Linux Tutorial

PART-1

CALENDAR:

ncal → calendar of that month

```
root@Goosari:~# ncal
    February 2025
Su      2  9 16 23
Mo      3 10 17 24
Tu      4 11 18 25
We      5 12 19 26
Th      6 13 20 27
Fr      7 14 21 28
Sa  1   8 15 22
root@Goosari:~#
```

ncal -y → calendar of the whole year

ncal 2002 -m 5 → may 2002

```
root@Goosari:~# cal 12 2017
```

date → shows the day, month, day, time, time zone, year

```
root@Goosari:~# date
Sun Feb 23 01:25:17 IST 2025
```

PRINTING:

echo → print the text

```
root@Goosari:~# echo hello
hello
```

HISTORY:

history → print the previous commands

history -c; history -w → clear the history and write history

CLEAR AND EXIT:

exit → close the terminal or **Ctrl+d**

clear → clear the terminal or use **Ctrl+l**

FORAMT OF A LINUX COMMAND:

```
root@Goosari:~# commandname options inputs
```

PATH:

echo $PATH → find where the command is → it searches from the left to the right

which <filename> → will find the executable file that would be run when you write a command in terminal

```
root@Goosari:~# which ncal
/usr/bin/ncal
```

whereis <filename> → will find the executables, source code, and manual pages.

```
root@Goosari:~# whereis ncal
ncal: /usr/bin/ncal /usr/share/man/man1/ncal.1.gz
```

MAN PAGE: Manual page

1. **User commands** → commands that can be run from the shell by normal user
2. **System calls** → low-level functions that provide an interface between a user program and the OS (fork, exec, open, read, write, close)
3. **C library functions** → functions available in the standard C lib (printf, scanf, malloc, sin)
4. **Devices and special file** → Describes device files which represent h/w devices and other special files within the filesystem
5. **File formats and conventions** → the structure and format of the types of specific configuration files (tar, crontab, ,etc.)
6. **Games** → document games and other recreational programs
7. **Miscellaneous** → information that doesn't fit in the other categories
8. **System administration** → commands that are typically used by system administrators to manage and maintain the system (useradd, apt-get, ifconfig, fdisk)

- 1, 5 and 8 mainly used

man -k which → gives the list of commands (-k → keyword search) and sections which contain the work which

```
root@Goosari:~# man -k which
getcpu (2)           - determine CPU and NUMA node on which the calling thread is running
getgrouplist (3)     - get list of groups to which a user belongs
IO::AtomicFile (3pm) - write a file which is updated atomically
lcf (1)              - Determine which of the historical versions of a config is installed
pam_exec (8)         - PAM module which calls an external command
pam_warn (8)         - PAM module which logs all PAM items if called
sched_getcpu (3)     - determine CPU on which the calling thread is running
securetty (5)        - list of terminals on which root is allowed to login
URI::WithBase (3pm)  - URIs which remember their base
which (1)            - locate a command
```

man 8 pam-exec → will open man page of that section command

man which → open up the section 1 of the which user command

```
WHICH(1)                                General Commands Manual                                WHICH(1)

NAME
       which - locate a command

SYNOPSIS
       which [-as] filename ...

DESCRIPTION
       which  returns  the  pathnames of the files (or links) which would be executed in the current environ-
       ment, had its arguments been given as commands in a strictly POSIX-conformant shell.  It does this  by
       searching the PATH for executable files matching the names of the arguments.  It does not canonicalize
       path names.

OPTIONS
       -a     print all matching pathnames of each argument

       -s     silently return 0 if all of the executables were found or 1 otherwise

EXIT STATUS
       0      if all specified commands are found and executable

       1      if one or more specified commands is nonexistent or not executable

       2      if an invalid option is specified

Debian                                        29 Jun 2016                                       WHICH(1)
 Manual page which(1) line 1/27 (END) (press h for help or q to quit)
```

which [-as] filename … → if something is in [] it means its optional and if there is … it means you can have multiple filename (input)

```
root@Goosari:~# which [-a | -f] <SOMETHING>
```

