

Unit-9

Unix Commands

Subject Code: 3140702

Subject Name: Operating System

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

cd Command Example

cd ..

Change Current directory to parent directory

File Edit View Search Terminal Help

```
[student@localhost ~]$ pwd
/home/student
[student@localhost ~]$ cd ..
[student@localhost home]$ pwd
/home
[student@localhost home]$
```

cd Command Example

cd ~

Move to users home directory from anywhere

File Edit View Search Terminal Help

```
[student@localhost Documents]$ ls
```

```
lab-1
```

```
[student@localhost Documents]$ cd ~
```

```
[student@localhost ~]$ ls
```

```
Desktop Documents Downloads Music Pictures Public Templates Videos
```

```
[student@localhost ~]$
```

cd Command Example

cd lab_1

Change from current working directory to lab_1

File Edit View Search Terminal Help

```
[student@localhost Documents]$ ls
```

```
lab-1
```

```
[student@localhost Documents]$ cd lab-1/
```

```
[student@localhost lab-1]$ pwd
```

```
/home/student/Documents/lab-1
```

```
[student@localhost lab-1]$
```

cd Command Example

cd ../downloads

If we are currently in `/home/student/documents` then we would be placed in `/home/student/downloads`

File Edit View Search Terminal Help

```
[student@localhost Documents]$ pwd
/home/student/Documents
[student@localhost Documents]$ cd ../Downloads/
[student@localhost Downloads]$ pwd
/home/student/Downloads
[student@localhost Downloads]$
```

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 Command

\$ ls -l /etc/hosts

Output:

-rw-r--r-- 1 root root 337 Oct 4 11:31 /etc/hosts

name of the file/dir

specifies owner of the
file/dir

files

-rw- r-- r--

owner

group

others

number of links for dir/file

specifies the date and time of
the last modification of the file

specifies group of the
file/dir

indicates the file/dir size in
bytes.

Is Command Example

ls -l

To show long listing information about the file/directory

File Edit View Search Terminal Help

```
[student@localhost ~]$ ls
calc.sh  Documents  hello.c  Pictures  Templates
Desktop  Downloads  Music    Public    Videos
[student@localhost ~]$
```

File Edit View Search Terminal Help

```
[student@localhost ~]$ ls -l
total 32
-rw-rw-r--. 1 student student    0 Mar 24 00:23 calc.sh
drwxr-xr-x. 2 student student 4096 Mar 24 00:23 Desktop
drwxr-xr-x. 3 student student 4096 Mar 24 00:18 Documents
drwxr-xr-x. 2 student student 4096 Mar 23 23:29 Downloads
-rw-rw-r--. 1 student student    0 Mar 24 00:22 hello.c
drwxr-xr-x. 2 student student 4096 Mar 23 23:29 Music
drwxr-xr-x. 2 student student 4096 Mar 23 23:29 Pictures
drwxr-xr-x. 2 student student 4096 Mar 23 23:29 Public
drwxr-xr-x. 2 student student 4096 Mar 23 23:29 Templates
drwxr-xr-x. 2 student student 4096 Mar 23 23:29 Videos
[student@localhost ~]$
```

Is Command Example

ls -a	List all files including hidden file starting with '.'
--------------	---

File Edit View Search Terminal Help

```
[student@localhost ~]$ ls -a
```

```
.          Desktop      .gtk-bookmarks  Pictures
..         .dmrc        .gvfs           Public
.bash_logout Documents    hello.c         .pulse
.bash_profile Downloads   .ICEauthority   .pulse-cookie
.bashrc    .esd_auth   .imsettings.log .recently-used.xbel
.cache     .gconf      .local          Templates
calc.sh    .gconfd     .mozilla        .thumbnails
.config    .gnome2     Music           Videos
.dbus      .gnome2_private .nautilus       .xsession-errors
[student@localhost ~]$
```

Is Command Example

ls -r

List in reverse order

File Edit View Search Terminal Help

```
[student@localhost ~]$ ls -r
```

```
Videos      Public      Music      Downloads  Desktop
```

```
Templates  Pictures  hello.c  Documents  calc.sh
```

```
[student@localhost ~]$
```

