

Linux Commands

1. cd (Change Directory) Command

The cd command is used to change the current directory (i.e., the directory in which the user is currently working)

Syntax :

cd [-Options] [Directory]

Example :

Option	Use
cd ..	Change Current directory to parent directory
cd ~	Move to users home directory from anywhere
cd lab_1	Change from current working directory to lab_1
cd ../downloads	If we are currently in /home/username/documents then we would be placed in /home/username/downloads.

The screenshot shows a terminal window with a dark background and light-colored text. It displays the following session:

```
~/test
└─ pwd
/home/cyrixninja/test

~/test
└─ cd ..
~
└─ cd test
~/test
```

The terminal shows the user navigating through directories. The first command, `pwd`, prints the full path. The `cd ..` command moves up one level to the root directory. The final `cd test` command moves back into the `test` directory, returning the user to their original starting point.

2. ls Command

List directory contents.

Syntax :

ls [Options] [file|dir]

Example :

Option	Use
ls -l	To show long listing information about the file/directory
ls -a	List all files including hidden file starting with ''
ls -r	List in reverse order
ls -t	Sort by time & date
ls -s	Sort by file size

```
~ 
ls -l
total 0
drwxr-xr-x. 1 cyrixninja cyrixninja   6 Jan 13 13:09 Android
drwxr-xr-x. 1 cyrixninja cyrixninja 142 Feb 23 15:25 AppImages
drwxr-xr-x. 1 cyrixninja cyrixninja    0 Dec 22 19:46 Desktop
drwxr-xr-x. 1 cyrixninja cyrixninja 170 Mar 11 18:32 Documents
drwxr-xr-x. 1 cyrixninja cyrixninja    0 Mar 12 15:06 Downloads
drwxr-xr-x. 1 cyrixninja cyrixninja   12 Jan  1 10:03 go
drwxr-xr-x. 1 cyrixninja cyrixninja    0 Dec 22 19:46 Music
drwxr-xr-x. 1 cyrixninja cyrixninja   42 Feb 22 14:57 Pictures
drwxr-xr-x. 1 cyrixninja cyrixninja    0 Dec 22 19:46 Public
drwxr-xr-x. 1 cyrixninja cyrixninja    0 Dec 22 19:46 Templates
drwxr-xr-x. 1 cyrixninja cyrixninja    0 Mar 12 15:01 test
drwxr-xr-x. 1 cyrixninja cyrixninja   18 Mar 11 14:41 Videos
```

```
~ 
ls -a
.               .dart-tool
..              Desktop
.android        Documents
Android         .dotnet
AppImages       Downloads
.bash_history   flutter
.bash_logout    flutter-devtools
.bash_profile   .ghidra
.bashrc          .gitconfig
.cache          .gnupg
.cargo          go
.cert           .gradle
.config         .ipython
.dart           .java
.dartServer     .jupyter
.local          .m2
.mongoDB        .mozilla
.Music          .npm
.nvm            .oh-my-zsh
Pictures        .p10k.zsh
.PKCS12          .profile
.pub-cache      .pki
.Public         .pub-cache
.python_history .profile
.zshrc          .python_history
```

3.man Command

It is the interface used to view the system's reference manuals.

Syntax :

man [command name]

Example:

```
man ls
LS(1)                               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 spec-
    ified.

    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 ..

Manual page ls(1) line 1 (press h for help or q to quit)
```

4.echo Command

Display a line of text/string on standard output or a file.

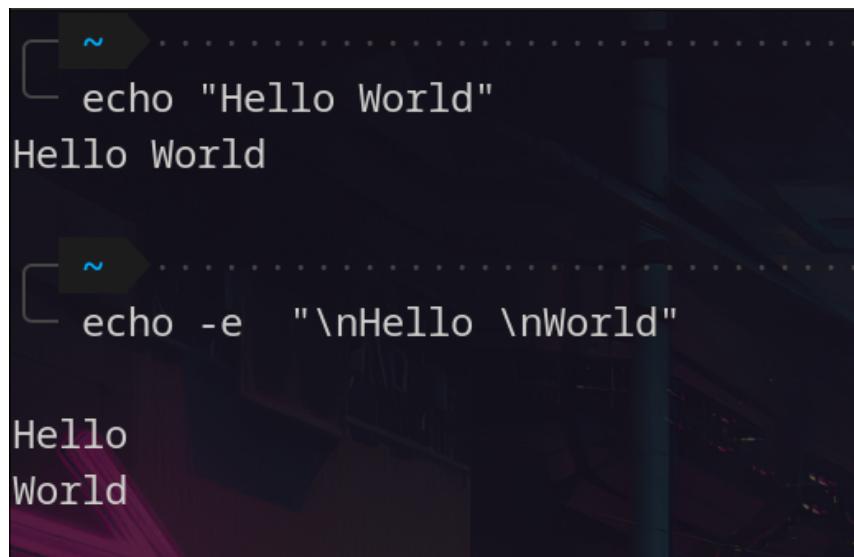
Syntax :

echo [option] [string]

Example :

Option	Use
echo -n	Do not output a trailing newline
echo -e	Enable interpretation of backslash escape sequences

Option	Use
\b	It removes all the spaces in between the text
\n	It creates new line from where it is used
\t	It creates horizontal tab spaces



```

~ echo "Hello World"
Hello World

~ echo -e "\nHello \nWorld"

Hello
World

```

5.cal Command

Displays a simple, formatted calendar in your terminal.

