CHAPTER 3: Ordinary file handling

Displaying and creating files (cat), Copying a file (cp), Deleting a file (rm), Renaming/ moving a file (mv), Paging output (more), Printing a file (lp), Knowing file type (file), Line, word and character counting (wc), Comparing files (cmp), Finding common between two files (comm), Displaying file differences (diff), Creating archive file (tar), Compress file (gzip), Uncompress file (gunzip), Archive file (zip), Extract compress file (unzip),

Brief idea about effect of cp, rm and mv command on directory.

Displaying and creating files (cat):

cat - concatenate files and print on the standard output

SYNOPSIS

cat [OPTION]... [FILE]...

DESCRIPTION

Concatenate FILE(s) to standard output.

-n, **--number**
number all output lines

-s, **--squeeze-blank**
suppress repeated empty output lines

-v, **--show-nonprinting**
use ^ and M- notation, except for LFD and TAB

--help display this help and exit

Displaying File Contents

The most basic and common usage of the cat command is to read the contents of files.

$cat filename

$cat filename1 filename2 filename3

Print Line Numbers

To display contents of a file with line numbers, use the `-n` option:

```
$ cat -n a.c aa.txt
     1  #include<stdio.h>
     2  int main()
     3  {
     4  printf("\nHello'");
     5  return 0;
     6  }
     7
     8  Wednesday-23-September-2020
```

Creating Files

Create a new file, use the `cat` command followed by the redirection operator (`>`) and the name of the file you want to create. Press `Enter`, type the text and once you are done, press the `CRTL+D` to save the file.

In the following example, we are creating a new file named `file1.txt`:

```
cat > file1.txt
```

If a file named `file1.txt` is present, it will be overwritten. Use the '`>>`' operator to append the output to an existing file.

```
cat >> file1.txt
```

Redirect Contents of File

Instead of displaying the output to stdout (on the screen) you can redirect it to a file.

The following command will copy the contents of `file1.txt` to `file2.txt` using the (`>`) operator :

```
cat file1.txt > file2.txt
```

If the `file2.txt` file doesn't exist, the command will create it. Otherwise, it will overwrite the file.

Use the (`>>`) operator to append the contents of `file1.txt` to `file2.txt` :

```
cat file1.txt >> file2.txt
```

if the file is not present, it will be created.

Concatenating Files

When passing two or more file names as arguments to the `cat` command the contents of the files will be concatenated. `cat` reads the files in the sequence given in its arguments and displays the file's contents in the same sequence.

The following command will concatenate the contents of `file1.txt` and `file2.txt` and write them to a new file `combinedfile.txt` using the (>) operator :

```
cat file1.txt file2.txt > combinedfile.txt
```

If the `combinedfile.txt` file doesn't exist, the command will create it. Otherwise, it will overwrite the file.

To concatenate the contents of `file1.txt` and `file2.txt` and append the result to `file3.txt` to use the (>>) operator:

```
cat file1.txt file2.txt >> file3.txt
```

If the file is not present, it will be created.

What are wildcards in Linux ?
A wildcard is a symbol used to replace or represent one or more characters. Wildcards are typically either an asterisk (*), which represents one or more characters or question mark (?),which represents a single character.

Copying a file (cp):

NAME

cp - copy files and directories

SYNOPSIS/SYNTAX:

cp [OPTION]... SOURCE... DIRECTORY

DESCRIPTION

Copy SOURCE to DEST, or multiple SOURCE(s) to DIRECTORY.

-i, **--interactive**

prompt before overwrite (overrides a previous **-n** option)

-v, **--verbose**

explain what is being done

-R, **-r**, **--recursive**

copy directories recursively

Linux Copy File Examples

To make a copy of a file called file.txt in the current directory as newfile.txt, enter:

```
$ cp file.txt newfile.txt

To display files:
$ ls -l *.txt
```

Sample outputs:

```
-rw-r--r--  1 student  student  20 Mar 20 17:42 file.txt
-rw-r--r--  1 student  student 20 Mar 20 17:43 newfile.txt
```

Copy a file to another directory

To copy a file from your current directory into another directory called /tmp/, enter:

```
$ cp filename /tmp
$ ls /tmp/filename
$ cd /tmp
$ ls
```