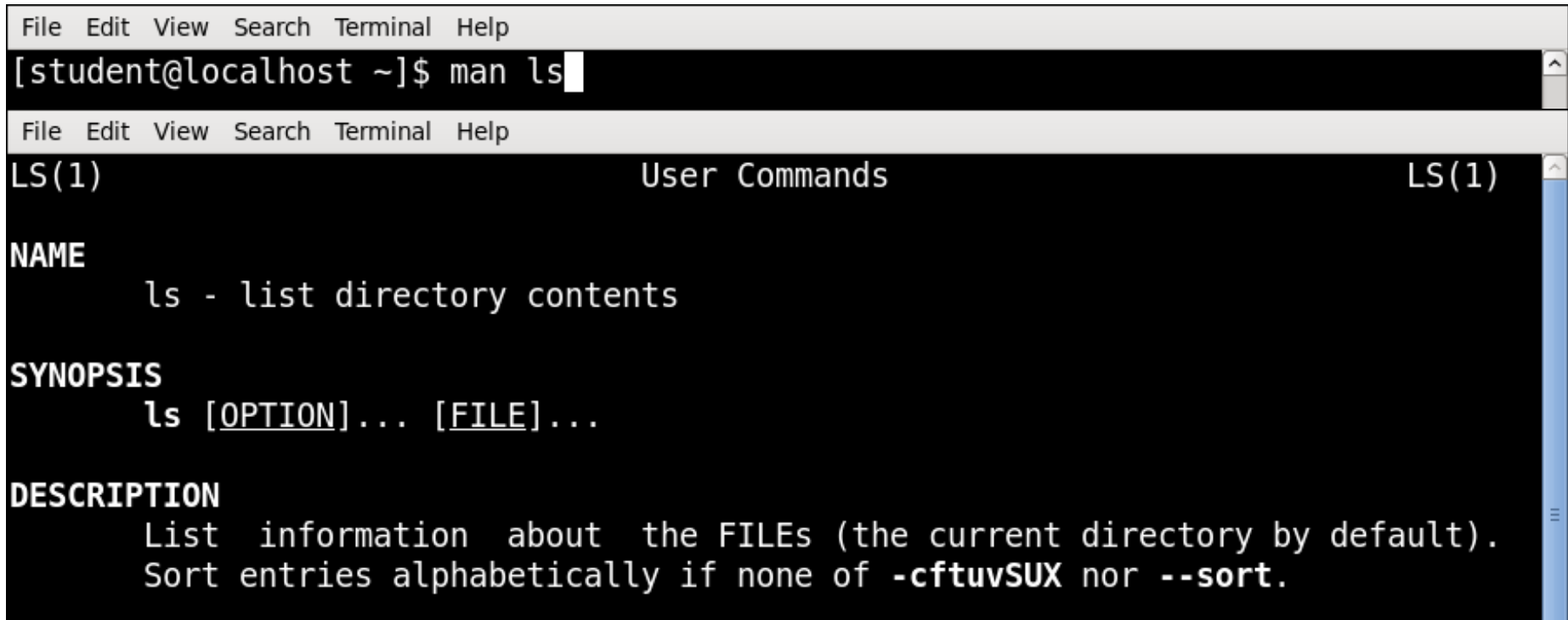
man Command

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

- **Syntax :**

`man [command name]`

- **Example**



```
File Edit View Search Terminal Help
[student@localhost ~]$ 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.
```

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 create horizontal tab spaces

echo Command Example

```
File Edit View Search Terminal Help
[student@localhost ~]$ echo "hello linux"
hello linux
[student@localhost ~]$
```

echo -n

Do not output a trailing newline

```
File Edit View Search Terminal Help
[student@localhost ~]$ echo -n "hello linux"
hello linux[student@localhost ~]$
```

echo Command Example

echo -e	Enable interpretation of backslash escape sequences
\b	It removes all the spaces in between the text
\n	It creates new line from where it is used
\t	It create horizontal tab spaces

```
File Edit View Search Terminal Help
[student@localhost ~]$ echo -e "Hi \bGood \bMorning"
HiGoodMorning
[student@localhost ~]$ echo -e "Hi \nGood \nMorning"
Hi
Good
Morning
[student@localhost ~]$ echo -e "Hi \tGood \tMorning"
Hi      Good      Morning
[student@localhost ~]$
```

cal Command

- Displays a simple, formatted calendar in your terminal.

- **Syntax :**

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

- **Example :**

Option	Use
<code>cal -1</code>	Display single month output. (This is the default.)
<code>cal -3</code>	Display three months spanning the date.
<code>cal -s</code>	Display Sunday as the first day of the week.
<code>cal -m</code>	Display Monday as the first day of the week.
<code>cal -j</code>	Use day-of-year numbering for all calendars. These are also called ordinal days. Ordinal days range from 1 to 366.
<code>cal -y</code>	Display a calendar for the whole year

cal Command Example

Cal or cal -1

Display single month output. (This is the default.)

File Edit View Search Terminal Help

```
[student@localhost ~]$ cal
```

March 2020

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				

```
[student@localhost ~]$
```

cal Command Example

cal -3

Display three months spanning the date.

File Edit View Search Terminal Help

```
[student@localhost ~]$ cal -3
```

February 2020							March 2020							April 2020						
Su	Mo	Tu	We	Th	Fr	Sa	Su	Mo	Tu	We	Th	Fr	Sa	Su	Mo	Tu	We	Th	Fr	Sa
						1	1	2	3	4	5	6	7				1	2	3	4
2	3	4	5	6	7	8	8	9	10	11	12	13	14	5	6	7	8	9	10	11
9	10	11	12	13	14	15	15	16	17	18	19	20	21	12	13	14	15	16	17	18
16	17	18	19	20	21	22	22	23	24	25	26	27	28	19	20	21	22	23	24	25
23	24	25	26	27	28	29	29	30	31					26	27	28	29	30		

```
[student@localhost ~]$
```

cal Command Example

cal -s

Display Sunday as the first day of the week.

File Edit View Search Terminal Help

```
[student@localhost ~]$ cal -s
```

March 2020

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				

```
[student@localhost ~]$
```

cal Command Example

cal -m

Display Monday as the first day of the week.

File Edit View Search Terminal Help

```
[student@localhost ~]$ cal -m
```

March 2020

Mo	Tu	We	Th	Fr	Sa	Su
----	----	----	----	----	----	----

						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					
----	----	--	--	--	--	--

```
[student@localhost ~]$
```

cal Command Example

cal -j

Use day-of-year numbering for all calendars. These are also called ordinal days. Ordinal days range from 1 to 366.

File Edit View Search Terminal Help

```
[student@localhost ~]$ cal -j
```

March 2020

Sun	Mon	Tue	Wed	Thu	Fri	Sat
61	62	63	64	65	66	67
68	69	70	71	72	73	74
75	76	77	78	79	80	81
82	83	84	85	86	87	88
89	90	91				

```
[student@localhost ~]$
```