Syntax :

cal [options] [[[day] month] year]

Example :

Option	Use
cal -1	Display single month output
cal -3	Display three months spanning the date.
cal -s	Display Sunday as the first day of the week.
cal -m	Display Monday as the first day of the week.
cal -j	Use day-of-year numbering for all

	calendars. These are also called ordinal days. Ordinal days range from 1 to 366.
cal -y	Display a calendar for the whole year

```

cal -3
February 2024            March 2024            April 2024
Su Mo Tu We Th Fr Sa   Su Mo Tu We Th Fr Sa   Su Mo Tu We Th Fr Sa
1 2 3                   3 4 5 6 7 8 9          1 2 3 4 5 6
4 5 6 7 8 9 10        10 11 12 13 14 15 16    7 8 9 10 11 12 13
11 12 13 14 15 16 17  17 18 19 20 21 22 23  14 15 16 17 18 19 20
18 19 20 21 22 23 24  24 25 26 27 28 29 30  21 22 23 24 25 26 27
25 26 27 28 29         31

cal -s
March 2024
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

```

6.date Command

Print or set the system date and time.

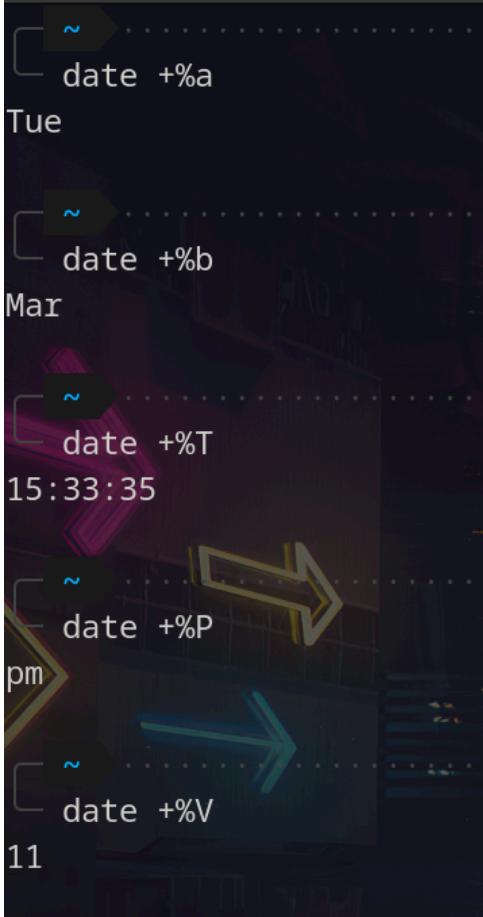
Syntax :

date [OPTION]... [+FORMAT]

Example :

Option	Use
date +%a	The abbreviated weekday name (e.g., Sun)
date +%A	The full weekday name (e.g., Sunday)
date +%b	The abbreviated month name (e.g., Jan)
date +%B	Locale's full month name

date +%C	The current century; like %Y, except omit last two digits (e.g., 20)
date +%w	day of week (0..6); 0 is Sunday
date +%d	Display the day of the month
date +%m	Displays the month of year (01 to 12)
date +%y	Displays last two digits of the year(00 to 99)
date +%Y	Display four-digit year.
date +%T	Display the time in 24 hour format as HH:MM:SS
date +%H	Display the hour
date +%M	Display the minute
date +%S	Display the seconds
date +%V	ISO week number, with Monday as first day of week (01..53)
date +%P	locale's equivalent of either AM or PM



A screenshot of a terminal window displaying several examples of the date command:

- date +%a: Tuesday
- date +%b: March
- date +%T: 15:33:35
- date +%P: pm
- date +%V: 11

7.clear Command

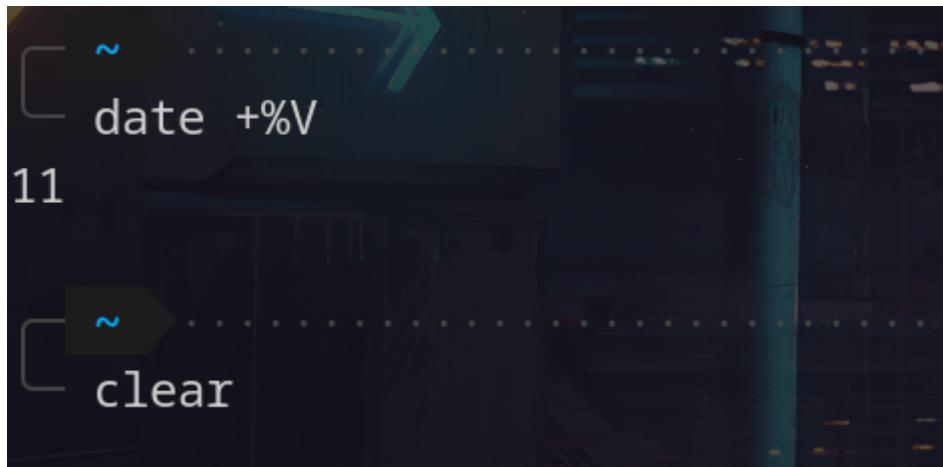
Clear the terminal screen.

If you take a detailed look after running the clear command, you'll find that it doesn't really clear the terminal. The tool just shifts the text upwards, out of the viewable area.