Copying all files

The star wildcard represents anything i.e. all files. To copy all the files in a directory to a new directory, enter:

```
$ cp * /home/student/backup
```

The star wildcard represents anything whose name ends with the .doc extension. So, to copy all the text files (*.txt) in a directory to a new directory, enter:

```
$ cp *.txt /home/student/backup
```

Recursive copy

To copy a directory, including all its files and subdirectories, to another directory, enter (copy directories recursively):

```
$ cp -R * /home/student/backup
```

To copy a directory(Student1 and all ts contents), including all its files and subdirectories, to another directory(Student2), enter (copy directories recursively):

```
$ cp -R Student1 Student2
```

rmdir command in Linux With Examples:

rmdir command is used remove empty directories from the filesystem in Linux. The rmdir command removes each and every directory specified in the command line only if these directories are empty. So if the specified directory has some directories or files in it then this cannot be removed by *rmdir* command.

NAME

rmdir - remove empty directories

SYNOPSIS

rmdir [OPTION]... DIRECTORY...

DESCRIPTION

Remove the DIRECTORY(ies), if they are empty.

Example : Remove the directory *mydir/mydir1* if it is empty. Then, remove directory mydir, if it is empty after *mydir/mydir1* was removed.

```
$rmdir mydir/mydir1 mydir
```

```
NOTE: First delete the mydir1 subdirectory the delete the mydir directory.
```

rm : Remove a file and directory

```
COMMAND NAME:
       rm - remove files or directories

SYNOPSIS
```

```
       rm [OPTION]... [FILE]...

DESCRIPTION
       This manual page documents the GNU version of rm.  rm removes each
specified file.  By default, it  does not remove directories.

OPTIONS

       -i     prompt before every removal / confirmation messages

       -r, -R, --recursive
              remove directories and their contents recursively

       -d, --dir
              remove empty directories

       -v, --verbose
              explain what is being done

       --help display this help and exit

Example:

To remove file(s)

$rm -I a.txt

$rm -i aa.txt bb.txt cc.txt

$rm a*.txt

Example:

To remove directory(s) and all its contents:

$rm -r INDIA
```

mv : move or rename a file(s) or directory(s):

```
NAME

       mv - move (rename) files


SYNOPSIS

       mv [OPTION]... SOURCE... DIRECTORY

DESCRIPTION

       Rename SOURCE(OLD) to DEST(NEW), or move SOURCE(s) to DIRECTORY.
```

```
Options:
       -i, --interactive
              prompt before overwrite
```

1. How to move a file to different directory

The first and the simplest example is to move a file. To do that, you just have to specify the source file and the destination directory or file.

mv source_file target_directory

This command will move the source_file and put it in the target_directory.

2. How to move multiple files

mv *.txt target_directory

Paging output (more):

NAME

more - file perusal filter for crt viewing

SYNOPSIS

more [options] file...

more command is used to view the text files in the command prompt, displaying one screen at a time in case the file is large (For example log files). The more command also allows the user do scroll up and down through the page.

Example:

debasis@LAPTOP-H3N6JCNE:~$ more caseExCal.sh

debasis@LAPTOP-H3N6JCNE:~$ cat -n caseExCal.sh | more

The Linux lp printing command:

Printing a file (lp)

The lp command is used to print files on Unix and Linux systems. The name " for **"line printer"**. As with most Unix commands there are a fairly large number of options available to enable flexible printing capabilities.

Let's look at some lp printing commands examples.

Linux lp printing command examples:

```
lp /etc/passwd
```

NAME

lp - line printer devices

SYNOPSIS

#include <linux/lp.h>

file command in Linux with examples:

Knowing file type (file)