cal Command Example

cal -y

Display a calendar for the whole year

File Edit View Search Terminal Help

[student@localhost ~]\$ cal -y

2020

January							February							March						
Su	Mo	Tu	We	Th	Fr	Sa	Su	Mo	Tu	We	Th	Fr	Sa	Su	Mo	Tu	We	Th	Fr	Sa
			1	2	3	4							1	1	2	3	4	5	6	7
5	6	7	8	9	10	11	2	3	4	5	6	7	8	8	9	10	11	12	13	14
12	13	14	15	16	17	18	9	10	11	12	13	14	15	15	16	17	18	19	20	21
19	20	21	22	23	24	25	16	17	18	19	20	21	22	22	23	24	25	26	27	28
26	27	28	29	30	31		23	24	25	26	27	28	29	29	30	31				
April							May							June						
Su	Mo	Tu	We	Th	Fr	Sa	Su	Mo	Tu	We	Th	Fr	Sa	Su	Mo	Tu	We	Th	Fr	Sa
			1	2	3	4						1	2		1	2	3	4	5	6
5	6	7	8	9	10	11	3	4	5	6	7	8	9	7	8	9	10	11	12	13
12	13	14	15	16	17	18	10	11	12	13	14	15	16	14	15	16	17	18	19	20
19	20	21	22	23	24	25	17	18	19	20	21	22	23	21	22	23	24	25	26	27
26	27	28	29	30			24	25	26	27	28	29	30	28	29	30				
							31													
July							August							September						
Su	Mo	Tu	We	Th	Fr	Sa	Su	Mo	Tu	We	Th	Fr	Sa	Su	Mo	Tu	We	Th	Fr	Sa
			1	2	3	4							1			1	2	3	4	5
5	6	7	8	9	10	11	2	3	4	5	6	7	8	6	7	8	9	10	11	12
12	13	14	15	16	17	18	9	10	11	12	13	14	15	13	14	15	16	17	18	19
19	20	21	22	23	24	25	16	17	18	19	20	21	22	20	21	22	23	24	25	26
26	27	28	29	30	31		23	24	25	26	27	28	29	27	28	29	30			
							30	31												
October							November							December						
Su	Mo	Tu	We	Th	Fr	Sa	Su	Mo	Tu	We	Th	Fr	Sa	Su	Mo	Tu	We	Th	Fr	Sa
				1	2	3	1	2	3	4	5	6	7			1	2	3	4	5
4	5	6	7	8	9	10	8	9	10	11	12	13	14	6	7	8	9	10	11	12
11	12	13	14	15	16	17	15	16	17	18	19	20	21	13	14	15	16	17	18	19

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 (e.g., January)
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 Command

Option	Use
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

date Command Example

date +%a

The abbreviated weekday name (e.g., Sun)

date +%A

The full weekday name (e.g., Sunday)

File Edit View Search Terminal Help

```
[student@localhost ~]$ date
Tue Mar 24 12:15:52 IST 2020
[student@localhost ~]$
```

File Edit View Search Terminal Help

```
[student@localhost ~]$ date +%a
Tue
[student@localhost ~]$ date +%A
Tuesday
[student@localhost ~]$
```

date Command Example

date +%b	The abbreviated month name (e.g., Jan)
date +%B	Locale's full month name (e.g., January)

File Edit View Search Terminal Help

```
[student@localhost ~]$ date +%b
```

```
Mar
```

```
[student@localhost ~]$ date +%B
```

```
March
```

```
[student@localhost ~]$
```

date Command Example

date +%c	Full date with IST timing
date +%C	The current century; like %Y, except omit last two digits (e.g., 20)

File Edit View Search Terminal Help

```
[student@localhost ~]$ date +%c
Tue 24 Mar 2020 12:16:47 PM IST
[student@localhost ~]$ date +%C
20
[student@localhost ~]$
```

date Command Example

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)

File Edit View Search Terminal Help

```
[student@localhost ~]$ date
Tue Mar 24 12:23:52 IST 2020
[student@localhost ~]$ date +%d
24
[student@localhost ~]$ date +%m
03
[student@localhost ~]$ date +%y
20
[student@localhost ~]$
```

date Command Example

date +%y	Displays last two digits of the year(00 to 99)
date +%Y	Display four-digit year.

File Edit View Search Terminal Help

```
[student@localhost ~]$ date +%y
20
[student@localhost ~]$ date +%Y
2020
[student@localhost ~]$
```

date Command Example

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

File Edit View Search Terminal Help

```
[student@localhost ~]$ date +%T
```

```
12:25:13
```

```
[student@localhost ~]$ date +%H
```

```
12
```

```
[student@localhost ~]$ date +%M
```

```
25
```

```
[student@localhost ~]$ date +%S
```

```
19
```

```
[student@localhost ~]$
```

date Command Example

date +%V

**ISO week number, with Monday as first day of week
(01..53)**

File Edit View Search Terminal Help

```
[student@localhost ~]$ date +%V
```

```
13
```

```
[student@localhost ~]$
```

date Command Example

date +%P

locale's equivalent of either AM or PM

File Edit View Search Terminal Help

```
[student@localhost ~]$ date +%p
```

```
PM
```

```
[student@localhost ~]$ date +%P
```