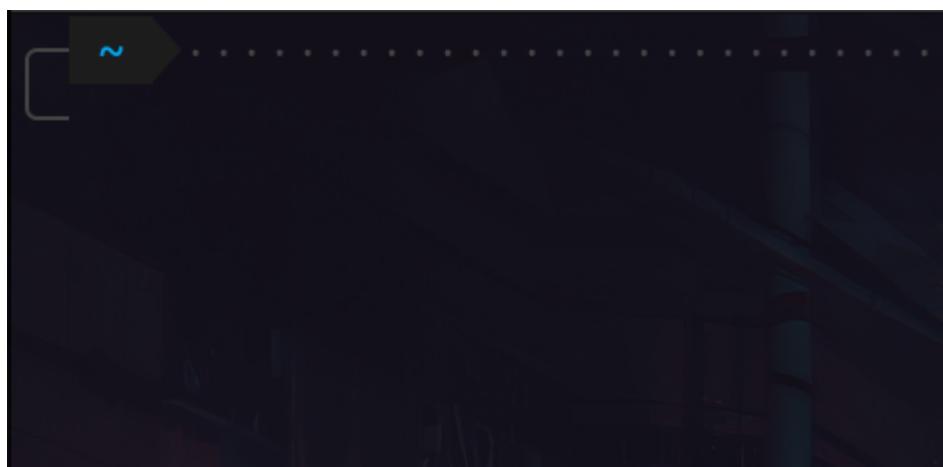
Syntax :

Clear

Example :



A screenshot of a terminal window. The first line shows the command 'date +%V' and its output '11'. The second line shows the command 'clear'.



8.cat Command

It is used to create, display and concatenate file contents.

Syntax :

cat [OPTION] [FILE]

Example :

Option	Use
cat -b	Omits line numbers for blank space in the output
cat -E	Displays a \$ (dollar sign) at the end of

	each line
cat -n	Line numbers for all the output lines
cat -s	Suppress repeated empty output lines
cat -T	Displays the tab characters as ^I in the output

```

~/test
└─ cat > hello.txt
Hello World
^C

~/test
└─ cat -E > world.txt
Hello
World
$ ^C
└─ ls
hello.txt  world.txt

```

9.pwd (Print working directory) Command

It prints the current working directory name with the complete path starting from root (/).

Syntax :

pwd [-OPTION]

Example :

```

~/test
└─ pwd
/home/cyrixninja/test

~/test
└─

```

10.who Command

It display the users that are currently logged into your Unix computer system.

Syntax :

who [-options] [filename]

Example :

Option	Use
who -b	Display the time of the last system boot
who -h	Print a line of column headings
who -q	Displays all login names, and a count of all logged-on users
who -a	Display all details of current logged in user

```
who -b
system boot 2024-03-12 14:21
at 15:45:06

who -q
cyrixninja cyrixninja
# users=2
at 15:45:10

who -a
system boot 2024-03-12 14:21
run-level 5 2024-03-12 14:21
cyrixninja ? seat0 2024-03-12 14:21 ?
cyrixninja + tty2 2024-03-12 14:21 01:24
pts/1 2024-03-12 14:22
at 15:45:16
2168 (login screen)
2168 (tty2)
4038 id=ts/1 term=0 exit=0
```

11.whoami Command

This command prints the username associated with the current effective user ID.

Syntax :

whoami [-OPTION]

Example :

Option	Use
whoami --help	Display a help message, and exit
whoami --version	Display version information, and exit

```
~ whoami
cyrixninja

~ whoami --help
Usage: whoami [OPTION]...
Print the user name associated with the current effective user ID.
Same as id -un.

--help      display this help and exit
--version   output version information and exit

GNU coreutils online help: <https://www.gnu.org/software/coreutils/>
Full documentation <https://www.gnu.org/software/coreutils/whoami>
or available locally via: info '(coreutils) whoami invocation'
```

```
~ whoami --version
whoami (GNU coreutils) 9.3
Copyright (C) 2023 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <https://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 Richard Mlynarik.
```

12.uname (unix name) Command

Print information about the current system.

Syntax :

uname [-OPTION]

Example :

Option	Use
uname -s	Print the kernel name
uname -n	Print the network node hostname

uname -v	Print the kernel version
uname -m	Print the machine hardware name
uname -o	Print the operating system

```

~ uname -o
GNU/Linux

~ uname -n
fedora

~ uname -v
#1 SMP PREEMPT_DYNAMIC Fri Mar 1 16:53:59 UTC 2024

~ uname -s
Linux

```

13.passwd Command

The passwd command is used to change the password of a user account.

Syntax :

passwd [-options] [username]

Example :

```

~ passwd
Changing password for user cyrixninja.
Current password:
New password:
BAD PASSWORD: The password is the same as the old one
passwd: Authentication token manipulation error

```

14.mkdir Command

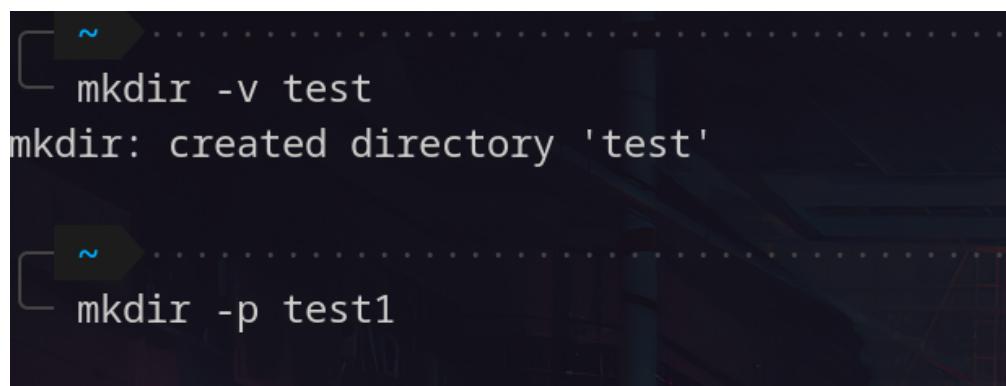
This command is used to make Directories.

