



DEPARTMENT OF COMPUTER ENGINEERING

Subject: OSY	Subject Code: 22516
Semester: 5 th Semester	Course: Computer Engineering
Laboratory No: V118	Name of Subject teacher: Prof. Natasha Brahmne
Name of Student: Siddharth Shah	Roll Id: 22203A0041

Experiment No:	6
Title of Experiment:	Execute file and directory commands.

• Practical Related Questions

1. What are the different options of ls command? Write down the command along with options and note down the output. (Use \$man command to check options)

→

```
mc@kali: ~/OSY
User Commands
ls(1)
NAME
ls - list directory contents
SYNOPSIS
ls [OPTION]... [FILE]...
DESCRIPTION
List information about the FILES (the current directory by default). Sort entries alphabetically if none of -cftuvSUX nor --sort is specified.
Mandatory arguments to long options are mandatory for short options too.
-a, --all
do not ignore entries starting with .
-A, --almost-all
do not list implied . and ..
--author
with -l, print the author of each file
-b, --escape
print C-style escapes for nongraphic characters
--block-size=SIZE
with -l, scale sizes by SIZE when printing them; e.g., '--block-size=M'; see SIZE format below
-B, --ignore-backups
do not list implied entries ending with ~
-c, with -lt: sort by, and show, ctime (time of last change of file status information); with -l: show ctime and sort by name; otherwise: sort by ctime, newest first
-C
list entries by columns
--color[=WHEN]
color the output WHEN; more info below
-d, --directory
list directories themselves, not their contents
-D, --dired
generate output designed for Emacs' dired mode
-f
list all entries in directory order
-F, --classify[=WHEN]
append indicator (one of */>@|) to entries WHEN
--file-type
likewise, except do not append '*'
--format=WORD
across -x, commas -m, horizontal -x, long -l, single-column -1, verbose -l, vertical -C
Manual page ls(1) line 1 (press h for help or q to quit)
```

2. What are two different options of mv command?

→

1. -i (interactive)
2. -f (Force)
3. -n (no-clobber)
4. -b(backup)
5. -version

3. what is use of split commands

→ The split command in Linux is a highly useful tool for dividing large files into smaller, more manageable pieces. This command is often used in scenarios where you need to break down a large file for easier data processing or distribution. In this article, we'll dissect the split command, its syntax, options, and some practical examples of its usage.

Syntax of split Command in Linux:

The basic syntax of the split command in Linux is as follows:

Split [option][file prefix]

Where:

- **OPTIONS:** These are optional parameters that modify the behavior of the split command.
- **FILE:** This is the input file that you want to split.
- **PREFIX:** This parameter specifies the prefix for the output files. The default prefix is 'x'.

4. How to use join command?

→ The Linux join command is a powerful tool that is used to merge two different files based on a common field. command reads contents of two files and merges them based on specified field, which can be a string or a numeric value. In this article, we will discuss various aspects of join command and its usage.

The syntax for join command is as follows –

join [options] file1 file2

t – This option is used to specify delimiter character used in files. By default, delimiter is a blank space.

- -1 – This option is used to specify field number in first file.
- -2 – This option is used to specify field number in second file.
- -a – This option is used to print all lines from both files, including those that do not match.
- -e – This option is used to replace missing fields with a specified value.

Example 1

File 1 –

1 Alpha 2 Bravo 3 Charlie 4 Delta 5 Echo

File 2 –

2 20 3 30 4 40 5 50 6 60

We can join these two files based on first field in each file using following command
– join file1 file2

The output will be as follows –

2 Bravo 20 3 Charlie 30 4 Delta 40 5 Echo 50

• EXERCISE

1. Write output of following commands

- a) Display all file names whose name starts with 'a' and ends with 'y'.

```
(mc@kali)-[~/OSY]
└─$ man ls

(mc@kali)-[~/OSY]
└─$ ls a*t
a1.txt  a1cp.txt  a2.txt  a3.txt
```