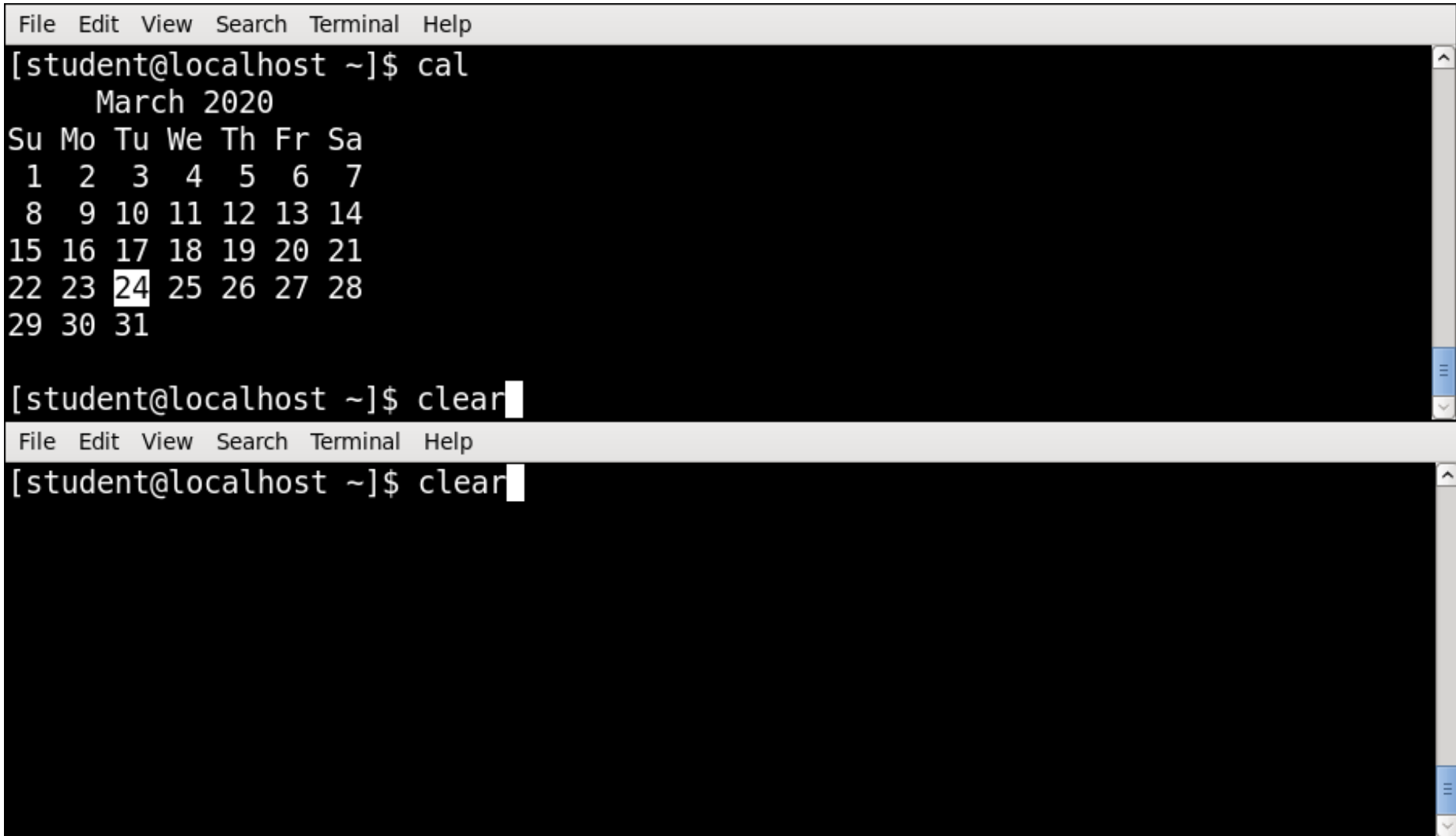
```
pm
```

```
[student@localhost ~]$
```


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

clear Command Example



The image displays two sequential terminal window screenshots. The top window shows the output of the 'cal' command, which displays a calendar for March 2020. The calendar is formatted with days of the week (Su, Mo, Tu, We, Th, Fr, Sa) and dates (1 through 31). The date 24 is highlighted. The bottom window shows the same terminal after the 'clear' command has been typed, resulting in a blank screen.

```
File Edit View Search Terminal Help
[student@localhost ~]$ cal
      March 2020
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

[student@localhost ~]$ clear
File Edit View Search Terminal Help
[student@localhost ~]$ clear
```

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

cat Command

▪ Example :

\$ cat > file1.txt

- It creates file1.txt and allow us to insert content for this file.
- After inserting content you can use ctrl+c to exit the file.

\$cat file.txt > newfile.txt

- Read the contents of file.txt and write them to newfile.txt, overwriting anything newfile.txt previously contained. If newfile.txt does not exist, it will be created.

\$cat file.txt >> newfile.txt

- Read the contents of **file.txt** and append them to the end of **newfile.txt**. If **newfile.txt** does not exist, it will be created.

cat Command

- **Example :**

cat file1.txt file2.txt

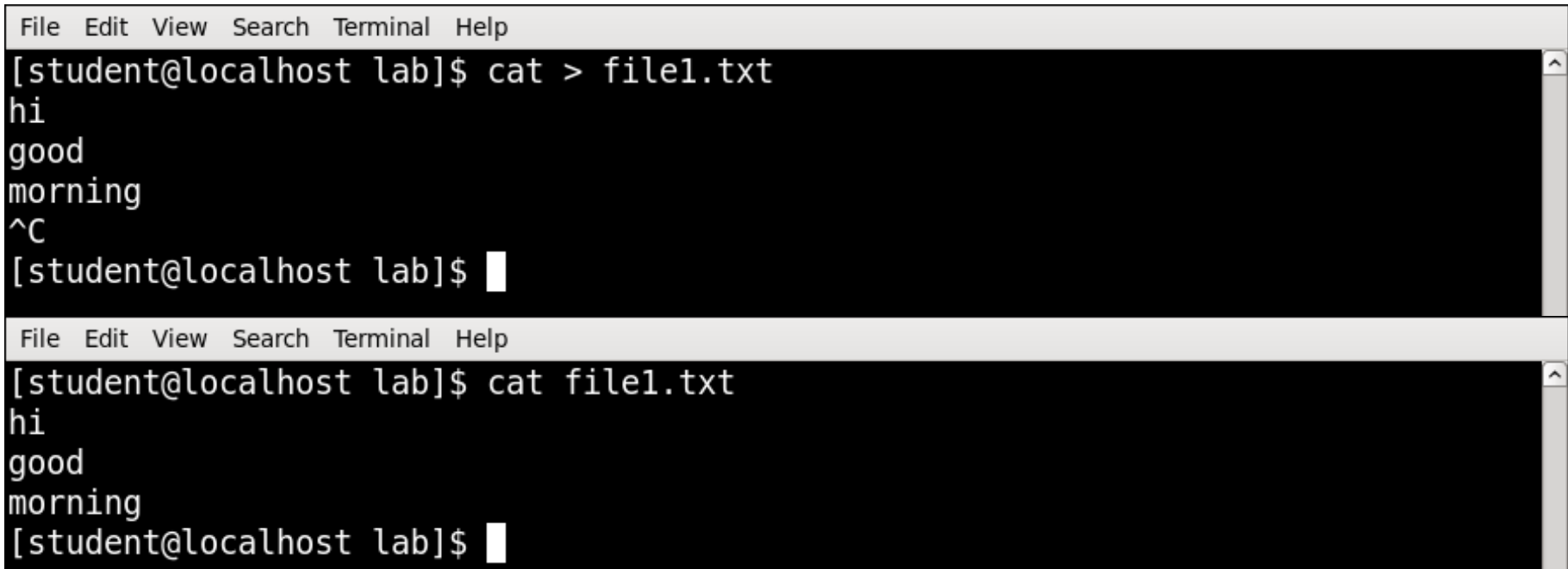
- It will read the contents of file1.txt and file2.txt and display the result in the terminal.