Syntax :

`mkdir [-OPTION] DIRECTORY`

Example :

Option	Use
<code>mkdir -v</code>	Print a message for each created directory
<code>mkdir -p</code>	No error if existing, make parent directories as needed
<code>mkdir -m</code>	To control the permissions of new directories



```
└─~ mkdir -v test
mkdir: created directory 'test'

└─~ mkdir -p test1
```

15.rmdir Command

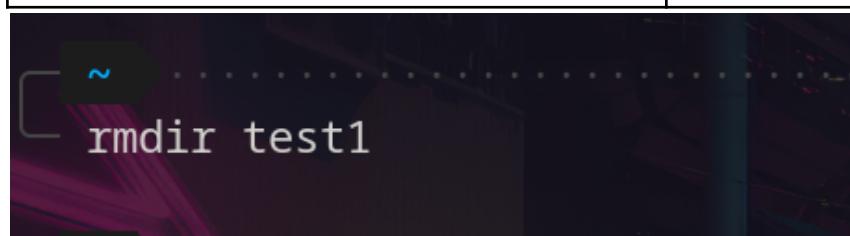
This command removes empty directories from your filesystem.

Syntax :

`rmdir [-OPTION] DIRECTORY`

Example :

Option	Use
<code>rmdir</code>	Remove directory and its ancestors... e.g., ‘rmdir -p a/b/c’ is similar to ‘rmdir a/b/c a/b a’



```
└─~ rmdir test1
```

16.cp(copy) Command

This command is used to copy files and directories.

Syntax :

cp [option] source destination/directory

Example :

Option	Use
cp -i	Interactive - ask before overwrite
cp -f	Force copy by removing the destination file if needed
cp -n	Do not overwrite an existing file
cp -u	Update - copy when source is newer than destination
cp -s	Make symbolic links instead of copying
cp -R	Copy directories recursively
cp -v	Print informative messages

```
~/test
cp hello.txt hello1.txt

~/test
cat hello1.txt
Hello
There%
```

```
~/test
cp -v hello.txt hello2.txt
'hello.txt' -> 'hello2.txt'
```

```
~/test
cp -f hello.txt hello3.txt
```

17.mv(move) Command

mv command is used to move files and directories.

Syntax :

mv [-options] source dest

Example :

Option	Use
mv -i	Interactive prompt before overwrite
mv -f	Force move by overwriting destination file without prompt
mv -n	Never overwrite any existing file
mv -u	Update - move when source is newer than destination
mv -v	Print informative messages

```
~/test
└─ ls
example hello1.txt hello2.txt hello3.txt hello.txt

~/test
└─ mv -i hello.txt ../example

~/test
└─ ls
example hello1.txt hello2.txt hello3.txt

~/test
└─ mv -v hello1.txt ../example
renamed 'hello1.txt' -> '../example'

~/test
└─ ls
example hello2.txt hello3.txt
```

18.rm(remove) Command

The ‘rm’ command is used to delete files and directories.

Syntax :

rm [-OPTION] Filename

Example :

Option	Use
rm -i	Prompt before every removal
rm -d	Delete a empty directory
rm -r	Remove directories and their contents recursively
rm -f	To remove the file forcefully

The screenshot shows a terminal window with five distinct command entries:

- ls
- example hello2.txt hello3.txt
- rm -r example
- ls
hello2.txt hello3.txt
- rm -i hello2.txt
rm: remove regular file 'hello2.txt'? y
- ls
hello3.txt

The terminal uses a dark theme with blue highlights for command names like 'rm' and 'ls'. The cursor is visible at the end of the final 'ls' command.

19.cut Command

The cut command extracts a given number of characters or columns from a file.

Syntax :

cut [-options] [file]

Example :

Option	Use
cut -c	Select only the characters from each line as specified in LIST
cut -b	Select only the bytes from each line as specified in LIST
cut -f	Cuts the input file using a list of fields. The default field to be used TAB. The default behaviour can be overwritten by use of -d option
cut -c	Specifies a delimiter to be used as a field. Default field is TAB and this option overwrites this default behaviour



```

~/test
cat hello3.txt
Hello
There%

```



```

~/test
cut -c 3 hello3.txt
l
e

```



```

~/test
cut -b 3 hello3.txt
l
e

```

20. paste Command

The paste command displays the corresponding lines of multiple files side-by-side.

Syntax :

```
paste [-options] [file]
```

Example :

Option	Use
paste -d	Reuse characters from LIST instead of tabs
paste -s	Paste one file at a time instead of in parallel

```
~/test
paste -d ':' no.txt char.txt
1:A
2:B
3:C
:D

~/test
paste -s no.txt char.txt
1      2      3
A      B      C      D
```

21. more Command

The more command is a command line utility for viewing the contents of a file or files once screen at a time.

Syntax :

```
more [-options] [file]
```

Example :

Option	Use
more -c	Clear screen before displaying
more -number	To Specify how many lines are printed

	in the screen for a given file
more -s	Doesn't display extra blank lines

```
~/test
more -c hello3.txt
```

Hello
There

```
~/test
more -s hello3.txt
Hello
There
```

22.cmp Command

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.

If a difference is found, it reports the byte and line number where the first difference is found.

If no differences are found, by default, cmp returns no output.

Syntax :

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

Example :

Option	Use
cmp -b	Print differing bytes
cmp -i	Skip a particular number of initial bytes from both the files
cmp -s	Do not print anything; only return an exit status indicating whether the files differ

<code>cmp -n</code>	Compare at most LIMIT bytes
<code>cmp -l</code>	Print byte position and byte value for all differing bytes

```

~/test
└── cmp -b hello.txt hello2.txt
hello.txt hello2.txt differ: byte 2, line 1 is 145 e 151 i

~/test
└── cmp -l hello.txt hello2.txt
2 145 151
3 154 40
4 154 167
6 40 162
7 167 154
8 157 144
cmp: EOF on hello2.txt after byte 8

```

23.comm Command

Compare two sorted files line by line.