<SOMETHING> → mean it is a requirement (i.e. have to)

| → have to use one or the other

```
root@Goosari:~# which -a date cal
/usr/bin/date
/bin/date
/usr/bin/cal
/bin/cal
```

Section	Meaning
[Thing]	Thing is optional
<Thing>	Thing is mandatory
Thing …	Thing can be repeated (limitless)
Thing1 | Thing2	Use Thing1 or Thing 2
Thing	Replace *Thing* with whatever appropriate

HOW TO FIND A COMMAND NEEDED USING MAN:

1. **Man -k 'List directory contents'** → this will keywork search 'list directory contents' all together and list the appropriate commands
2. If I want to search ls then use '**man ls**' to open the manual page

```
root@Goosari:~# man -k 'list directory contents'
dir (1)              - list directory contents
ls (1)               - list directory contents
vdir (1)             - list directory contents
root@Goosari:~# man ls
```

3. In the man to search option use '**/-[option]**'

```
REPORTING BUGS
       GNU coreutils online help: <https://www.gnu.org/software/coreutils/>
       Report any translation bugs to <https://translationproject.org/team/>
/-L
```

help <command> → similar to manual page but print it in the terminal (does not open a separate page)

```
root@Goosari:~# help pwd
pwd: pwd [-LP]
    Print the name of the current working directory.

    Options:
      -L        print the value of $PWD if it names the current working
                directory
      -P        print the physical directory, without any symbolic links

    By default, `pwd' behaves as if `-L' were specified.

    Exit Status:
    Returns 0 unless an invalid option is given or the current directory
    cannot be read.
```

INPUT AND OUTPUT:

```
root@Goosari:~# name="Khushi"
root@Goosari:~# echo "Hello, $name"
Hello, Khushi
root@Goosari:~#
```

- Do not use space while initializing a variable

- Redirection and piping are powerful mechanisms for manipulating input and output streams of commands.

REDIRECTION → allows you to change where a command's input comes from or where output goes (rerouting the flow of data).

Input redirection: (<)

```
root@Goosari:~# command < input_file.txt
```

Output redirection: (>) → truncate the text

```
root@Goosari:~# command > output.txt
```

Append Output Redirection: (>>) → doesn't truncate the text

```
root@Goosari:~# command >> output.txt
```

Standard error redirecting: (2>)

```
root@Goosari:~# cat -k bla 2> error.txt
root@Goosari:~# cat error.txt
cat: invalid option -- 'k'
Try 'cat --help' for more information.
root@Goosari:~#
```

Example:

```
root@Goosari:/mnt/c/Users/khush/Desktop/Linux_tutorial# cat < input.txt >> output.txt 2> error.txt
```

tty → tells us the name of the current terminal

```
root@Goosari:/mnt/c/Users/khush/Desktop/Linux_tutorial# tty
/dev/pts/2
```

In a new terminal:

```
root@Goosari:/mnt/c/Users/khush/Desktop/Linux_tutorial# cat < input.txt > /dev/pts/2
root@Goosari:/mnt/c/Users/khush/Desktop/Linux_tutorial#
```

Will print the output in the old terminal

```
root@Goosari:/mnt/c/Users/khush/Desktop/Linux_tutorial# Khushi Goosari
cbaiuiefbie
```

PIPING → " | " connect output of one command to the input of another command. The output of the first command becomes the input for the second command and so on.

CUT:

cut → cuts the input [-d → delimiter] [-f → field]

```
root@Goosari:~/Linux_tutorial# cut < date.txt -d " " -f 1
Sun
```