NAME

file — determine file type

Syntax:
file option filename

```
debasis@LAPTOP-H3N6JCNE:~$ file abcd.txt
abcd.txt: ASCII text, with overstriking
debasis@LAPTOP-H3N6JCNE:~$ file student
student: directory
```

wc Command in Linux (Count Number of Lines, Words, and Characters):

```
       wc - print newline, word, and byte counts for each file

SYNOPSIS

       wc [OPTION]... [FILE]...

DESCRIPTION

       Print  newline,  word, and byte counts for each FILE, and a total
line if more than one FILE is specified.  A word is a non-zero-length
sequence of characters delimited by white space.

       With no FILE, or when FILE is -, read standard input.

       The options below may be used to select which counts are printed,
always in the following order:

 new-line, word, character, byte, maximum line length.

        -l, --lines

              print the newline counts

        -w, --words

              print the word counts

       -c, --bytes

              print the byte counts

       -m, --chars

              print the character counts
```

Example:

```
debasis@LAPTOP-H3N6JCNE:~$ cal>abcd.txt

debasis@LAPTOP-H3N6JCNE:~$ cat -n abcd.txt

     1     November 2020

     2  Su Mo Tu We Th Fr Sa

     3   1  2  3  4  5  6  7

     4   8  9 10 11 12 13 14
```

```
     5  15 16 17 18 19 20 21
     6  22 23 24 25 26 27 28
     7  29 30
     8
debasis@LAPTOP-H3N6JCNE:~$ wc abcd.txt
 8  39 188 abcd.txt
debasis@LAPTOP-H3N6JCNE:~$ wc -m abcd.txt
188 abcd.txt
debasis@LAPTOP-H3N6JCNE:~$ wc -l abcd.txt
8 abcd.txt
debasis@LAPTOP-H3N6JCNE:~$ wc -w abcd.txt
39 abcd.txt
debasis@LAPTOP-H3N6JCNE:~$ wc -c abcd.txt
188 abcd.txt
```

wc stands for **word count**. As the name implies, it is mainly used for counting purpose.

- It is used to find out **number of lines**, **word count**, **byte and characters count** in the files specified in the file arguments.
- By default it displays **four-columnar output.**
- First column shows number of lines present in a file specified, second column shows number of words present in the file, third column shows number of characters present in file and fourth column itself is the file name which are given as argument.

Syntax:

```
wc [OPTION]... [FILE]...
```

Let us consider two files having name **state.txt** and **capital.txt** containing 5 names of the Indian states and capitals respectively.

```
$ cat state.txt
Andhra Pradesh
Arunachal Pradesh
Assam
Bihar
Chhattisgarh

$ cat capital.txt
Hyderabad
```

```
Itanagar
Dispur
Patna
Raipur
```

Passing only one file name in the argument.

```
$ wc state.txt
 5  7 63 state.txt
       OR
$ wc capital.txt
 5  5 45 capital.txt
```

Passing more than one file name in the argument.

```
$ wc state.txt capital.txt
  5   7  63 state.txt
  5   5  45 capital.txt
 10  12 108 total
```

cmp Command in Linux with examples:

Comparing files (cmp)

cmp command in Linux/UNIX is used to compare the two files byte by byte and helps you to find out whether the two files are identical or not.

- When cmp is used for comparison between two files, it reports the location of the first mismatch to the screen if difference is found and if no difference is found *i.e* the files compared are identical.
- cmp displays no message and simply returns the prompt if the the files compared are identical.

```
GNU cmp - compare two files byte by byte

SYNOPSIS

       cmp [OPTION]... FILE1 [FILE2 [SKIP1 [SKIP2]]]

DESCRIPTION

       Compare two files byte by byte.

The  optional  SKIP1 and SKIP2 specify the number of bytes to skip at the
beginning of each file (zero by default).

Mandatory arguments to long options are mandatory for short options too.

       -b, --print-bytes

              print differing bytes
```

```
#Example using cmp command:

debasis@LAPTOP-H3N6JCNE:~/search$ cat a.txt
I am a student of BCA.
job over

debasis@LAPTOP-H3N6JCNE:~/search$ cat b.txt
I am a student of MCA.
Job Over

debasis@LAPTOP-H3N6JCNE:~/search$ cmp  a.txt b.txt
a.txt b.txt differ: byte 19, line 1
```

