PRACTICAL-1

1. **TOUCH** :- 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.

Syntax: touch<filename> | e.g., touch abc1.txt

```
21012021003@telnetserver:~$ touch abc.txt
21012021003@telnetserver:~$ ls
abc1.txt  abc.txt
```

2. **CAT** :- Cat command is very frequently used in Linux. It reads data from the file and gives their content as output. It helps us to create, view, concatenate files.

Syntax: cat<filename> | e.g. cat abc.txt

```
21012021003@telnetserver:~$ cat abc.txt
hello this is amit
```

3. **LS** :- LS is a Linux shell command that lists directory contents of files and directories.

Syntax: we have to go in particular directory and have to print ls.

1. **LS** :- LS is a Linux shell command that lists directory contents of files and directories.

```
a  a1  a2  b  c  s1
```

2. ls –a :

```
.            19012011117  19012021080
..           19012011118  19012021081
18012011074  19012011119  19012021082
18012011082  19012011120  19012021083
18012011084  19012011121  19012021084
19012011001  19012011122  19012021086
19012011002  19012011123  19012021087
19012011003  19012011124  19012021088
19012011004  19012011125  19012021089
```

3. ls -A :

```
18012011074  19012011119  19012021082
18012011082  19012011120  19012021083
18012011084  19012011121  19012021084
19012011001  19012011122  19012021086
19012011002  19012011123  19012021087
19012011003  19012011124  19012021088
```

4. ls –c:

```
20012011187  21012011142  20012531027
21012021037  20012011014  20012531036
21012021016  21012021011  20012531001
21012021006  22172012045  20012531016
21012021101  21012011040  20012531033
```

5. ls -C:

```
18012011074  19012011119  19012021082
18012011082  19012011120  19012021083
18012011084  19012011121  19012021084
19012011001  19012011122  19012021086
19012011002  19012011123  19012021087
19012011003  19012011124  19012021088
19012011004  19012011125  19012021089
```

6. Ls –f:

```
.            20012531019  22172012032
..           21012011103  19012021054
22172022008  19012021030  20012011125
21012011014  20012021035  19012011034
20012531013  20012011133  20012021010
19012011072  19012021094  19012011080
20012531031  19012021002  22172012009
20012011180  20012011129  22012572006
```

7. Ls –F:

```
18012011074/  19012011128/  19012021101/
18012011082/  19012011129/  19012021102/
18012011084/  19012011130/  19012021103/
19012011001/  19012011131/  19012021104/
19012011002/  19012011135/  19012021105/
19012011003/  19012011136/  19012021106/
19012011004/  19012011137/  19012021107/
19012011005/  19012011138/  19012531001/
```

8. Ls –g:

```
total 4956
drwxr-xr-x  5 18012011074  4096 Nov 28 13:20 18012011074
drwxr-xr-x  4 18012011082  4096 Nov 25 14:15 18012011082
drwxr-xr-x  4 18012011084  4096 Jul 29  2022 18012011084
drwxr-xr-x  5 19012011001  4096 Nov 26 14:57 19012011001
drwxr-xr-x 15 19012011002  4096 Nov 26 14:11 19012011002
```

9. Ls -i:

```
3276814 18012011074  3277711 19012021046
3276822 18012011082  3277715 19012021047
3276826 18012011084  3277719 19012021048
3276816 19012011001  3277723 19012021050
3276834 19012011002  3277727 19012021051
3276838 19012011003  3277731 19012021052
3276842 19012011004  3277735 19012021053
3276846 19012011005  3277739 19012021054
3276831 19012011007  3277743 19012021056
3276859 19012011008  3277747 19012021057
```

10. Ls -l:-

```
total 4956
drwxr-xr-x  5 18012011074  18012011074  4096 Nov 28 13:20 18012011074
drwxr-xr-x  4 18012011082  18012011082  4096 Nov 25 14:15 18012011082
drwxr-xr-x  4 18012011084  18012011084  4096 Jul 29  2022 18012011084
drwxr-xr-x  5 19012011001  19012011001  4096 Nov 26 14:57 19012011001
drwxr-xr-x 15 19012011002  19012011002  4096 Nov 26 14:11 19012011002
drwxr-xr-x  6 19012011003  19012011003  4096 Nov 25 14:42 19012011003
drwxr-xr-x  5 19012011004  19012011004  4096 Nov 26 15:02 19012011004
```

