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

- 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	OS boot files
4	/dev	Device files
5	/etc	Configuration files
6	/home	Personal files
7	/lib	Kernel modules & Shared libraries
8	/proc	Hardware & Process Information
9	/sbin	System command binaries
10	/tmp	Temporary files
11	/var	Variable data files
12		
13		

- In Linux, there are three different files

  Regular file

  - A standard file that contains data, such as text, executable binaries, or other content.

  Directory

  - A folder that contains other files or directories, organizing the file system hierarchy.

  Special file

  - A file used to represent hardware devices or other special resources in the system.

  Block file

  - A special file that provides buffered access to hardware devices, such as a hard drive.

  Character file

  - A special file that provides unbuffered, direct access to hardware devices, such as a keyboard or terminal.

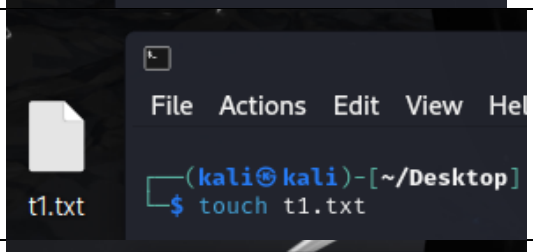
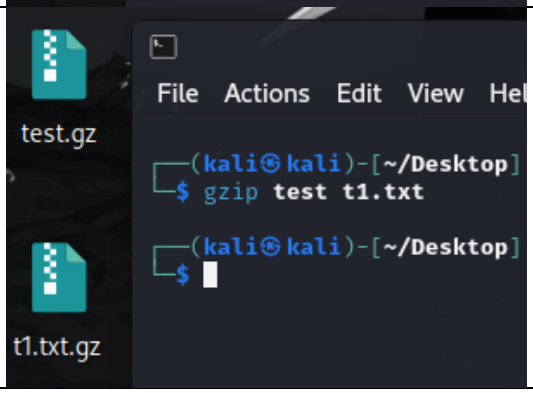
  Socket file

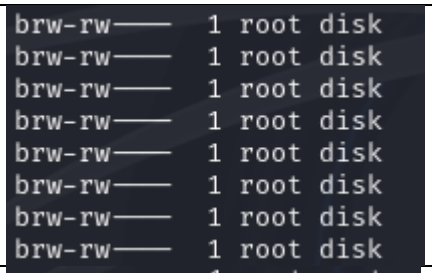
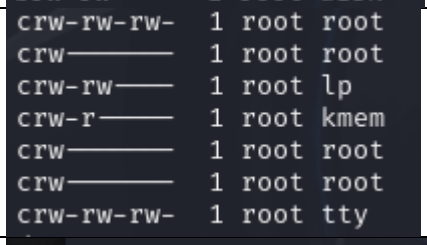
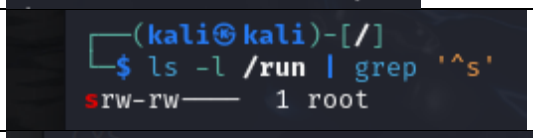
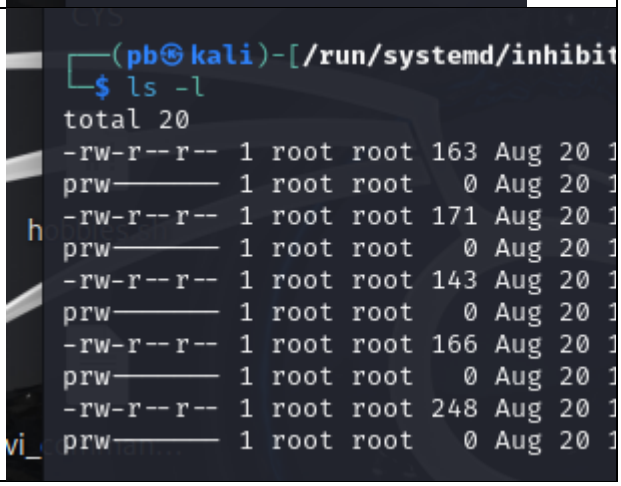
  - A special file used for inter-process communication, allowing data exchange between processes, often over a network.