MAKAUT 2019-20 Q 9(a).
Write a shell script , which receives two filenames as arguments , checks whether the two files contents are same or NOT and if they are same then the second file is deleted.

```
debasis@LAPTOP-H3N6JCNE:~/BCA$ vi cmpRmv.sh

#Remove second file if both the contents are SAME using COMMAND Line
argument
#!/bin/bash

file1=$1
file2=$2

if cmp $file1 $file2
then
        echo "Both are SAME."
        rm -i $file2
else
        echo "Contents are DFFERENT."
fi
echo "Job OVER."

Output:
debasis@LAPTOP-H3N6JCNE:~/BCA$ sh cmpRmv.sh a3.txt a33.txt
Both are SAME.
rm: remove regular file 'a33.txt'? y
Job OVER.
```

comm command in Linux with examples:

Finding common between two files (comm)

```
NAME

       comm - compare two sorted files line by line

SYNOPSIS

       comm [OPTION]... FILE1 FILE2

DESCRIPTION

       Compare sorted files FILE1 and FILE2 line by line.
```

With no options, produce three-column output. Column one contains lines unique to FILE1, column two contains lines unique to FILE2, and column three contains lines common to both files.

```
   [OPTION]:  -1     suppress column 1 (lines unique to FILE1)

   [OPTION]:  -2     suppress column 2 (lines unique to FILE2)

   [OPTION]:  -3     suppress column 3 (lines that appear in both files)

 Manual

   [OPTION]:  -12     suppress column 1 and 2 (Display only matching word
column 3)
```

```
debasis@LAPTOP-H3N6JCNE:~/search$ cat comm1.txt
Akash
Amit
Anirban
Bikash
Rajat
jayanta
kajal
```

```
debasis@LAPTOP-H3N6JCNE:~/search$ cat comm2.txt
Akash
Amit
Anirban
Bikash
```

```
Prakash
Rajat
Sanjay
Joy
debasis@LAPTOP-H3N6JCNE:~/search$ comm  comm1.txt comm2.txt
output:
                Akash
                Amit
                Anirban
                Bikash
        Prakash
                Rajat
        Sanjay
jayanta
        joy
kajal

output:
```
Column one contains lines unique to comm1.txt, column two contains lines unique to comm2.txt, and column three contains lines common to both files.

#suppresh column 3:

```
debasis@LAPTOP-H3N6JCNE:~/search$ comm -3 comm1.txt comm2.txt
output:
        Prakash
        Sanjay
jayanta
        joy
kajal
```
#suppresh column 1 and 2:

```
debasis@LAPTOP-H3N6JCNE:~/search$ comm -1 -2  comm1.txt comm2.txt

Akash
```
Amit
Anirban
Bikash
Rajat

SAME OUTPUT:

debasis@LAPTOP-H3N6JCNE:~/search$ comm -12 comm1.txt comm2.txt

Akash
Amit
Anirban
Bikash
Rajat

diff command in Linux with examples:

Displaying file differences (diff)

diff stands for **difference**. This command is used to display the differences in the files by comparing the files line by line. Unlike its fellow members, cmp and comm, **it tells us which lines in one file have is to be changed to make the two files identical.**

NAME

GNU diff - compare files line by line

SYNOPSIS

diff [OPTION]... FILES

debasis@LAPTOP-H3N6JCNE:~$ cat -n a.txt

1 today is Monday.

2 This is testing file.

debasis@LAPTOP-H3N6JCNE:~$ cat -n b.txt

1 Today is Monday.job over.

2 MSIT

3 INDIA

4 Test

debasis@LAPTOP-H3N6JCNE:~$ diff a.txt b.txt

1,2c1,4

< today is Monday.

< This is testing file.

> Today is Monday.job over.

> MSIT

> INDIA

> Test

Creating archive file (tar), Compress file (gzip), Uncompress file (gunzip), Archive file (zip), Extract compress file (unzip)

touch command:

The ***touch*** command is a standard command used in UNIX/Linux operating system which is used to create, change and modify timestamps of a file. Basically, there are two different commands to create a file in the Linux system which is as follows:

- **cat command**: It is used to create the file with content.
- **touch command:** It is used to create a file without any content. The file created using touch command is empty. This command can be used when the user doesn't have data to store at the time of file creation.
- **Touch command Syntax to create a new file:** You can create a single file at a time using touch command.