11. Ls -L:-

```
18012011074  19012011119  19012021082  20012011063
18012011082  19012011120  19012021083  20012011065
18012011084  19012011121  19012021084  20012011066
19012011001  19012011122  19012021086  20012011067
19012011002  19012011123  19012021087  20012011068
```

12. Ls -m:-

```
18012011074, 18012011082, 18012011084,
19012011011, 19012011012, 19012011013,
19012011023, 19012011024, 19012011025,
19012011036, 19012011038, 19012011039,
19012011053, 19012011054, 19012011055,
19012011066, 19012011067, 19012011068,
19012011079, 19012011080, 19012011082,
19012011094, 19012011095, 19012011096,
19012011109, 19012011111, 19012011113,
19012011123, 19012011124, 19012011125,
```

13. Ls -n:-

```
total 4956
drwxr-xr-x  5 1001 1001 4096 Nov 28 13:20 18012011074
drwxr-xr-x  4 1002 1002 4096 Nov 25 14:15 18012011082
drwxr-xr-x  4 1003 1003 4096 Jul 29  2022 18012011084
drwxr-xr-x  5 1004 1004 4096 Nov 26 14:57 19012011001
drwxr-xr-x 15 1005 1005 4096 Nov 26 14:11 19012011002
```

14. Ls -o:-

```
total 4956
drwxr-xr-x  5 18012011074  4096 Nov 28 13:20 18012011074
drwxr-xr-x  4 18012011082  4096 Nov 25 14:15 18012011082
drwxr-xr-x  4 18012011084  4096 Jul 29  2022 18012011084
drwxr-xr-x  5 19012011001  4096 Nov 26 14:57 19012011001
```

15. Ls -p:-

```
18012011074/  19012011128/  19012021101/
18012011082/  19012011129/  19012021102/
18012011084/  19012011130/  19012021103/
19012011001/  19012011131/  19012021104/
19012011002/  19012011135/  19012021105/
19012011003/  19012011136/  19012021106/
19012011004/  19012011137/  19012021107/
19012011005/  19012011138/  19012531001/
19012011007/  19012011139/  19012531002/
```

16. Ls -r:-

```
ucp             22012022004  21012571037
telnet          22012022003  21012571035
t22014021002    22012022002  21012571034
t22014021001    22012022001  21012571033
t22014011005    22012012040  21012571032
t22014011004    22012012039  21012571031
t22014011003    22012012038  21012571030
t22014011002    22012012037  21012571029
```

17. Ls -R:-

```
1  2  3  file1  file2  hello  hello.txt  kashish  newdir  pra-2_1.txt  pra-2_3.txt

./21012021118/hello:

./21012021118/hello.txt:
```

18. Ls -s:-

```
total 4956
4 18012011074  4 19012011143  4 19012531017
4 18012011082  4 19012011144  4 19012531018
4 18012011084  4 19012011145  4 19012531019
4 19012011001  4 19012011146  4 19012531020
4 19012011002  4 19012011147  4 19012531021
4 19012011003  4 19012011148  4 20001201163
4 19012011004  4 19012011149  4 20012011001
```

19. Ls -t:-

```
20012011187  20012011048  20012531013
22012532005  21012022022  20012531018
21012021001  21012011142  20012531027
21012021038  20012011014  20012531036
21012021012  21012021011  20012531001
21012021006  22172012045  20012531016
```

20. Ls -u:-

```
22012532001  21012021108  22012012009
21012021001  21012021087  bhs
21012021017  20012011048  19012011024
```

21. Ls -x:-

```
18012011074   18012011082   18012011084
19012011010   19012011011   19012011012
19012011021   19012011022   19012011023
19012011032   19012011034   19012011035
19012011049   19012011050   19012011051
```

22. **MKDIR** :- MKDIR command in Linux allows the user to create directories.

Syntax: mkdir<dirname>|e.g. mkdir f1

```
21012021003@telnetserver:~$ mkdir A1
21012021003@telnetserver:~$ ls
A1  abc1.txt  abc.txt
```

23. **RMDIR** :- RMDIR command is used remove empty directories from the file system in Linux.

Syntax: rmdir<dirname>|e.g. rmdir f1

```
21012021003@telnetserver:~$ rmdir A1
21012021003@telnetserver:~$ ls
abc1.txt  abc.txt
```

24. **CD** :- CD command in linux known as change directory command. It is used to change current working directory.