  Pipe file

  - A special file that facilitates communication between processes by allowing data to be passed from one process to another.

Fill the below table:

File Type	Represented by (Hint ls)	Role	How to create	Screen shot
Regular file	-filename	Store data	touch filename	
• Text file	-filename.txt	Store information	touch filename .txt	
• Compressed file	- filename.zip/.tar /.rar etc	Reduce data size	gzip filename	
• Image	- filename.jpg/.png.bmp	Visual representation	NA	NA
Directory	d directoryname	Store files	mkdir	

Block file	b	Hardware interfaces that handle data in fixed-size blocks	NA	
Character file	c	Hardware interfaces that handle data one character at a time	NA	
Socket file	s	Inter-process communication(IPC)	NA	
pipe file	p	Inter-process communication(1 to another)	NA	

- Globbing
- Go back to CYS
- Create multiple subdirectories using single command

LS

Unit1

command

glob

Unit2

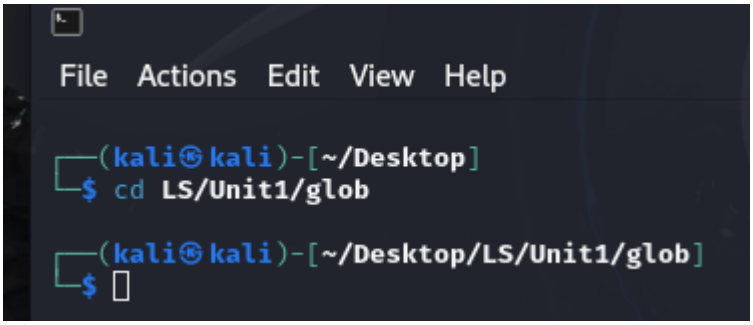
command

grep

Unit3

constructs

- Navigate to unit1/glob



- 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
```

```
┌──(pb㉿kali)-[~/Desktop/LS/Unit1/glob]
└─$ touch Commands.txt Commands1.txt Commands2.txt page{1..3}.html file1 file
10 file11 file2 File2 File3 file33 fileAB filea fileA fileAAA 'file(' 'file 2
'
```

- List all files starting with file

```
┌──(pb㉿kali)-[~/Desktop/LS/Unit1/glob]
└─$ ls file*
'file 2'   file1    file11   file33   fileAAA   filea
'file('    file10   file2    fileA    fileAB
```

- List all files starting with File

```
┌──(pb㉿kali)-[~/Desktop/LS/Unit1/glob]
└─$ ls File*
File2  File3
```

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

```
┌──(pb㉿kali)-[~/Desktop/LS/Unit1/glob]
└─$ ls file*[0-9]
'file 2'   file1   file10   file11   file2   file33
```

- List all files starting with file and ending with a letter

```
┌──(pb㉿kali)-[~/Desktop/LS/Unit1/glob]
└─$ ls file*[a-zA-Z]
fileA  fileAAA  fileAB  filea
```

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

```
┌──(pb㉿kali)-[~/Desktop/LS/Unit1/glob]
└─$ ls File[0-9]*
File2  File3
```

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

```
┌──(pb㉿kali)-[~/Desktop/LS/Unit1/glob]
└─$ ls File[0-9]
File2  File3
```

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

```
┌──(pb㉿kali)-[~/Desktop/LS/Unit1/glob]
└─$ ls [a-zA-Z]*[0-9]
 File2   File3  'file 2'   file1   file10   file11   file2   file33
```

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

```
┌──(pb㉿kali)-[~/Desktop/LS/Unit1/glob]
└─$ ls ?????
 File2   File3  'file('   file1   file2   fileA   filea
```

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