Syntax :

`comm [OPTION]... FILE1 FILE2`

Example :

Option	Use
<code>comm -1</code>	Suppress column 1 (lines unique to FILE1)
<code>comm -2</code>	Suppress column 2 (lines unique to FILE2)
<code>comm -3</code>	Suppress column 3 (lines that appear in both files)

```
~/test
comm -1 hello.txt hello2.txt
hi world

~/test
comm -2 hello.txt hello2.txt
hello world

~/test
comm -3 hello.txt hello2.txt
hello world
    hi world
```

24.diff(difference) Command

This command is used to display the differences in the files by comparing the files line by line. Diff analyses two files and prints the lines that are different. Essentially, it outputs a set of instructions for how to change one file to make it identical to the second file.

Syntax :

```
diff [options] File1 File2
```

Example :

Option	Use
diff -b	Ignores spacing differences
diff -i	Ignores case

```
~/test
diff -b hello.txt hello2.txt
1c1
< hello world
\ No newline at end of file
---
> hi world
\ No newline at end of file

~/test
diff -i hello.txt hello2.txt
1c1
< hello world
\ No newline at end of file
---
> hi world
\ No newline at end of file
```

25.chmod(change mode) Command

chmod is used to change the permissions of files or directories.

Syntax :

chmod [reference][operator][mode] file...

Example :

Reference	Class	Description
u	owner	file's owner
g	group	users who are members of the file's group
o	others	users who are neither the file's owner nor members of the file's group
a	all	All three of the above

Operator	Description
+	Adds the specified modes to the specified classes
-	Removes the specified modes from the specified classes
=	The modes specified are to be made the exact modes for the specified classes

Permission	Description
r	Permission to read the file
w	Permission to write (or delete) the file
x	Permission to execute the file, or, in the case of a directory, search it

```
~/test
chmod +x hello.sh

~/test
chmod 764 hello.txt
```

```
~/test
ls -l
total 24
-rw-r--r--. 1 cyrixninja cyrixninja 8 Mar 12 16:24 char.txt
-rw-r--r--. 1 cyrixninja cyrixninja 8 Mar 12 16:33 hello2.txt
-rw-r--r--. 1 cyrixninja cyrixninja 11 Mar 12 16:05 hello3.txt
-rwxr-xr-x. 1 cyrixninja cyrixninja 19 Mar 12 16:49 hello.sh
-rwxrw-r--. 1 cyrixninja cyrixninja 11 Mar 12 16:33 hello.txt
-rw-r--r--. 1 cyrixninja cyrixninja 6 Mar 12 16:23 no.txt
```

26.chown(change owner) Command

The chown command changes ownership of files and directories in a Linux filesystem.

Syntax :

chown [OPTIONS] USER[:GROUP] FILE(s)

Example :

```
~/test
sudo chown root example
[sudo] password for cyrixninja:

~/test
ls -l
total 24
-rw-r--r--. 1 cyrixninja cyrixninja 8 Mar 12 16:24 char.txt
drwxr-xr-x. 1 root      cyrixninja 0 Mar 12 16:53 example
-rw-r--r--. 1 cyrixninja cyrixninja 8 Mar 12 16:33 hello2.txt
-rw-r--r--. 1 cyrixninja cyrixninja 11 Mar 12 16:05 hello3.txt
-rwxr-xr-x. 1 cyrixninja cyrixninja 19 Mar 12 16:49 hello.sh
-rwxrw-r--. 1 cyrixninja cyrixninja 11 Mar 12 16:33 hello.txt
-rw-r--r--. 1 cyrixninja cyrixninja 6 Mar 12 16:23 no.txt
```

27.chgrp(change group) Command

The chgrp command is used to change group ownership of a file/directory.

Syntax :

chgrp [OPTION]... GROUP FILE/DIR...

Example :

```
~/test
└─ sudo chgrp root example

~/test
└─ ls -a
.. char.txt example hello2.txt hello3.txt hello.sh hello.txt no.txt

~/test
└─ ls -l
total 24
-rw-r--r-- 1 cyrixninja cyrixninja 8 Mar 12 16:24 char.txt
drwxr-xr-x. 1 root      root        0 Mar 12 16:53 example
-rw-r--r-- 1 cyrixninja cyrixninja 8 Mar 12 16:33 hello2.txt
-rw-r--r-- 1 cyrixninja cyrixninja 11 Mar 12 16:05 hello3.txt
-rwxr-xr-x. 1 cyrixninja cyrixninja 19 Mar 12 16:49 hello.sh
-rwxrw-r--. 1 cyrixninja cyrixninja 11 Mar 12 16:33 hello.txt
-rw-r--r--. 1 cyrixninja cyrixninja 6 Mar 12 16:23 no.txt
```

28.file Command

The file command is used to determine a file's type.

Syntax :

file [OPTIONS] file1 file2 ...

Example :

```
~/test
└─ file hello.txt
hello.txt: ASCII text, with no line terminators

~/test
└─ file -i hello.sh
hello.sh: text/plain; charset=us-ascii
```

29.finger Command

finger looks up and displays information about system users.

Syntax :

finger [-option] [username]

Example :

```
~/test
└ finger root
Login: root
Directory: /root
Last login Fri Feb 23 19:42 (IST) on pts/3
Name: Super User
Shell: /bin/bash
No mail.
No Plan.

~/test
└ finger -m
Login      Name      Tty      Idle  Login Time  Office
cyrixninja Harsh Kumar *seat0    80d  Mar 12 14:21
(reen)
cyrixninja Harsh Kumar   tty2     2:40  Mar 12 14:21
```

30.sleep Command

The sleep command is used to delay for a specified amount of time.