Syntax: cd<dirname> | e.g. cd f2 // cd path

```
21012021003@telnetserver:~$ cd A1
21012021003@telnetserver:~/A1$
```

25. **CLEAR** :- clear is a standard Unix computer operating system command that is used to clear the terminal screen.i

Syntax: clear

```
21012021003@telnetserver:~/A1$
```

26. **CP** :- cp stands for copy. This command is used to copy files or group of files or directory. It creates an exact image of a file on a disk with different file name. cp command require at least two filenames in its arguments.

Syntax: cp <existing file name> <new file name> |e.g. cp f1.txt f2.txt

```
21012021003@telnetserver:~$ cp abc.txt abc1.txt
21012021003@telnetserver:~$ ls
A1  abc1.txt  abc.txt
21012021003@telnetserver:~$ ls abc1.txt
abc1.txt
21012021003@telnetserver:~$ cat abc1.txt
hello this is amit
```

27. **CAL** :- cal command is a calendar command in Linux which is used to see the calendar of a specific month or a whole year.

Syntax: cal [[month]year] | e.g. cal [[January]2020]

```
21012021003@telnetserver:~$ cal
    January 2023
Su Mo Tu We Th Fr Sa
 1  2  3  4  5  6  7
 8  9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30 31
```

28. **HISTORY** :- history command is used to view the previously executed command.

Syntax: history

```
21012021003@telnetserver:~$ history
    1  spell
    2  echo
    3  echo hello
    4  finger amit
    5  ls -A
    6  ls c
    7  ls -c
    8  ls - d
    9  ls -d
   10  ls -F
   11  touch -21012021003
   12  touch --help
   13  mkdir
```

29. **CHMOD** :- In Unix-like operating systems, the chmod command is used to change the access mode of a file.

Syntax: chmod [reference][operator][mode] file

```
21012021003@telnetserver:~$ chmod -rwx A1
21012021003@telnetserver:~$ ls
A1  abc1.txt  abc.txt
21012021003@telnetserver:~$ ls -l
total 12
d--------- 3 21012021003 21012021003 4096 Jan 31 14:14 A1
-rw-rw-r-- 1 21012021003 21012021003   19 Jan 31 14:19 abc1.txt
-rw-rw-r-- 1 21012021003 21012021003   19 Jan 31 14:17 abc.txt
21012021003@telnetserver:~$ chmod +rx A1
21012021003@telnetserver:~$ ls -l
total 12
dr-xr-xr-x 3 21012021003 21012021003 4096 Jan 31 14:14 A1
-rw-rw-r-- 1 21012021003 21012021003   19 Jan 31 14:19 abc1.txt
-rw-rw-r-- 1 21012021003 21012021003   19 Jan 31 14:17 abc.txt
21012021003@telnetserver:~$ chmod o-rwx A1
21012021003@telnetserver:~$ ls -l
total 12
dr-xr-x--- 3 21012021003 21012021003 4096 Jan 31 14:14 A1
-rw-rw-r-- 1 21012021003 21012021003   19 Jan 31 14:19 abc1.txt
-rw-rw-r-- 1 21012021003 21012021003   19 Jan 31 14:17 abc.txt
```

30. **HEAD** :- The head command, as the name implies, print the top N number of data of the given input. By default, it prints the first 10 lines of the specified files. If more than one file name is provided then data from each file is preceded by its file name.

Syntax: head [command] <filename>

```
21012021003@telnetserver:~$ head abc.txt
hello this is amit
```

31. **TAIL** :- The tail command, as the name implies, print the last N number of data of the given input. By default it prints the last 10 lines of the specified files. If more than one file name is provided then data from each file is precedes by its file name.

Syntax: tail [option] <filename> | e.g. tail abc.txt

```
21012021003@telnetserver:~$ tail abc1.txt
hello this is amit
```

32. **DATE** :- date command is used to display the system date and time. date command is also used to set date and time of the system.

Syntax: date [option] [format] | e.g. date "10 days ago"

```
21012021003@telnetserver:~$ date
Tue Jan 31 14:31:31 IST 2023
```

33. **EXPR** :- The **expr** command in Unix evaluates a given expression and displays its corresponding output. Basic operations like addition, subtraction, multiplication, division, and modulus on integers.

Syntax: expr expression | e.g. expr --version

```
21012021003@telnetserver:~$ expr --version
expr (GNU coreutils) 8.28
Copyright (C) 2017 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Written by Mike Parker, James Youngman, and Paul Eggert.
```

34. **WHO** :- The who command is used to get information about currently logged in user on to system.