- b) Enlist all files beginning with 'm' and ending with any range 1 to 5.

```
(mc@kali)-[~/OSY]
└─$ ls a*[1-5]
a1  a2  a3
```

- c) Show the contents of the files whose file names contains exactly two characters.

```
(mc@kali)-[~/OSY]
└─$ cat ??
Hello
Mohit
Atharva
Shravan
Neel
Hitesh
Shelar
Parth
Abhishek
Vedant
Rane
Pranit
Hello Kali Linux
```

- d) Create a file ABCD.txt, create a copy with XXXX.txt .Rename the original file with AACD.txt. Delete the file XXXX.txt.

```
(mc@kali)-[~/OSY]
└─$ cat > ABCD.txt
Mohit Chaudhari
^C

(mc@kali)-[~/OSY]
└─$ cp ABCD.txt XXXX.txt

(mc@kali)-[~/OSY]
└─$ mv ABCD.txt AACD.txt

(mc@kali)-[~/OSY]
└─$ ls
AACD.txt  a1cp.txt  a3      chapter1  combined1.txt  names  xz
XXXX.txt  a2        a3.txt  chapter2  dte            surname
a1        a2.txt    abc     combine.txt  name           xaa

(mc@kali)-[~/OSY]
└─$ rm AACD.txt

(mc@kali)-[~/OSY]
└─$ ls
XXXX.txt  a1cp.txt  a2.txt  a3.txt  chapter1  combine.txt  dte  names  xaa
a1        a2        a3      abc     chapter2  combined1.txt  name  surname  xz
```

- e) Display the inodes of any 2 files at the same time.

```
(mc@kali)-[~/OSY]
└─$ cat a1

(mc@kali)-[~/OSY]
└─$ cat > a1
Mohit
Atharva
^C

(mc@kali)-[~/OSY]
└─$ ls -i a1 a2
1966216 a1 1969438 a2
```

2. List all file processing commands.

→

- cat
- less
- more
- head
- tail
- file
- diff
- wc
- nano

- vim
- sed
- awk

3. How many lines will be displayed with head command if number is not specified?

→ If the number of lines is not specified with the head command, it will display the first 10 lines of the file by default.

4. Create 2 files chapter1 and chapter2 and perform the following operations

- 1) Copy content of chapter1 to chapter2 by asking the user before overwrite.
- 2) Display the inodes of 2 files
- 3) Rename the file 'chapter1' to 'lesson1'.

```
(mc@kali)-[~/OSY]
└─$ cat > chapter1
22203A0029
^C

(mc@kali)-[~/OSY]
└─$ cat > chapter2
22203A0012
^C

(mc@kali)-[~/OSY]
└─$ cat chapter1 chapter2
22203A0029
22203A0012

(mc@kali)-[~/OSY]
└─$ cp -i chapter1 chapter2
cp: overwrite 'chapter2'? yes

(mc@kali)-[~/OSY]
└─$ cp chapter1 chapter2

(mc@kali)-[~/OSY]
└─$ cat chapter1 chapter2
22203A0029
22203A0029
```

5. Execute the following command.

- 1) \$ls a*n
- 2) \$ls s?
- 3) \$cat abc>> xz

```
(mc@kali)-[~/0SY]
└─$ ls a*n
ls: cannot access 'a*n': No such file or directory

(mc@kali)-[~/0SY]
└─$ ls x*a
xaa

(mc@kali)-[~/0SY]
└─$ ls
a1          a1cp.txt  a2.txt    a3.txt    chapter2  xaa
a1.txt     a2        a3        chapter1  combine.txt xz
```

```
(mc@kali)-[~/0SY]
└─$ ls s?
ls: cannot access 's?': No such file or directory
```

```
(mc@kali)-[~/0SY]
└─$ cat abc >> xz

(mc@kali)-[~/0SY]
└─$ cat abc
Hello Kali Linux
```

- Program code

1. Create four files a1,a2,a3

→

```
(mc@kali)-[~/OSY]
└─$ cat a1 a2 a3

(mc@kali)-[~/OSY]
└─$ ls
a1          a2  abc      chapter2  xaa
a1cp.txt    a3  chapter1 combine.txt xz
```