Syntax :

sleep NUMBER[SUFFIX]...

Example :

```
~/test
└ sleep 5

~/test
└ sleep 2
```

31.ps Command

Reports a snapshot of the status of currently running processes.

Syntax :

ps [option]

Example :

Option	Use
ps -e	Display every active process on a Linux system in generic (Unix/Linux) format
ps -x	View all processes owned by you
ps -f	To provide more information on processes
ps -u	Filter processes by its user

```
~/test
└─ ps -e
    PID TTY      TIME CMD
      1 ?        00:00:01 systemd
      2 ?        00:00:00 kthreadd
      3 ?        00:00:00 pool_workqueue_release
      4 ?        00:00:00 kworker/R-rcu_g
      5 ?        00:00:00 kworker/R-rcu_p
      6 ?        00:00:00 kworker/R-slab_
      7 ?        00:00:00 kworker/R-netns
      9 ?        00:00:00 kworker/0:0H-kblockd
     12 ?        00:00:00 kworker/R-mm_pe
     14 ?        00:00:00 rcu_tasks_kthread
     15 ?        00:00:00 rcu_tasks_rude_kthread
     16 ?        00:00:00 rcu_tasks_trace_kthread
     17 ?        00:00:00 ksoftirqd/0
     18 ?        00:00:02 rcu_preempt
```

```
~/test
- ps -f
UID      PID  PPID  C STIME TTY      TIME CMD
cyrixni+ 11979  11954  0 15:01 pts/0    00:00:03 zsh
cyrixni+ 12044  2107   0 15:01 pts/0    00:00:00 zsh
cyrixni+ 12363  2107   0 15:01 pts/0    00:00:00 zsh
cyrixni+ 12364  2107   0 15:01 pts/0    00:00:00 zsh
cyrixni+ 12366  12044  0 15:01 pts/0    00:00:00 /home/cyrixninja/.cache/gitstatus/git
cyrixni+ 37788  11979  0 17:07 pts/0    00:00:00 ps -f
```

32.kill Command

It is used to terminate processes manually.

kill command sends a signal to a process which terminates the process.

If the user doesn't specify any signal which is to be sent along with kill command then default TERM signal is sent that terminates the process..

Syntax :

kill [option] PID

Example:

```
~/test
- ps -f
UID      PID  PPID  C STIME TTY      TIME CMD
cyrixni+ 11979  11954  0 15:01 pts/0    00:00:03 zsh
cyrixni+ 12044  2107   0 15:01 pts/0    00:00:00 zsh
cyrixni+ 12363  2107   0 15:01 pts/0    00:00:00 zsh
cyrixni+ 12364  2107   0 15:01 pts/0    00:00:00 zsh
cyrixni+ 12366  12044  0 15:01 pts/0    00:00:00 /home/cyrixninja/.cache/gitstatus/git
cyrixni+ 37788  11979  0 17:07 pts/0    00:00:00 ps -f

~/test
kill 11979
```

```
~/test
kill -l
HUP INT QUIT ILL TRAP IOT BUS FPE KILL USR1 SEGV USR2 PIPE ALRM TER
M STKFLT CHLD CONT STOP TSTP TTIN TTOU URG XCPU XFSZ VTALRM PROF WI
NCH POLL PWR SYS
```

33.wc Command

It's used to find out number of newline count, word count, byte and characters count in a files specified by the file arguments.

Syntax :

wc [options] filenames

Example:

Option	Use
wc -l	Prints the number of lines in a file
wc -w	Prints the number of words in a file
wc -c	Displays the count of bytes in a file
wc -L	Prints only the length of the longest line in a file

```
~/test
└─ cat hello.txt
hello world% 

~/test
└─ wc hello.txt
0 2 11 hello.txt

~/test
└─ wc -l hello.txt
0 hello.txt

~/test
└─ wc -c hello.txt
11 hello.txt
```

34.ln Command

ln creates links between files.

ln creates hard links by default, or symbolic links if the -s (–symbolic) option is specified. When creating hard links, each TARGET must exist.

Syntax :

ln [OPTION]... [-T] TARGET LINK_NAME

Example:

The screenshot shows a terminal window with four command-line sessions. The first session creates a symbolic link named 'hello6.txt' pointing to 'hello2.txt'. The second session lists files in the directory, showing 'hello6.txt' as a file. The third session creates another symbolic link 'hello5.txt' pointing to 'hello2.txt'. The fourth session lists files again, now showing both 'hello6.txt' and 'hello5.txt' as files.

```
~/test
└─ ln -s hello2.txt hello6.txt
  ↗ 1 x | at 17:32:14

~/test
└─ ls
char.txt  hello2.txt  hello6.txt  hello.txt
example   hello3.txt  hello.sh    no.txt
  ↗ ✓ | at 17:32:20

~/test
└─ ln hello2.txt hello5.txt
  ↗ ✓ | at 17:32:22

~/test
└─ ls
char.txt  hello2.txt  hello5.txt  hello.sh  no.txt
example   hello3.txt  hello6.txt  hello.txt
  ↗ ✓ | at 17:33:00
```

35.nl Command

nl command numbers the lines in a file.

Syntax :

nl [OPTION]... [FILE]...