Syntax: who [option] <filename>

```
21012021003@telnetserver:~$ who
21012021068 pts/10       2023-01-31 13:36 (49.34.218.255)
20012011187 pts/9        2023-01-31 13:08 (192.168.25.138)
21012011171 pts/2        2023-01-31 13:50 (192.168.26.245)
21012021011 pts/18       2023-01-31 14:05 (192.168.26.232)
21012021012 pts/19       2023-01-31 13:58 (192.168.26.242)
21012021064 pts/14       2023-01-31 13:47 (49.34.104.229)
22012012024 pts/21       2023-01-31 14:10 (192.168.26.246)
21012021001 pts/25       2023-01-31 14:05 (192.168.26.60)
21012021016 pts/28       2023-01-31 13:56 (192.168.12.87)
21012021015 pts/26       2023-01-31 13:56 (192.168.26.238)
21012021064 pts/23       2023-01-31 14:01 (49.34.104.229)
21012021006 pts/29       2023-01-31 13:57 (192.168.26.232)
```

35. **UNAME** :- The command uname displays the information about the system.

Syntax: uname [option]

```
21012021003@telnetserver:~$ uname
Linux
```

36. **FINGER** :- Finger command is a user information lookup command which gives details of all the users logged in. This tool is generally used by system administrators. It provides details like login name, user name, idle time, login time, and in some cases their email address even.

Syntax: finger [option] username

```
21012021003@telnetserver:~$ finger 21012021003

Command 'finger' not found, but can be installed with:

apt install finger
Please ask your administrator.
```

37. **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.

Syntax: cmp <filename1> <filename2>

38. **COMM** :- comm compare two sorted files line by line and write to standard output; the lines that are common and the lines that are unique.

Syntax: comm <filename1> <filename2>

```
21012021003@telnetserver:~$ comm abc.txt abc1.txt
                hello this is amit
```

39. **SORT** :- SORT command is used to sort a file, arranging the records in a particular order. By default, the sort command sorts file assuming the contents are ASCII. Using options in the sort command can also be used to sort numerically.

Syntax: sort <filename>

```
21012021003@telnetserver:~$ sort abc.txt
hello this is amit
```

40. **SPELL** :- Spell command is used as a spell checker in Linux. Generally, it will scan the given files or anything from standard input then it check for misspellings. Finally it allows the user to correct the words interactively.

Syntax: spell [option] <filename>

```
21012021003@telnetserver:~$ spell abc.txt

Command 'spell' not found, but can be installed with:

apt install spell
Please ask your administrator.
```

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

Syntax: wc <filename>

```
21012021003@telnetserver:~$ wc abc.txt
 1  4 19 abc.txt
```

42. **TYPE** :- The type command is used to describe how its argument would be translated if used as commands. It is also used to find out whether it is built-in or external binary file.

Syntax: type [command name]

43. **ECHO** :- Echo command in linux is used to display line of text/string that are passed as an argument . This is a built in command that is mostly used in shell scripts and batch files to output status text to the screen or a file.

Syntax: echo [string] | e.g. echo “hello world”

```
21012021003@telnetserver:~$ echo "hello world"
hello world
```

44. **MAN** :- Man command in Linux is used to display the user manual of any command that we can run on the terminal. It provides a detailed view of the command which includes NAME, SYNOPSIS, DESCRIPTION, OPTIONS, EXIT STATUS, RETURN VALUES, ERRORS, FILES, VERSIONS.

Syntax: man [command name] | e.g. man printf

```
21012021003@telnetserver:~$ man printf
PRINTF(1)                       User Commands                       PRINTF(1)

NAME
       printf - format and print data

SYNOPSIS
       printf FORMAT [ARGUMENT]...
       printf OPTION

DESCRIPTION
       Print ARGUMENT(s) according to FORMAT, or execute according to OPTION:

       --help display this help and exit

       --version
              output version information and exit

       FORMAT controls the output as in C printf.  Interpreted sequences are:

       \"     double quote

       \\     backslash
```

45. **MORE** :- 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.

Syntax: more[option] filename | e.g. more -d ab.txt

```
21012021003@telnetserver:~$ more -d abc.txt
hello this is amit
```

46. **PASSWD** :- passwd command in Linux is used to change the user account passwords.

Syntax: passwd [option] [username]

```
21012021003@telnetserver:~$ passwd 21012021003
Changing password for 21012021003.
```

47. **PWD** :- pwd stands for **P**rint **W**orking **D**irectory. It prints the path of the working directory, starting from the root.

Syntax: pwd

```
21012021003@telnetserver:~$ pwd
/home/21012021003
```

48. **GREP** :- The grep filter searches a file for a particular pattern of characters, and displays all lines that contain that pattern.