2. Apply different commands like ls, mv, cp, rm, join, split, and check the list of files at the end

1. ls

```
(mc@kali)-[~/OSY]
└─$ touch a1.txt a2.txt a3.txt

(mc@kali)-[~/OSY]
└─$ ls
a1          a1cp.txt  a2.txt  a3.txt  chapter1  combine.txt  xz
a1.txt     a2         a3      abc     chapter2  xaa
```

2. mv

```
(mc@kali)-[~/OSY]
└─$ mv a1.txt /home/mc/Desktop

(mc@kali)-[~/OSY]
└─$ la
a2.txt  a3.txt

(mc@kali)-[~/OSY]
└─$ cd ..

(mc@kali)-[~]
└─$ cd Desktop

(mc@kali)-[~/Desktop]
└─$ ls
a1.txt
```


3. rm

```
(mc@kali)-[~/05Y]
$ rm a1.txt

(mc@kali)-[~/05Y]
$ ls
a1          a2          a3          abc          chapter2    xaa
a1cp.txt    a2.txt      a3.txt      chapter1     combine.txt xz
```

4. join

```
(mc@kali)-[~/05Y]
$ cat a1.txt
Hello there! I'm Mohit

(mc@kali)-[~/05Y]
$ cat a2.txt
Chaudhari from C05IA
```

```
(mc@kali)-[~/05Y]
$ cat combine.txt
Hello there! I'm Mohit
Chaudhari from C05IA

(mc@kali)-[~/05Y]
$ ls
a1.txt  a1cp.txt  a2.txt  a3.txt  combine.txt
```

5. split

```
(mc@kali)-[~/05Y]
$ split -12 a1.txt

(mc@kali)-[~/05Y]
$ ls
a1.txt  a1cp.txt  a2.txt  a3.txt  combine.txt  xaa

(mc@kali)-[~/05Y]
$ cat xaa
Hello there! I'm Mohit
```

6. ls -l

```
(mc@kali)~[~/OSY]  
$ ls -l  
total 28  
-rw-rw-r-- 1 mc mc 0 Sep 2 23:50 a1  
-rw-rw-r-- 1 mc mc 23 Sep 2 23:12 a1cp.txt  
-rw-rw-r-- 1 mc mc 0 Sep 2 23:50 a2  
-rw-rw-r-- 1 mc mc 0 Sep 3 00:18 a2.txt  
-rw-rw-r-- 1 mc mc 0 Sep 2 23:50 a3  
-rw-rw-r-- 1 mc mc 0 Sep 3 00:18 a3.txt  
-rw-rw-r-- 1 mc mc 17 Sep 3 00:11 abc  
-rw-rw-r-- 1 mc mc 11 Sep 3 00:01 chapter1  
-rw-rw-r-- 1 mc mc 11 Sep 3 00:03 chapter2  
-rw-rw-r-- 1 mc mc 45 Sep 2 23:18 combine.txt  
-rw-rw-r-- 1 mc mc 23 Sep 2 23:29 xaa  
-rw-rw-r-- 1 mc mc 17 Sep 3 00:11 xz
```

- Set 5 questions

1. Execute file manipulation commands: ls, rm, mv, cp, join, split, ls), head, tail, touch.

→

1. ls

```
(mc@kali)-[~/OSY]
$ touch a1.txt a2.txt a3.txt

(mc@kali)-[~/OSY]
$ ls
a1          a1cp.txt  a2.txt    a3.txt    chapter1  combine.txt  xz
a1.txt     a2         a3         abc       chapter2  xaa
```

2. mv

```
(mc@kali)-[~/OSY]
$ mv a1.txt /home/mc/Desktop

(mc@kali)-[~/OSY]
$ la
a2.txt  a3.txt

(mc@kali)-[~/OSY]
$ cd ..

(mc@kali)-[~]
$ cd Desktop

(mc@kali)-[~/Desktop]
$ ls
a1.txt
```