Example :

Option	Use
nl -i	Line number increment at each line
nl -s	Add STRING after (possible) line number
nl -w	Use NUMBER columns for line numbers

The image shows a terminal window with two separate command executions. The first execution shows the output of the command `nl -i 2 hello2.txt`, which prints the file `hello2.txt` with line numbers starting from 1. The second execution shows the output of the command `nl -s 2 hello2.txt`, which prints the file `hello2.txt` with line numbers starting from 12.

```
~/test
nl -i 2 hello2.txt
 1 hello
 3 world
 5 hi

~/test
nl -s 2 hello2.txt
12hello
22world
32hi
```

36.head Command

head makes it easy to output the first part (10 lines by default) of files.

Syntax :

`head [OPTION]... [FILE]...`

Example :

Option	Use
<code>head -n</code>	Print the first n lines instead of the first 10; with the leading '-', print all but the last n lines of each file
<code>head -c</code>	Print the first n bytes of each file; with a leading '-', print all but the last n bytes of each file
<code>head -q</code>	Never print headers identifying file names

```
~/test
cat hello2.txt
hello
world
hi

~/test
head -n 2 hello2.txt
hello
world

~/test
head -c 2 hello2.txt
he%
```

37.tail Command

tail is a command which prints the last few number of lines (10 lines by default) of a certain file, then terminates.

Syntax :

tail [OPTION]... [FILE]...

Example :

Option	Use
tail -n	Output the last num lines, instead of the default (10)
tail -c	Output the last num bytes of each file
tail -q	Never output headers

```

~/test
cat hello2.txt
hello
world
hi

~/test
tail -n 2 hello2.txt
world
hi

~/test
tail -c 2 hello2.txt
i

```

38.sort Command

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 sort command, it can also be used to sort numerically.

Syntax :

sort [OPTION]... [FILE]...

Option	Use
sort -c	To check if the file given is already sorted or not
sort -r	Reverse the result of comparisons
sort -n	Compare according to string numerical value
sort -nr	To sort a file with numeric data in reverse order
sort -k	Sorting a table on the basis of any column
sort -b	Ignore leading blanks

Example :

```
~/test
cat hello2.txt
hello
world
hi

~/test
sort -r hello2.txt
world
hi
hello

~/test
sort -c hello2.txt
sort: hello2.txt:3: disorder: hi

~/test
sort hello2.txt
hello
hi
world
```

39.find Command

find command searches for files in a directory hierarchy.

Syntax :

find [option] [path...] [expression]

Option	Use
find -name filename	Search for files that are specified by 'filename'
find -newer filename	Search for files that were modified/created after 'filename'
find -user name	Search for files owned by user name or ID 'name'
find -size +N/-N	Search for files of 'N' blocks; 'N'

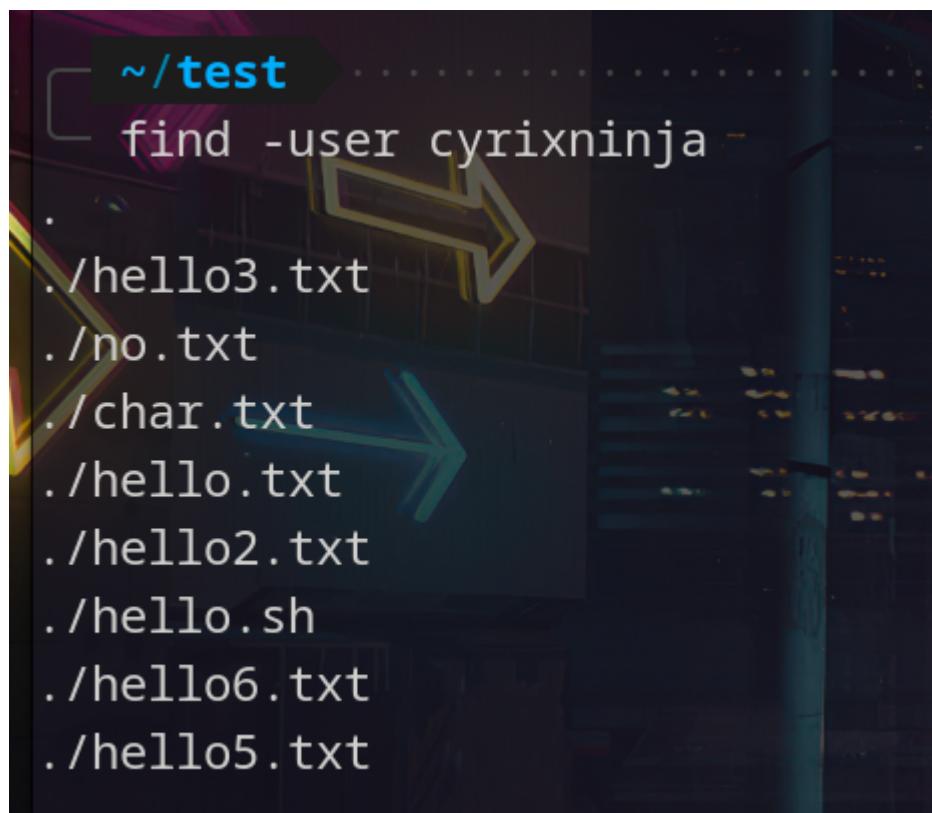
	followed by ‘c’ can be used to measure size in characters
find -empty	Search for empty files and directories
find -perm octal	Search for the file if permission is ‘octal’

Example :



```
~/test
└── find h*
    hello2.txt
    hello3.txt
    hello5.txt
    hello6.txt
    hello.sh
    hello.txt

~/test
└── find -newer hello2.txt
```



```
~/test
└── find -user cyrixninja
    .
    ./hello3.txt
    ./no.txt
    ./char.txt
    ./hello.txt
    ./hello2.txt
    ./hello.sh
    ./hello6.txt
    ./hello5.txt
```

40.uniq Command

uniq reports or filters out repeated lines in a file.