```
┌──(pb㉿kali)-[~/Desktop/LS/Unit1/glob]
└─$ ls [fF]*[3A]
File3  file33  fileA  fileAAA
```

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

```
┌──(pb㉿kali)-[~/Desktop/LS/Unit1/glob]
└─$ ls f[iR]*[0-9]
'file 2'   file1   file10   file11   file2   file33
```

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

```
┌──(pb㉿kali)-[~/Desktop/LS/Unit1/glob]
└─$ ls [\!F]*
 Commands.txt   'file 2'   file10   file33   fileAB       page2.html
 Commands1.txt  'file('    file11   fileA    filea        page3.html
 Commands2.txt   file1     file2    fileAAA  page1.html
```

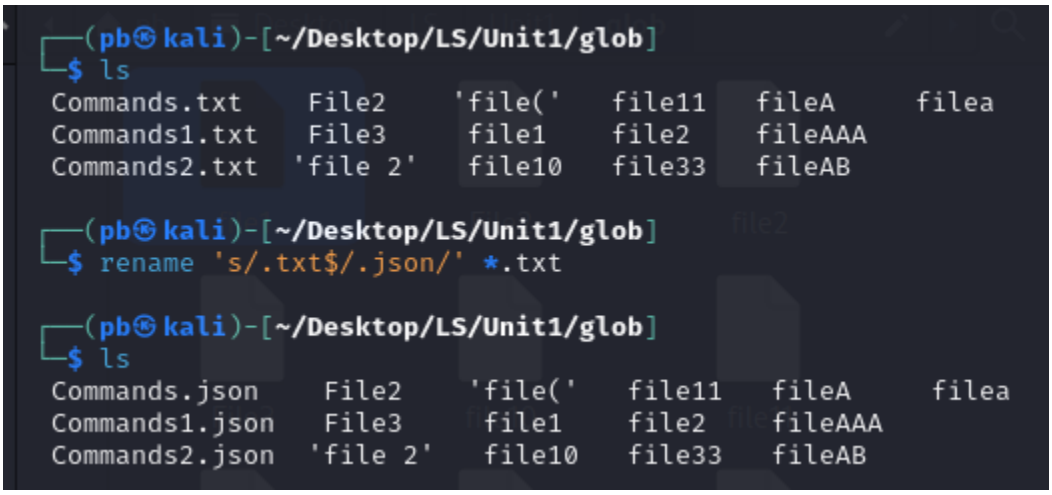
- Remove all the *.html

```
┌──(pb㉿kali)-[~/Desktop/LS/Unit1/glob]
└─$ ls
 Commands.txt    File3     file10   fileA    page1.html
 Commands1.txt  'file 2'   file11   fileAAA  page2.html
 Commands2.txt  'file('    file2    fileAB   page3.html
 File2           file1     file33   filea