Syntax:

```
touch file_name
```

touch command to create multiple files: touch command can be used to create the multiple numbers of files at the same time. These files would be empty while creation.

Syntax:

```
touch File1_name1 File2_name File3_name
```

touch command to create multiple files with filename 1 to 20 with text files:

```
debasis@LAPTOP-H3N6JCNE:~/student$ touch {1..20}.txt

debasis@LAPTOP-H3N6JCNE:~/student$ ls
1.txt   11.txt  13.txt  15.txt  17.txt  19.txt  20.txt  4.txt  6.txt  8.txt
```

```
10.txt  12.txt  14.txt  16.txt  18.txt  2.txt   3.txt   5.txt  7.txt  9.txt
```

```
debasis@LAPTOP-H3N6JCNE:~/student$ touch a{1..5}.sh
output:
debasis@LAPTOP-H3N6JCNE:~/student$ ls -l
total 0
-rw-r--r-- 1 debasis debasis 0 Jan  8 17:04 a1.sh
-rw-r--r-- 1 debasis debasis 0 Jan  8 17:04 a2.sh
-rw-r--r-- 1 debasis debasis 0 Jan  8 17:04 a3.sh
-rw-r--r-- 1 debasis debasis 0 Jan  8 17:04 a4.sh
-rw-r--r-- 1 debasis debasis 0 Jan  8 17:04 a5.sh
```

tar command:

The Linux 'tar' stands for **tape archive**, is used to create Archive and extract the Archive files. tar command in Linux is one of the important command which provides archiving functionality in Linux.

We can use Linux tar command to create compressed or uncompressed Archive files and also maintain and modify them.

What is an Archive file?
An Archive file is a file that is composed of one or more files along with metadata. Archive files are used to collect multiple data files together into a single file for easier portability and storage, or simply to compress files to use less storage space.

The Linux "**tar**" stands for tape archive, which is used by large number of Linux/**Unix** system administrators to deal with tape drives backup. The **tar command** used to rip a collection of files and directories into highly compressed archive file commonly called tarball or **tar**, gzip and bzip in Linux

Syntax:

tar [options] [archive-file] [file or directory to be archived]
Options:
-c : Creates Archive
-x : Extract the archive

-f : creates archive with given filename
-t : displays or lists files in archived file
-u : archives and adds to an existing archive file
-v : Displays Verbose Information
-A : Concatenates the archive files
-z : zip, tells tar command that create tar file using gzip
-j : filter archive tar file using tbzip
-W : Verify a archive file
-r : update or add file or directory in already existed .tar file

-c : Creates Archive
Creating an uncompressed tar Archive using option -cvf :

```
debasis@LAPTOP-H3N6JCNE:~/student$ tar -cvf   fileTAR.tar  *.txt
1.txt
10.txt
11.txt
12.txt
13.txt
14.txt
15.txt
16.txt
17.txt
18.txt
19.txt
2.txt
20.txt
3.txt
4.txt
5.txt
6.txt
7.txt
8.txt
9.txt
```

```
debasis@LAPTOP-H3N6JCNE:~/student$ file fileTAR.tar
fileTAR.tar: POSIX tar archive (GNU)
```

x : Extract the archive
Extracting files from Archive using option -xvf :

```
debasis@LAPTOP-H3N6JCNE:~/student/stud$ tar -xvf fileTAR.tar
1.txt
10.txt
11.txt
12.txt
13.txt
14.txt
```

15.txt
16.txt
17.txt
18.txt
19.txt
2.txt
20.txt
3.txt
4.txt
5.txt
6.txt
7.txt
8.txt
9.txt
debasis@LAPTOP-H3N6JCNE:~/student/stud$ ls
1.txt 11.txt 13.txt 15.txt 17.txt 19.txt 20.txt 4.txt 6.txt 8.txt fileTAR.tar
10.txt 12.txt 14.txt 16.txt 18.txt 2.txt 3.txt 5.txt 7.txt 9.txt

-r : update or add file or directory in already existed .tar file:

debasis@LAPTOP-H3N6JCNE:~/student/stud$ tar -rvf fileTAR.tar *.c
a.c
b.c

Ex. Update existing tar file in Linux

```
$ tar -rvf file.tar *.c
```