3. rm

```
(mc@kali)-[~/OSY]
$ rm a1.txt

(mc@kali)-[~/OSY]
$ ls
a1          a2          a3          abc          chapter2    xaa
a1cp.txt    a2.txt     a3.txt     chapter1    combine.txt  xz
```

4. join

```
(mc@kali)-[~/OSY]
$ cat a1.txt
Hello there! I'm Mohit
```

```
(mc@kali)-[~/OSY]
$ cat a2.txt
Chaudhari from C05IA
```

```
(mc@kali)-[~/OSY]
$ cat combine.txt
Hello there! I'm Mohit
Chaudhari from C05IA

(mc@kali)-[~/OSY]
$ ls
a1.txt  a1cp.txt  a2.txt  a3.txt  combine.txt
```

5. split

```
(mc@kali)-[~/OSY]
$ split -l2 a1.txt

(mc@kali)-[~/OSY]
$ ls
a1.txt  a1cp.txt  a2.txt  a3.txt  combine.txt  xaa

(mc@kali)-[~/OSY]
$ cat xaa
Hello there! I'm Mohit
```

6. ls -l

```
(mc@kali)-[~/OSY]
$ ls -l
total 28
-rw-rw-r-- 1 mc mc  0 Sep  2 23:50 a1
-rw-rw-r-- 1 mc mc 23 Sep  2 23:12 a1cp.txt
-rw-rw-r-- 1 mc mc  0 Sep  2 23:50 a2
-rw-rw-r-- 1 mc mc  0 Sep  3 00:18 a2.txt
-rw-rw-r-- 1 mc mc  0 Sep  2 23:50 a3
-rw-rw-r-- 1 mc mc  0 Sep  3 00:18 a3.txt
-rw-rw-r-- 1 mc mc 17 Sep  3 00:11 abc
-rw-rw-r-- 1 mc mc 11 Sep  3 00:01 chapter1
-rw-rw-r-- 1 mc mc 11 Sep  3 00:03 chapter2
-rw-rw-r-- 1 mc mc 45 Sep  2 23:18 combine.txt
-rw-rw-r-- 1 mc mc 23 Sep  2 23:29 xaa
-rw-rw-r-- 1 mc mc 17 Sep  3 00:11 xz
```

7. head

```
(mc@kali)~[/OSY]
└─$ cat > a2
Hello
Mohit
Atharva
Shravan
Neel
Hitesh
Shelar
Parth
Abhishek
Vedant
Rane
Pranit
^C

(mc@kali)~[/OSY]
└─$ head a2
Hello
Mohit
Atharva
Shravan
Neel
Hitesh
Shelar
Parth
Abhishek
Vedant
```

8. Touch

```
(mc@kali)~[/OSY]
└─$ ls
a1 a1cp.txt a2 a2.txt a3 a3.txt abc chapter1 chapter2 combine.txt dte xaa xz
```

2. Write command to display prompt before copy the content of one file to another

→

```
(mc@kali)-[~/0SY]
$ cat chapter1 chapter2
22203A0029
22203A0012

(mc@kali)-[~/0SY]
$ cp -i chapter1 chapter2
cp: overwrite 'chapter2'? yes
```

3. Explain the different use of cat command with example

→

The **cat** command is one of the most useful commands in Linux – it is used for displaying, combining, and manipulating text files.

1. for creating new file:

```
(mc@kali)-[~/OSY]
$ cat > names
Mohit
Atharva
Shravan
```

2. For displaying content

```
(mc@kali)-[~/OSY]
$ cat names
Mohit
Atharva
Shravan
```

3. For combining files

```
(mc@kali)-[~/OSY]
$ cat names surname > combined1.txt

(mc@kali)-[~/OSY]
$ cat combined1.txt
Mohit
Atharva
Shravan
Chaudhari
Jadhav
Salgaonkar
```

4. Append content to an existing file :

```
(mc@kali)-[~/OSY]
$ cat surname >> name

(mc@kali)-[~/OSY]
$ cat name
Chaudhari
Jadhav
Salgaonkar
```