It can remove duplicates, show a count of occurrences, show only repeated lines, ignore certain characters and compare on specific fields.

Syntax :

uniq [OPTION]... [INPUT [OUTPUT]]

Option	Use
uniq -u	Prints only unique lines
uniq -d	Only print duplicated lines
uniq -D	Print all duplicate lines
uniq -c	Prefix lines with a number representing how many times they occurred
uniq -i	Ignore case when comparing

Example :

```
~/test
└─ uniq -u test.txt
world
hi

~/test
└─ uniq -d test.txt
hello

~/test
└─ cat test.txt
hello
hello
world
hi
```

41. grep Command

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 globally search for regular expression and print out.

Syntax :

grep [options] pattern [files]

Option	Use
grep -c	Prints only a count of the lines that match a pattern
grep -h	Display the matched lines, but do not display the filenames
grep -l	Displays list of a filenames only
grep -i	Ignores, case for matching

Example :



The screenshot shows a terminal window with three distinct command executions:

- The first execution shows the command `cat test.txt` followed by the contents of the file: "hello", "hello", "world", and "hi".
- The second execution shows the command `grep -h he test.txt` followed by the lines "hello" and "hello" from the file, with the matching characters highlighted in red.
- The third execution shows the command `grep -n he test.txt` followed by the output "1:hello" and "2:hello", where the line numbers and matching characters are highlighted in red.

42. pipe () Command

It redirects the command STDOUT or standard output into the given next command STDIN or standard input.

In short, the output of each process directly as input to the next one like a pipeline.

The symbol '|' denotes a pipe.

Pipes help you mash-up two or more commands at the same time and run them consecutively.

Syntax :

```
command_1 | command_2 | command_3 | .... | command_N...
```

Example :

```
~/test
cat > test2.txt
1 hello
2 there
3 hi
4 hello
5 world
^C

~/test
cat test.txt | tail -2 | head -2
world
hi
```

43. tr(translate) Command

The tr command in UNIX is a command line utility for translating or deleting characters.

It supports a range of transformations including uppercase to lowercase, squeezing repeating characters, deleting specific characters and basic find and replace.

It can be used with UNIX pipes to support more complex translation.
tr stands for translate.

Syntax :

tr [OPTION] SET1 [SET2]

POSIX Character set supported by tr command :

- [:digit:] Only the digits 0 to 9.
- [:alnum:] Any alphanumeric character.
- [:alpha:] Any alpha character A to Z or a to z.
- [:blank:] Space and TAB characters only.
- [:digit:] Hexadecimal notation 0-9, A-F, a-f.
- [:upper:] Any alpha character A to Z.
- [:lower:] Any alpha character a to z..

Option	Use
tr -s	Replaces repeated characters listed in the set1 with single occurrence
tr -d	Delete characters in string1 from the input
tr -c	complements the set of characters in string. i.e., operations apply to characters not in the given set
tr -cd	Remove all characters except digits

Example :

```
~/test
cat test2.txt | tr -cd 1-9
12345%
```



```
~/test
cat test2.txt
1 hello
2 there
3 hi
4 hello
5 world
```

```
~/test
cat test.txt
hello
hello
world
hi

~/test
cat test.txt | tr -d 'hell'
o
o
word
i
```

44. history Command

history command is used to view the previously executed command.

Syntax :

```
history
```

Example :

```
~/test
history
 1 nano ~/.zshrc
 2 git clone https://github.com/denysdovhan/spaceship-prompt.git "$ZSH_CUSTOM/themes/spaceship-prompt"
 3 nano ~/.zshrc
 4 git clone --depth=1 https://github.com/romkatv/powerlevel10k.git $ZSH_CUSTOM/themes/powerlevel10k
 5 ls
 6 cd
 7 ls
 8 nano ~/.zshrc
 9 zsh
10 git clone https://github.com/denysdovhan/spaceship-prompt.git "$ZSH_CUSTOM/themes/spaceship-prompt"
11 zsh
12 zshCopy code
```

45. write Command

write sends a message to another user.

Syntax :

write user [ttynname]

Option	Use
user	The user to write to
tty	The specific terminal to write to, if the user is logged in to more than one session

Example :

A screenshot of a terminal window on a Linux desktop. The terminal shows the following session:

```
~ who
cyrixninja seat0
cyrixninja tty2

~ write cyrixninja
hi
^C%
```

On the right side of the terminal, there is a status bar with two entries:

- 1 x | at 18:34:41
- ✓ | at 18:35:24

The status bar also displays the date and time: 2024-03-12 14:21 (login screen) and 2024-03-12 14:21 (tty2). In the background, there is a blurred image of a video game character with the text "LEVEL UP".

46. wall Command

wall send a message to everybody's terminal.

wall sends a message to everybody logged in with their mesg permission set to yes.

Syntax :

wall [-n] [-t TIMEOUT] [file]

Option	Use
wall -n	--nobanner Suppress banner
wall -t	--timeout TIMEOUT Write timeout to terminals in seconds. TIMEOUT must be positive integer. Default value is 300 seconds, which is a legacy from time when people ran terminals over modem lines.

Example :

```
~ . . . . .
sudo wall "Hello"
INT ✘ | took 60s | at 18:32:14

Broadcast message from root@fedora (pts/0) (Tue Mar 12 18:32:19 2024):
Hello
```

```
~ . . . . . 1 x | at 18:33:02
[sudo] password for root:
root@fedora:~# sudo wall -t 2 "Hello"
Broadcast message from root@fedora (pts/0) (Tue Mar 12 18:33:09 2024):
Hello
```