cat file1.txt file2.txt > combinedfile.txt

- It will concatenate the contents of file1.txt and file2.txt and write them to a new file combinedfile.txt using the (>) operator.
- If the combinedfile.txt file doesn't exist the command will create it. Otherwise it will overwrite the file.

cat Command Example

- \$ cat > file1.txt
 - It creates file1.txt and allow us to insert content for this file.
 - After inserting content you can use ctrl+c to exit the file.



```
File Edit View Search Terminal Help
[student@localhost lab]$ cat > file1.txt
hi
good
morning
^C
[student@localhost lab]$

File Edit View Search Terminal Help
[student@localhost lab]$ cat file1.txt
hi
good
morning
[student@localhost lab]$
```

The image displays two terminal window screenshots. The top screenshot shows the command `cat > file1.txt` being executed, followed by the input of the text "hi", "good", and "morning" on separate lines. The session is terminated with `^C`. The bottom screenshot shows the command `cat file1.txt` being executed, which outputs the content of the file: "hi", "good", and "morning" on separate lines.

cat Command Example

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

File Edit View Search Terminal Help

```
[student@localhost lab]$ cat -E file1.txt
```

```
hi$
```

```
good$
```

```
morning$
```

```
[student@localhost lab]$ cat -b file1.txt
```

```
1 hi
```

```
2 good
```

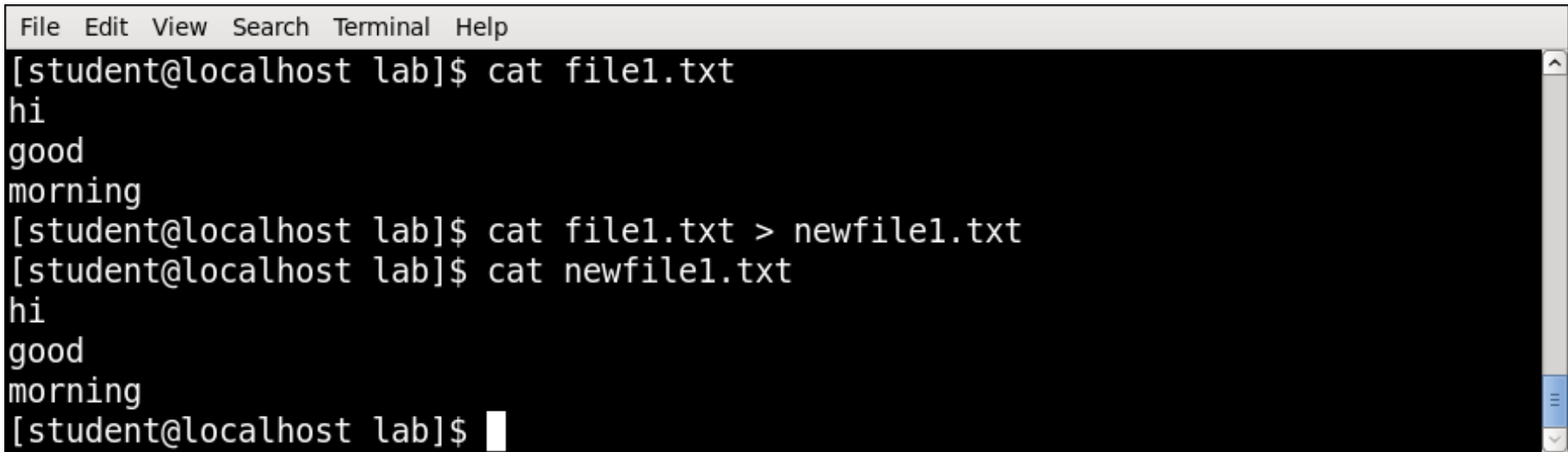
```
3 morning
```

```
[student@localhost lab]$
```

