19:11 example.txt
-rw-rw-r-- 1 ceyona ceyona 0 Aug 23 19:11 file1
-rw-rw-r-- 1 ceyona ceyona 0 Aug 23 19:11 file1.txt
-rw-rw-r-- 1 ceyona ceyona 0 Aug 23 19:11 file2
-rw-rw-r-- 1 ceyona ceyona 0 Aug 23 19:11 file2.txt
-rwx------ 1 ceyona ceyona 0 Aug 23 19:11 readme.txt
```

e) Change the permission set of /work/readme.txt so that any user can read it, the group can read/write to it and the user (owner) can read/write/execute it. Use relative mode.

```
┌──(ceyona㉿kali)-[~/work]
└─$ chmod u+rwx,go=rw,o=rw readme.txt

┌──(ceyona㉿kali)-[~/work]
└─$ ls -l
total 0
-rw-rw-r-- 1 ceyona ceyona 0 Aug 23 19:11 example.txt
-rw-rw-r-- 1 ceyona ceyona 0 Aug 23 19:11 file1
-rw-rw-r-- 1 ceyona ceyona 0 Aug 23 19:11 file1.txt
-rw-rw-r-- 1 ceyona ceyona 0 Aug 23 19:11 file2
-rw-rw-r-- 1 ceyona ceyona 0 Aug 23 19:11 file2.txt
-rwxrw-rw- 1 ceyona ceyona 0 Aug 23 19:11 readme.txt
```

f) Change the permission set of /work/readme.txt so that only the user (owner) can read/write/ execute, group, and any user can execute it. However, whenever anyone executes it, it should run with the privileges of the group. Use absolute mode.

```
┌──(ceyona㉿kali)-[~/work]
└─$ chmod 711 readme.txt

┌──(ceyona㉿kali)-[~/work]
└─$ ls-l
ls-l: command not found

┌──(ceyona㉿kali)-[~/work]
└─$ ls -l
total 0
-rw-rw-r-- 1 ceyona ceyona 0 Aug 23 19:11 example.txt
-rw-rw-r-- 1 ceyona ceyona 0 Aug 23 19:11 file1
-rw-rw-r-- 1 ceyona ceyona 0 Aug 23 19:11 file1.txt
-rw-rw-r-- 1 ceyona ceyona 0 Aug 23 19:11 file2
-rw-rw-r-- 1 ceyona ceyona 0 Aug 23 19:11 file2.txt
-rwx--x--x 1 ceyona ceyona 0 Aug 23 19:11 readme.txt
```

g) Change the permission set of /work/readme.txt so that only the owner can rename or delete this file while maintaining the existing permissions. Use absolute mode.

```
┌──(ceyona㉿kali)-[~/work]
└─$ sudo chattr +i readme.txt
[sudo] password for ceyona:

┌──(ceyona㉿kali)-[~/work]
└─$

┌──(ceyona㉿kali)-[~/work]
└─$ lasttr readme.txt
Command 'lasttr' not found, did you mean:
  command 'lsattr' from deb e2fsprogs
Try: sudo apt install <deb name>

┌──(ceyona㉿kali)-[~/work]
└─$ lsattr readme.txt
----i---------e------- readme.txt
```

h) What are the default permissions for the new file?

```
┌──(ceyona㉿kali)-[~/work]
└─$ touch newfile.txt

┌──(ceyona㉿kali)-[~/work]
└─$ ls -l newfile.txt
-rw-rw-r-- 1 ceyona ceyona 0 Aug 23 21:06 newfile.txt
```

i) What was the command to view the file permissions?