Syntax: grep [option] pattern filename | e.g. gerp -l “unix” f1.txt f2.txt

49. **PS** :- Linux provides us a utility called ps for viewing information related with the processes on a system which stands as abbreviation for **“Process Status”.** ps command is used to list the currently running processes and their PIDs along with some other information depends on different options.

Syntax: pwd -p/-l

```
21012021003@telnetserver:~$ pwd -p/-l
-bash: pwd: -p: invalid option
pwd: usage: pwd [-LP]
```

50. **RM** :- rm stands for remove here. rm command is used to remove objects such as files, directories, symbolic links and so on from the file system like UNIX.

Syntax: rm <filename> | e.g. rm b.txt

51. **SET** :- Linux set command is used to set and unset certain flags or settings within the shell environment.

Syntax: set[options]

52. **CUT** :- The cut command in UNIX is a command for cutting out the sections from each line of files and writing the result to standard output.

Syntax: cut[option] filename |e.g. cut abc.txt

53. **READ** :- read command in Linux system is used to read from a file descriptor. Basically, this command read up the total number of bytes from the specified file descriptor into the buffer. If the number or count is zero then this command may detect the errors. But on success, it returns the number of bytes read.

Syntax: read text | e.g. read Hello World

```
21012021003@telnetserver:~$ read
Hello WOrld
```

54. **AWK** :- Awk is a scripting language used for manipulating data and generating reports. The awk command programming language requires no compiling and allows the user to use variables, numeric functions, string functions, and logical operators.

Syntax: awk option inputfilename>outputfilename |e.g. awk -f ab1.txt ab2.txt

55. **LN** :- The ln command is used to create links between files.

Syntax: ln [option]... target... directory

56. **KILL** :- kill command in Linux (located in /bin/kill), is a built-in command which is used to terminate processes manually. kill command sends a signal to a process which terminates the process.

Syntax: kill -l

```
21012021003@telnetserver:~$ kill -l
 1) SIGHUP       2) SIGINT       3) SIGQUIT      4) SIGILL       5) SIGTRAP
 6) SIGABRT      7) SIGBUS       8) SIGFPE       9) SIGKILL     10) SIGUSR1
11) SIGSEGV     12) SIGUSR2     13) SIGPIPE     14) SIGALRM     15) SIGTERM
16) SIGSTKFLT   17) SIGCHLD     18) SIGCONT     19) SIGSTOP     20) SIGTSTP
21) SIGTTIN     22) SIGTTOU     23) SIGURG      24) SIGXCPU     25) SIGXFSZ
26) SIGVTALRM   27) SIGPROF     28) SIGWINCH    29) SIGIO       30) SIGPWR
31) SIGSYS      34) SIGRTMIN    35) SIGRTMIN+1  36) SIGRTMIN+2  37) SIGRTMIN+3
38) SIGRTMIN+4  39) SIGRTMIN+5  40) SIGRTMIN+6  41) SIGRTMIN+7  42) SIGRTMIN+8
43) SIGRTMIN+9  44) SIGRTMIN+10 45) SIGRTMIN+11 46) SIGRTMIN+12 47) SIGRTMIN+13
48) SIGRTMIN+14 49) SIGRTMIN+15 50) SIGRTMAX-14 51) SIGRTMAX-13 52) SIGRTMAX-12
53) SIGRTMAX-11 54) SIGRTMAX-10 55) SIGRTMAX-9  56) SIGRTMAX-8  57) SIGRTMAX-7
58) SIGRTMAX-6  59) SIGRTMAX-5  60) SIGRTMAX-4  61) SIGRTMAX-3  62) SIGRTMAX-2
63) SIGRTMAX-1  64) SIGRTMAX
```

57. **FIND** :- The find command in UNIX is a command line utility for walking a file hierarchy. It can be used to find files and directories and perform subsequent operations on them.

Syntax: find [where to start searching from] [expression determines what to find] [-options] [what to find]

```
21012021003@telnetserver:~$ find
.
./.gnupg
./.gnupg/private-keys-v1.d
./.bash_logout
./.bashrc
./.cache
./.cache/motd.legal-displayed
./abc.txt
./A1
./A1/abc.txt
./.profile
./.bash_history
```

58. **INFO** :- info command reads documentation in the info format. It will give detailed information for a command when compared with the man page.

Syntax: info [OPTION]... [MENU-ITEM...] | e.g. info -a cvs