cat Command Example

- \$cat file.txt > newfile.txt

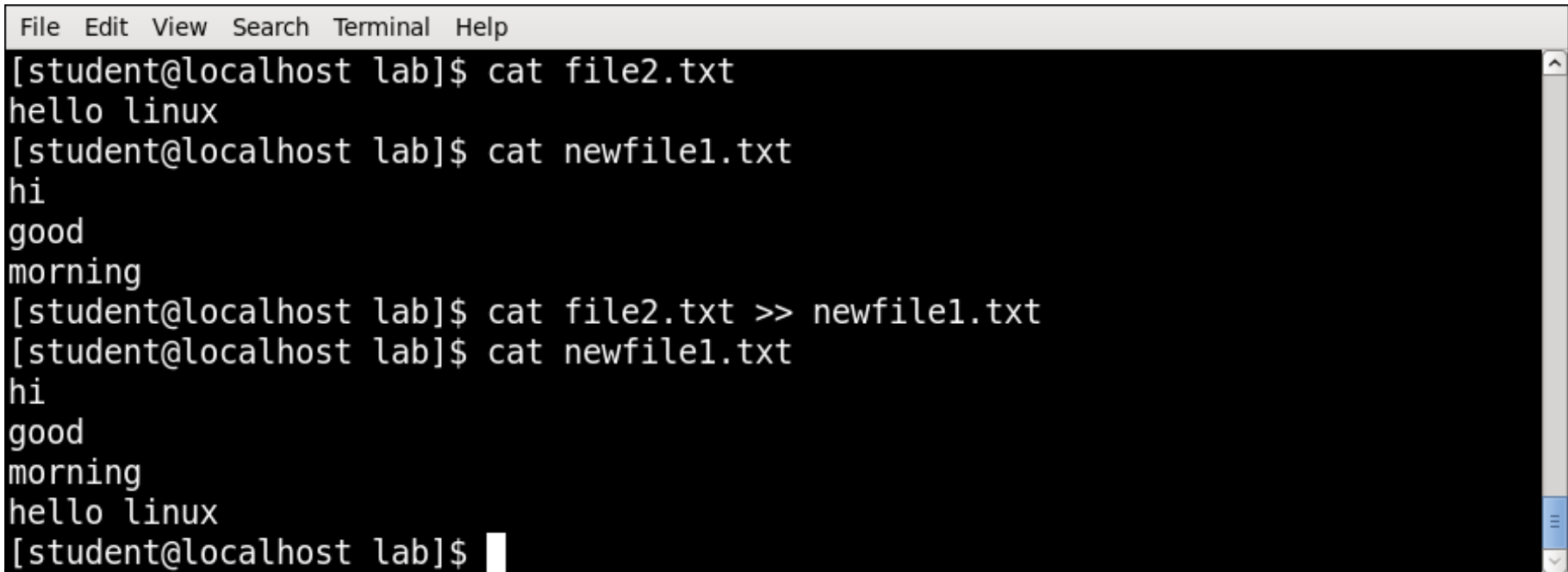
- Read the contents of file.txt and write them to newfile.txt, overwriting anything newfile.txt previously contained.
- If newfile.txt does not exist, it will be created.

A terminal window with a menu bar (File, Edit, View, Search, Terminal, Help) and a dark background. The prompt is [student@localhost lab]\$. The first command is cat file1.txt, which outputs 'hi', 'good', and 'morning'. The second command is cat file1.txt > newfile1.txt. The third command is cat newfile1.txt, which also outputs 'hi', 'good', and 'morning'. The prompt is now [student@localhost lab]\$ with a cursor.

```
File Edit View Search Terminal Help
[student@localhost lab]$ cat file1.txt
hi
good
morning
[student@localhost lab]$ cat file1.txt > newfile1.txt
[student@localhost lab]$ cat newfile1.txt
hi
good
morning
[student@localhost lab]$
```


cat Command Example

- \$cat file.txt >> newfile.txt
 - Read the contents of **file.txt** and append them to the end of **newfile.txt**. If **newfile.txt** does not exist, it will be created.

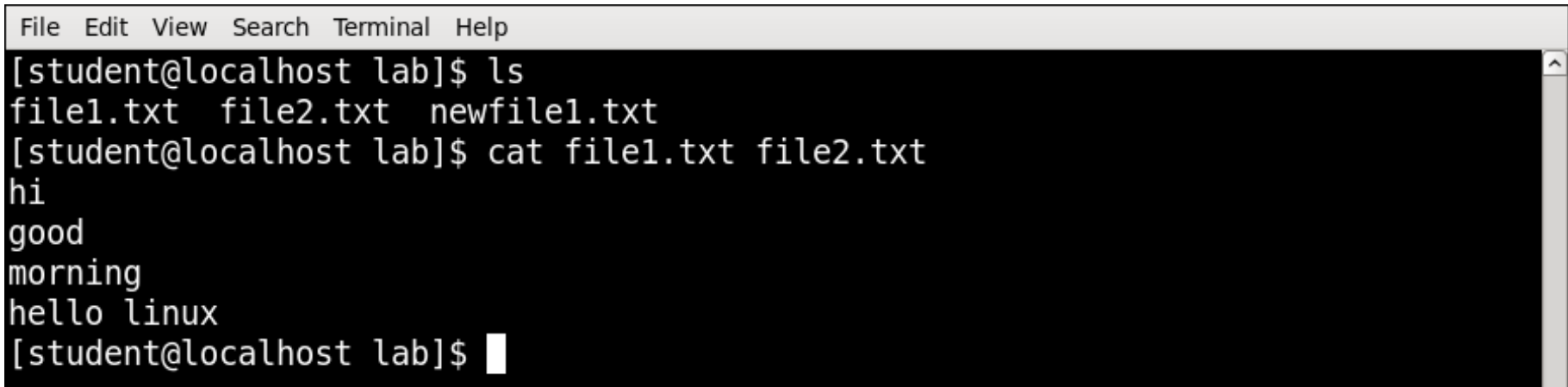
A terminal window with a menu bar (File, Edit, View, Search, Terminal, Help) and a dark background. It shows a sequence of commands and their outputs. First, 'cat file2.txt' outputs 'hello linux'. Then, 'cat newfile1.txt' outputs 'hi', 'good', and 'morning'. Finally, 'cat file2.txt >> newfile1.txt' is executed, followed by 'cat newfile1.txt' which outputs the concatenated content: 'hi', 'good', 'morning', 'hello linux'. The prompt '[student@localhost lab]\$' is visible at the end of each command line.

```
File Edit View Search Terminal Help
[student@localhost lab]$ cat file2.txt
hello linux
[student@localhost lab]$ cat newfile1.txt
hi
good
morning
[student@localhost lab]$ cat file2.txt >> newfile1.txt
[student@localhost lab]$ cat newfile1.txt
hi
good
morning
hello linux
[student@localhost lab]$
```