```
┌──(ceyona㉿kali)-[~/work]
└─$ ls -l
total 0
```

j) Change chmod.exercises permissions to -r--r--r—

```
┌──(ceyona㉿kali)-[~/work]
└─$ ls -l
total 0
-rw-rw-r-- 1 ceyona ceyona 0 Aug 23 19:11 example.txt
-rw-rw-r-- 1 ceyona ceyona 0 Aug 23 19:11 file1
-rw-rw-r-- 1 ceyona ceyona 0 Aug 23 19:11 file1.txt
-rw-rw-r-- 1 ceyona ceyona 0 Aug 23 19:11 file2
-rw-rw-r-- 1 ceyona ceyona 0 Aug 23 19:11 file2.txt
-rw-rw-r-- 1 ceyona ceyona 0 Aug 23 21:06 newfile.txt
-rwx--x--x 1 ceyona ceyona 0 Aug 23 19:11 readme.txt

┌──(ceyona㉿kali)-[~/work]
└─$ touch chmod.exercises

┌──(ceyona㉿kali)-[~/work]
└─$ chmod 444 chmod.exercises

┌──(ceyona㉿kali)-[~/work]
└─$ ls -l chmod.exercises
-r--r--r-- 1 ceyona ceyona 0 Aug 23 21:11 chmod.exercises
```

k) Change the file permissions to Read only for the owner, group and all other users.

```
┌──(ceyona㉿kali)-[~/work]
└─$ chmod 444 example.txt

┌──(ceyona㉿kali)-[~/work]
└─$ ls -l example.txt
-r--r--r-- 1 ceyona ceyona 0 Aug 23 19:11 example.txt
```

l) What was the command for changing the file permissions to -r--r--r--?

```
┌──(ceyona㉿kali)-[~/work]
└─$ chmod 444 example.txt

┌──(ceyona㉿kali)-[~/work]
└─$ ls -l example.txt
-r--r--r-- 1 ceyona ceyona 0 Aug 23 19:11 example.txt
```

m) Change chmod.exercises permissions to -rw-r-----

```
┌──(ceyona㉿kali)-[~/work]
└─$ chmod 640 chmod.exercises

┌──(ceyona㉿kali)-[~/work]
└─$ ls -l chmod.exercises
-rw-r----- 1 ceyona ceyona 0 Aug 23 21:11 chmod.exercises
```

n) Change the file permissions to match the following:

a. owner: Read and Write
b. group: Read
c. other: no permissions (None)

```
┌──(ceyona㉿kali)-[~/work]
└─$ chmod 640 example.txt

┌──(ceyona㉿kali)-[~/work]
└─$ ls -l example.txt
-rw-r----- 1 ceyona ceyona 0 Aug 23 19:11 example.txt
```

o) What was the command for changing the file permissions to -rw-r-----?

```
┌──(ceyona㉿kali)-[~/work]
└─$ chmod 640 example.txt

┌──(ceyona㉿kali)-[~/work]
└─$ ls -l example.txt
-rw-r----- 1 ceyona ceyona 0 Aug 23 19:11 example.txt
```

p) Change chmod.exercises permissions to -rwxr-x—x

```
┌──(ceyona㉿kali)-[~/work]
└─$ chmod 755 chmod.exercises

┌──(ceyona㉿kali)-[~/work]
└─$ ls -l chmod.exercises
-rwxr-xr-x 1 ceyona ceyona 0 Aug 23 21:11 chmod.exercises
```

q) Change the file permissions to match the following:

a. owner: Read, Write and Execute
b. group: Read and Execute
c. other: Execute

```
┌──(ceyona㉿kali)-[~/work]
└─$ chmod 755 chmod.exercises

┌──(ceyona㉿kali)-[~/work]
└─$ ls -l chmod.exercises
-rwxr-xr-x 1 ceyona ceyona 0 Aug 23 21:11 chmod.exercises
```

r) What was the command for changing the file permissions to -rwxr-x--x?

```
┌──(ceyona㉿kali)-[~/work]
└─$ chmod 751 chmod.exercises

┌──(ceyona㉿kali)-[~/work]
└─$ ls -l chmod.exercises
-rwxr-x--x 1 ceyona ceyona 0 Aug 23 21:11 chmod.exercises
```

Evaluation :

Marks : 10 (Deadline : 4 – Originality :3 – Completeness :3)

Deadline: 06.08.2024

In life there are no shortcuts. All things are connected. For success there is no fast lane. Work hard. Focus your energy, practice, remain honest, Truthful, loyal and committed.

-unknown