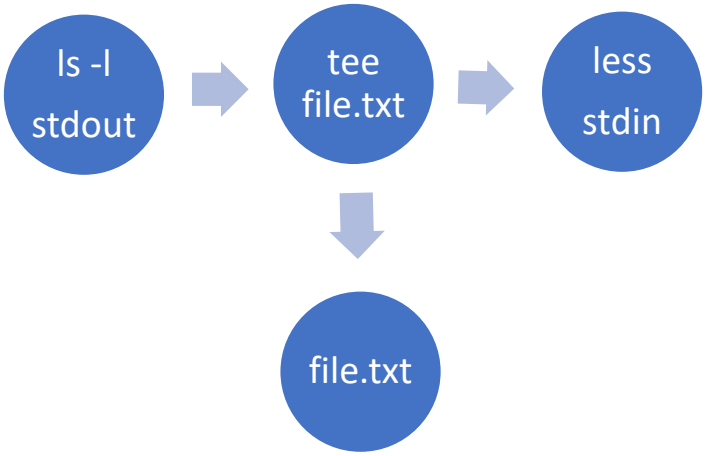
Options	Description
-b	Bytes
-c	Characters
-d	Delimiter
-s	Do not print lines not containing delimiters
-z	Line delimiter is null, not new line
-n	Ignored

With piping:

```
root@Goosari:~/Linux_tutorial# date | cut -d " " -f 1
Sun
```

tee <filename> → copy standard input to each file and also to standard output

```
root@Goosari:~/Linux_tutorial# ls -l | tee file.txt | less
```



- The data flows in, and then it's split, with one copy going to the standard output and another copy going to a file.

Options	Description
-a	Append
-i	ignore interrupts
-p	Operate in more appropriate mode with pipes
--output-error[=MODE]	Set behaviour on write error

```
root@Goosari:~/Linux_tutorial# date | tee fulldate.txt | cut -d " " -f 1
Sun
root@Goosari:~/Linux_tutorial# cat fulldate.txt
Sun Feb 23 20:47:24 IST 2025
```

xargs → It takes input (often a list of items) and converts it into arguments for another command.

```
root@Goosari:~/Linux_tutorial# date | xargs echo
Sun Feb 23 21:04:18 IST 2025
root@Goosari:~/Linux_tutorial#
```

```
root@Goosari:~/Linux_tutorial# ls -lrt
total 12
-rw-r--r-- 1 root root 262 Feb 23 18:36 output.txt
-rw-r--r-- 1 root root  29 Feb 23 19:03 date.txt
-rw-r--r-- 1 root root  20 Feb 23 21:09 fileocode.txt
root@Goosari:~/Linux_tutorial# cat fileocode.txt | xargs rm
root@Goosari:~/Linux_tutorial# ls -lrt
total 4
-rw-r--r-- 1 root root 20 Feb 23 21:09 fileocode.txt
```

REMOVE FILE OR DIRECTORY:

rm <filename> → remove empty file

rm -rf <filename> → remove file recursively and forcefully

rmdir <filename> → remove empty directory

ALIASES:

- Create a **bash** file.
  - A bash file is a script containing a sequence of commands that are executed by the Bash shell.

```
root@Goosari:~/Linux_tutorial# touch ~/.bash_aliases
```

- Write using **alias** keyword

```
alias getdates='date |  tee /root/Linux_tutorial/fulldate.txt | cut -d " " -f 1 | tee /root/Linux_tutorial/short
date.txt | xargs echo hello'
```

- Reset the bash file

```
root@Goosari:~/Linux_tutorial# source ~/.bash_aliases
```

- Creates 2 files name fulldate.txt, shortdate.txt and echo the text.

```
root@Goosari:~/Linux_tutorial# getdates
hello Sun
root@Goosari:~/Linux_tutorial# ls -lrt
total 12
-rw-r--r-- 1 root root 20 Feb 23 21:09 fileocode.txt
-rw-r--r-- 1 root root  4 Feb 23 21:40 shortdate.txt
-rw-r--r-- 1 root root 29 Feb 23 21:40 fulldate.txt
root@Goosari:~/Linux_tutorial# cat fulldate.txt
Sun Feb 23 21:40:16 IST 2025
root@Goosari:~/Linux_tutorial# cat shortdate.txt
Sun
```

Link to the commands cheat sheet:

- https://drive.google.com/file/d/1B0d8RlZpY_Qa5SZ_MgcCgI5HRnStx_Xc/view?usp=sharing