cat Command Example

- **cat file1.txt file2.txt**

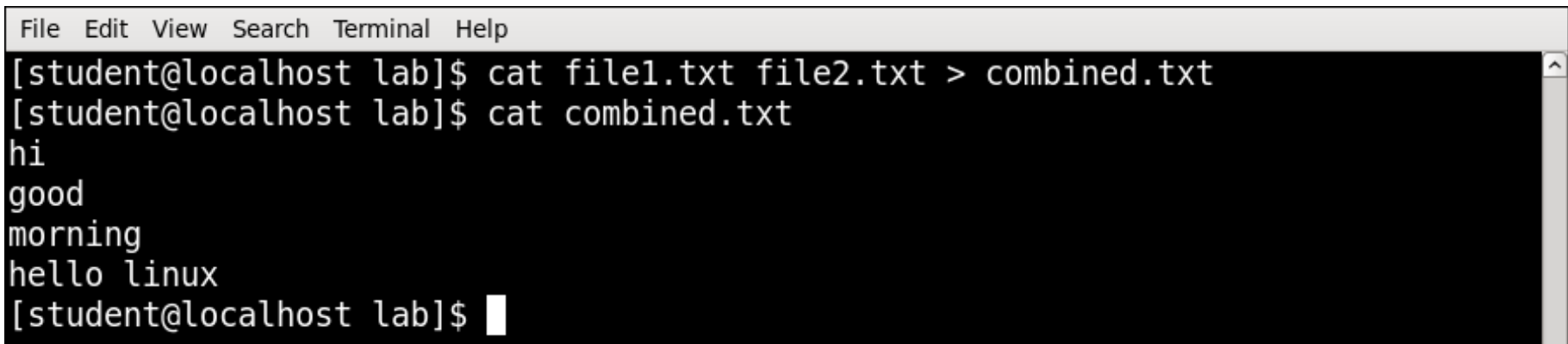
- It will read the contents of file1.txt and file2.txt and display the result in the terminal.

A terminal window with a menu bar (File, Edit, View, Search, Terminal, Help) and a dark background. The text shows a user at a localhost lab prompt running 'ls' to list files (file1.txt, file2.txt, newfile1.txt) and then 'cat file1.txt file2.txt' to concatenate and display the contents of file1.txt and file2.txt, which are 'hi', 'good', 'morning', and 'hello linux' respectively. The prompt returns to the shell.

```
File Edit View Search Terminal Help
[student@localhost lab]$ ls
file1.txt  file2.txt  newfile1.txt
[student@localhost lab]$ cat file1.txt file2.txt
hi
good
morning
hello linux
[student@localhost lab]$
```

cat Command Example

- **cat file1.txt file2.txt > combinedfile.txt**
- It will concatenate the contents of file1.txt and file2.txt and write them to a new file combinedfile.txt using the (>) operator.
- If the combinedfile.txt file doesn't exist the command will create it otherwise it will overwrite the file.

A terminal window with a menu bar (File, Edit, View, Search, Terminal, Help) and a dark background. The prompt is [student@localhost lab]\$. The command cat file1.txt file2.txt > combined.txt is entered. The prompt is [student@localhost lab]\$. The command cat combined.txt is entered. The output is hi, good, morning, hello linux. The prompt is [student@localhost lab]\$.

```
File Edit View Search Terminal Help
[student@localhost lab]$ cat file1.txt file2.txt > combined.txt
[student@localhost lab]$ cat combined.txt
hi
good
morning
hello linux
[student@localhost lab]$
```

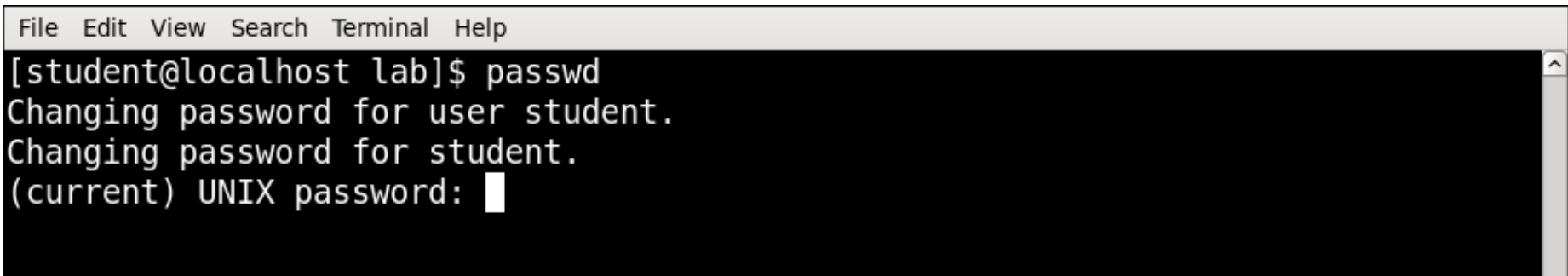
pwd (Print working directory) Command

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

- **Syntax :**

pwd [-OPTION]

- **Example :**

A terminal window with a menu bar containing 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. The terminal text shows a user at a shell prompt typing 'passwd', followed by two confirmation messages and a prompt for the current password.

```
File Edit View Search Terminal Help
[student@localhost lab]$ passwd
Changing password for user student.
Changing password for student.
(current) UNIX password: 
```

