LINUX COMMANDS

cut

Linux cut command is useful for selecting a specific column of a file. It is used to cut a specific sections by byte position, character, and field and writes them to standard output.

Syntax : cut OPTION... [FILE]...

-b, --bytes=LIST: It is used to cut a specific section by bytes.

-c, --characters=LIST: It is used to select the specified characters.

-d, --delimiter=DELIM: It is used to cut a specific section by a delimiter.

Example : cut -b 2 exm.txt // Cut by byte

cut -c 1,6 exm.txt // Cut by character

cut -d ',' -f 2 file.csv // Cut by delimiter; **-f** option to specify the field(s).

Paste

It allows you to merge lines of files horizontally. It outputs lines consisting of the sequentially corresponding lines of each file specified as an argument, separated by tabs.

Syntax : paste OPTION... [FILE]...

Example :paste file1 file2 > file3

File 1

Iron Man
Thor
Captain America
Hulk
Spider Man

File 2

Black Widow

Captain Marvel
Dark Phoenix
Nebula

File 3

Iron Man Black Widow
Thor Captain Marvel
Captain America Dark Phoenix
Hulk Nebula
Spider Man

To use the _ (underscore) character as a delimiter instead of TAB, you would type:

```
paste -d '_' file1 file2
```

```
Iron Man_Black Widow
Thor_Captain Marvel
Captain America_Dark Phoenix
Hulk_Nebula
Spider Man_
```

du

The du command, short for “disk usage” reports the estimated amount of disk space used by given files or directories. It is practically useful for finding files and directories taking up large amounts of disk space.

Options : -a or –all : Displays disk usage information for all files and directories, including hidden ones.
-h or –human-readable : Displays sizes in human-readable format, using units such as KB, MB, GB, etc. This option makes it easier to interpret the disk usage information.

Syntax : du [OPTIONS]... FILE…
Example:

du -h /home

Output:

44K /home/mandeep/test/data

2.0M /home/mandeep/test/system design

24K /home/mandeep/test/table/sample_table/tree

df

df is used to check the overall disk space usage for mounted file systems,
Syntax : df OPTION... [FILE]...

Options : -a, --all: It is used to include pseudo, duplicate, remote file systems.
-h or –human-readable : Displays sizes in human-readable format, using units such as KB, MB, GB, etc. This option makes it easier to interpret the disk usage information.

tar

The Linux ‘tar’ stands for tape archive, which is used to create Archive and extract the Archive files.
Syntax : tar [options] [archive-file] [file or directory to be archived]

Options:
-c : Creates an archive by bundling files and directories together.
-x : Extracts files and directories from an existing archive.
-f : Specifies the filename of the archive to be created or extracted.
-t : Displays or lists the files and directories contained within an
-v :Displays verbose output, showing the progress of the archiving process.

archive.

Example:

1. **tar -cf myarchive.tar file1 file2**

   This command creates an archive named archive.tar with the content of file1 and file2

2. **Listing files from tar file**

   tar -tf myarch.tar

   Displays or lists the files and directories contained within the archive, myarch.

3. **Creating an uncompressed tar Archive using option -cvf**

This command creates a tar file called file.tar which is the Archive of all .c files in the current directory.

tar cvf file.tar *.c

- '-c': Creates a new archive.
- '-v': Displays verbose output, showing the progress of the archiving process.

- ‘-f’: Specifies the filename of the archive

zip

Zip is a file packaging and compression utility for Unix. All the files are stored inside a single file, i.e., .zip {.zip-filename} along with the .zip extension.

Syntax : zip [options] zipfile files_list

Example: $ zip myfile.zip filename.txt

Extract files from a zip file

Unzip can extract, test, or list files from the ZIP archive which is commonly detected on Unix. The default nature is for extracting into the current directory each file through the particular ZIP archive.

Example:

$unzip myfile.zip

uname

The command ‘uname‘ displays the information about the system.

Syntax: uname [OPTION]

Function	Shortcut
Kernel Name	-s
Kernel Release	-r
Kernel Version*	-v
Network Node Name (Hostname)	-n
Machine architecture	-m
Processor architecture	-p
Hardware Platform (OS architecture)	-i
Operating System	-o

chmod

chmod stands for change mode. This command is used for changing the mode of access.

chmod can be used in 2 ways.