┌──(pb㉿kali)-[~/Desktop/LS/Unit1/glob]
└─$ rm *.html

┌──(pb㉿kali)-[~/Desktop/LS/Unit1/glob]
└─$ ls
 Commands.txt    File2    'file('   file11   fileA    filea
 Commands1.txt   File3     file1    file2    fileAAA
 Commands2.txt  'file 2'   file10   file33   fileAB
```

- Rename *.txt to *.json

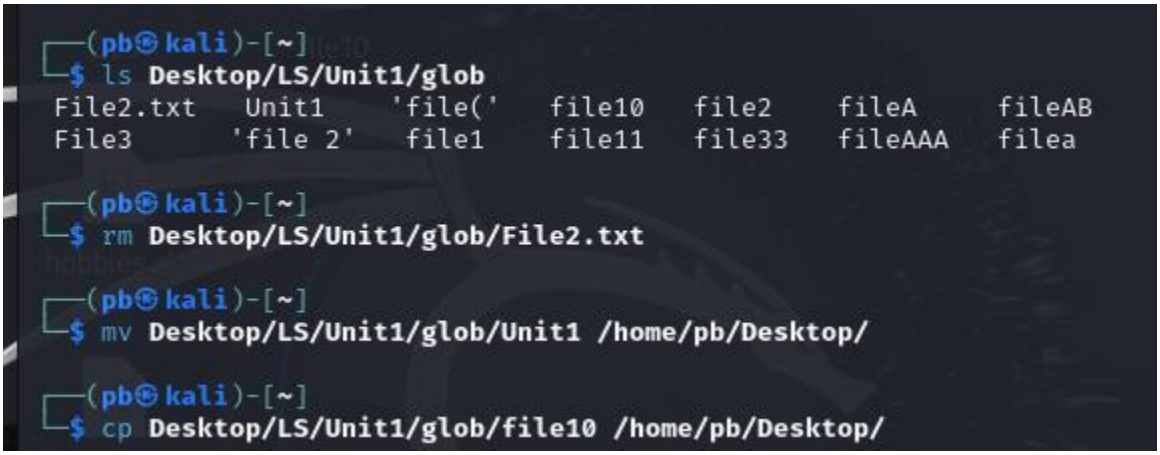


- Absolute path and relative path

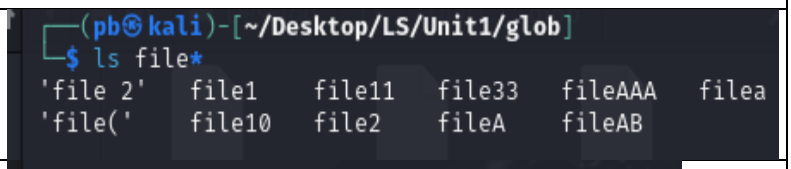
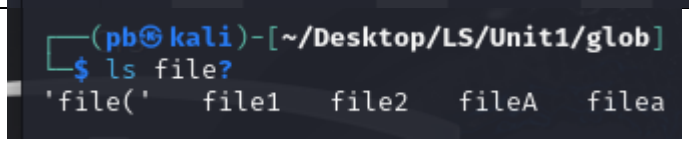
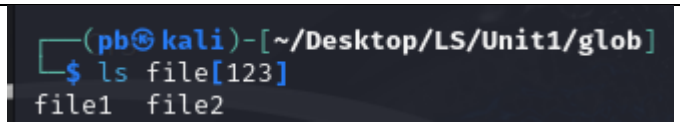
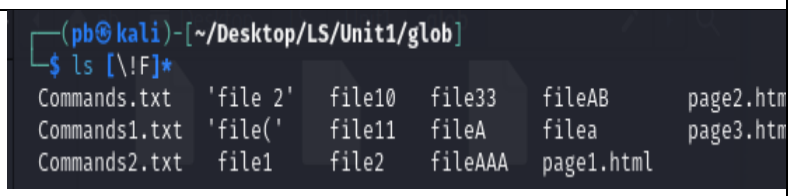
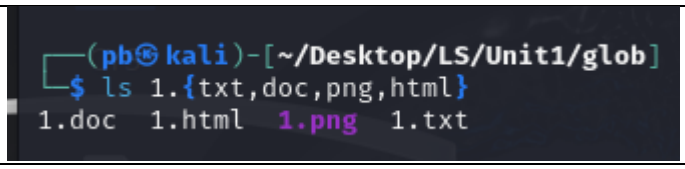
Use rm, mv, cp, ls with absolute path and relative path as per your choice.

Relative Path

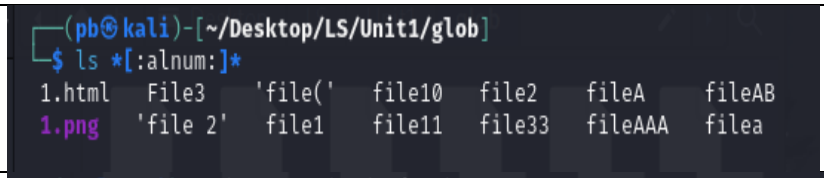
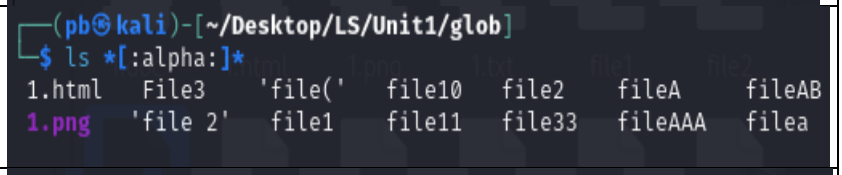
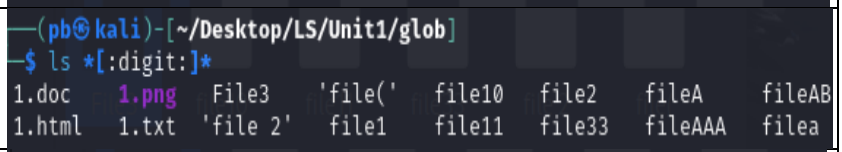
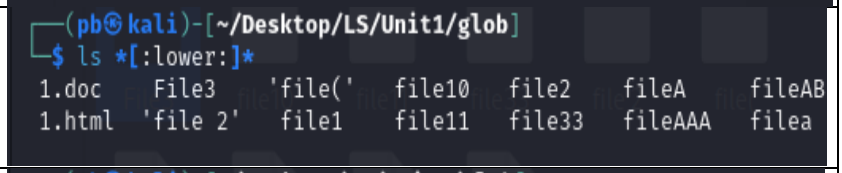
Absolute Path



- Wildcards

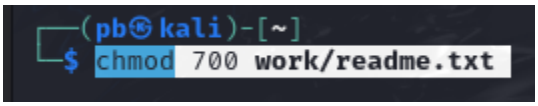
Notation	Use	Example	Screenshot
*	Match any character (0 or more)	ls file*	
?	Match any 1 character per question mark	ls file?	
[ ]	Match any one of the characters inside the brackets	ls file[123]	
[!]	Matches any character that is not a member of the set characters	ls [!F]*	
{ }	Match all characters inside braces	ls 1.{txt,doc,png,html}	

More on Character class

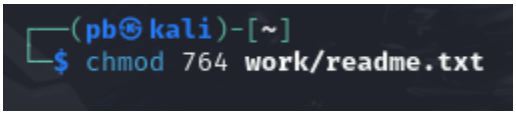
Notation	Use	Example	Screenshot
[:alnum:]	Matches any alphanumeric character	ls *[:alnum:]*	
[:alpha:]	Matches any alphabetical character	ls *[:alpha:]*	
[:digit:]	Matches any digit	ls *[:digit:]*	
[:lower:]	Matches any lowercase letter	ls *[:lower:]*	
[:upper:]	Matches any uppercase letter	ls *[:upper:]*	

4. change permission

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



- 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.



- 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.

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

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

- 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.

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

- 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.

```
┌──(pb㉿kali)-[~/work]
└─$ chmod 2711 readme.txt
```

- 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.

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

- What are the default permissions for the new file?

```
-rw-rw-r-- 1 pb pb 0 Nov 12 19:05 new.txt
```

- What was the command to view the file permissions?

```
┌──(pb㉿kali)-[~/work]
└─$ ls -l
total 0
-rw-rw-r-- 1 pb pb 0 Nov 12 19:05 new.txt
-rwx------ 1 pb pb 0 Aug 21 19:47 readme.txt
```

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

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

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

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

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

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

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

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

- Change the file permissions to match the following:
    - owner: Read and Write
    - group: Read
    - other: no permissions (None)

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

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

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

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

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

- Change the file permissions to match the following:
    - owner: Read, Write and Execute
    - group: Read and Execute
    - other: Execute

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

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

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