- The first is using symbolic arguments
- The second is using numeric arguments.

Symbolic arguments

- **a** stands for all
- **u** stands for user
- **g** stands for group

❖ **o** stands for others

➢ Type either + or - to add a permission, or to remove it.
➢ Then you enter one or more permissions symbols (r , w , x).
➢ All followed by the file or folder name
.

Here are some examples:

- **chmod a+r filename** #everyone can now read
- **chmod a+rw filename** #everyone can now read and write
- **chmod o-rwx filename** #others cannot read, write and execute.
- **chmod og-r filename** #other and group can't read any

(**Note :** ls -l will display the files with their access permissions)

Numeric arguments

You use a digit that represents the permissions of the files. This number value can be a maximum of 7, and it's calculated in this way:

1 if has execution permission
2 if has write permission
4 if has read permission

This gives us 4 combinations:

- 0 no permissions
- 1 can execute
- 2 can write
- 3 can write, execute
- 4 can read
- 5 can read, execute
- 6 can read, write
- 7 can read, write and execute

We use them in pairs of 3, to set the permissions of all the 3 groups altogether:

chmod 777 filename
chmod 755 filename
chmod 644 filename

grep

The grep filter searches a file for a particular pattern of characters, and displays all lines that contain that pattern. The pattern that is searched in the file is referred to as the regular expression (grep stands for global search for regular expression and print out).

Syntax: grep [options] pattern [files]

Options:

-c : This prints only a count of the lines that match a pattern
-h : Display the matched lines, but do not display the filenames.
-i : Ignores, case for matching
-l : Displays list of a filenames only.
-n : Display the matched lines and their line numbers.
-v : This prints out all the lines that do not matches the pattern
-e exp : Specifies expression with this option. Can use multiple times.
-f file : Takes patterns from file, one per line.
-E : Treats pattern as an extended regular expression (ERE)
-w : Match whole word
-o : Print only the matched parts of a matching line,
with each such part on a separate output line.
-A n : Prints searched line and nlines after the result.
-B n : Prints searched line and n line before the result.
-C n : Prints searched line and n lines after the result.

Example :

os.txt

unix is great os. unix was developed in Bell labs.

learn operating system.

Unix linux which one you choose.

uNix is easy to learn.unix is a multiuser os.Learn unix .unix is a powerful.

1. Case insensitive search : The -i option enables to search for a string case insensitively in the given file. It matches the words like “UNIX”, “Unix”, “unix”.

```
$grep -i "UNix" os.txt
```

Output:

```
unix is great os. unix was developed in Bell labs.
Unix linux which one you choose.
uNix is easy to learn.unix is a multiuser os.Learn
unix .unix is a powerful.
```

2. Displaying the count of number of matches : We can find the number of lines that matches the given string/pattern

```
$grep -c "unix" os.txt
```

Output:

```
    2
```

3. Display the file names that matches the pattern : We can just display the files that contains the given string/pattern.

```
$grep -l "unix" *
```

or

```
$grep -l "unix" f1.txt f2.txt f3.xt f4.txt
```

Output:

4. Checking for the whole words in a file : By default, grep matches the given string/pattern even if it is found as a substring in a file. The -w option to grep makes it match only the whole words.

```
$ grep -w "unix" os.txt
```

Output:

```
unix is great os. unix was developed in Bell labs.
uNix is easy to learn.unix is a multiuser os.Learn
unix .unix is a powerful.
```

5. Matching the lines that start with a string : The ^ regular expression pattern specifies the start of a line. This can be used in grep to match the lines which start with the given string or pattern.

```
$ grep "^unix" os.txt
```

Output:

unix is great os. unix is free os.

6. Matching the lines that end with a string : The $ regular expression pattern specifies the end of a line. This can be used in grep to match the lines which end with the given string or pattern.

```
$ grep "os$" geekfile.txt
```

7. Regular Expression

```
grep -E 'Linux|linux' os.txt
```