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 Command Example

File Edit View Search Terminal Help

```
[student@localhost lab]$ who
student  tty1          2020-03-24 10:48 (:0)
student  pts/0         2020-03-24 10:48 (:0.0)
[student@localhost lab]$
```

who -b

Display the time of the last system boot

File Edit View Search Terminal Help

```
[student@localhost lab]$ who -b
      system boot  2020-03-24 10:48
[student@localhost lab]$
```

who Command Example

who -H	Print a line of column headings
---------------	--

File Edit View Search Terminal Help

```
[student@localhost lab]$ who -H
```

NAME	LINE	TIME	COMMENT
student	tty1	2020-03-24 10:48	(:0)
student	pts/0	2020-03-24 10:48	(:0.0)

```
[student@localhost lab]$
```

who Command Example

who -q

Displays all login names, and a count of all logged-on users

File Edit View Search Terminal Help

```
[student@localhost lab]$ who -q
student student
# users=2
[student@localhost lab]$
```


who Command Example

who -a

Display all details of current logged in user

File Edit View Search Terminal Help

```
[student@localhost lab]$ who -a
      system boot  2020-03-24 10:48
      run-level 5  2020-03-24 10:48
LOGIN tty2        2020-03-24 10:48          1254 id=2
LOGIN tty3        2020-03-24 10:48          1256 id=3
LOGIN tty4        2020-03-24 10:48          1260 id=4
LOGIN tty5        2020-03-24 10:48          1263 id=5
LOGIN tty6        2020-03-24 10:48          1267 id=6
student - tty1    2020-03-24 10:48  old      1383 (:0)
student + pts/0   2020-03-24 10:48  .        1804 (:0.0)
[student@localhost lab]$
```

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 Command Example

whoami --help

Display a help message, and exit

File Edit View Search Terminal Help

```
[student@localhost lab]$ 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
```

```
Report whoami bugs to bug-coreutils@gnu.org
```

```
GNU coreutils home page: <http://www.gnu.org/software/coreutils/>
```

```
General help using GNU software: <http://www.gnu.org/gethelp/>
```

```
For complete documentation, run: info coreutils 'whoami invocation'
```

```
[student@localhost lab]$
```

whoami Command Example

whoami --version

Display version information, and exit

File Edit View Search Terminal Help

```
[student@localhost lab]$ whoami
student
[student@localhost lab]$ whoami --version
whoami (GNU coreutils) 8.5
Copyright (C) 2010 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 Richard Mlynarik.
[student@localhost lab]$
```

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 Command Example

uname -s	Print the kernel name
----------	-----------------------

File Edit View Search Terminal Help

```
[student@localhost lab]$ uname
```

```
Linux
```

```
[student@localhost lab]$ uname -s
```

```
Linux
```

```
[student@localhost lab]$
```

uname Command Example

uname -n

Print the network node hostname

File Edit View Search Terminal Help

```
[student@localhost lab]$ uname -n  
localhost.localdomain  
[student@localhost lab]$
```

uname Command Example

uname -v	Print the kernel version
-----------------	---------------------------------

File Edit View Search Terminal Help

```
[student@localhost lab]$ uname -v  
#1 SMP Mon Oct 18 23:56:17 UTC 2010  
[student@localhost lab]$
```


uname Command Example

uname -m	Print the machine hardware name
-----------------	--

File Edit View Search Terminal Help

```
[student@localhost lab]$ uname -m
```

```
i686
```

```
[student@localhost lab]$
```

uname Command Example

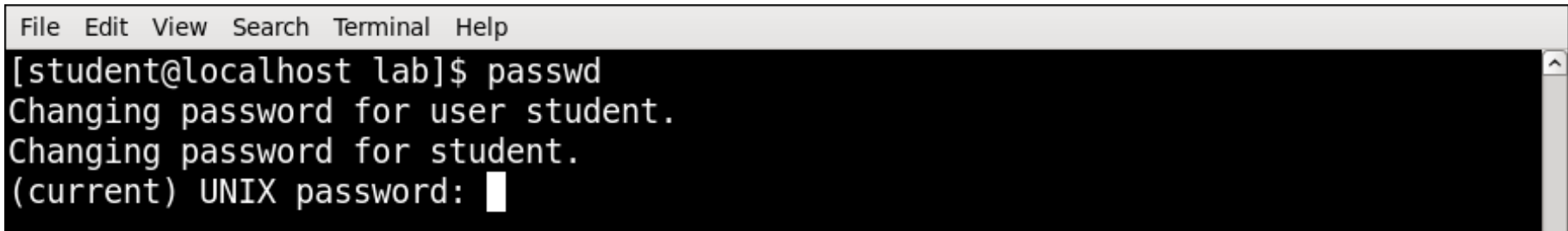
uname -o	Print the operating system
-----------------	-----------------------------------

File Edit View Search Terminal Help

```
[student@localhost lab]$ uname -o
GNU/Linux
[student@localhost lab]$
```

passwd Command

- The passwd command is used to change the password of a user account.
- **Syntax :**
passwd [-options] [username]
- **Example :**

A terminal window with a menu bar (File, Edit, View, Search, Terminal, Help) and a dark background. The text shows the command 'passwd' being executed as 'student@localhost lab', followed by two confirmation messages and a prompt for the current password.

```
File Edit View Search Terminal Help
[student@localhost lab]$ passwd
Changing password for user student.
Changing password for student.
(current) UNIX password: 
```

mkdir Command

- This command is used to make Directories.

- **Syntax :**

mkdir [-OPTION] DIRECTORY

- **Example :**

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

mkdir Command Example

mkdir -v

Print a message for each created directory

File Edit View Search Terminal Help

```
[student@localhost lab]$ mkdir lab-1
[student@localhost lab]$ ls
combined.txt  file1.txt  file2.txt  lab-1  newfile1.txt
[student@localhost lab]$ mkdir -v lab-2
mkdir: created directory `lab-2'
[student@localhost lab]$ ls
combined.txt  file1.txt  file2.txt  lab-1  lab-2  newfile1.txt
[student@localhost lab]$
```

mkdir Command Example

mkdir -p

No error if existing, make parent directories as needed

File Edit View Search Terminal Help

```
[student@localhost lab]$ ls
combined.txt  file1.txt  file2.txt  lab-1  lab-2  newfile1.txt
[student@localhost lab]$ mkdir lab-2
mkdir: cannot create directory `lab-2': File exists
[student@localhost lab]$ mkdir -p lab-2
[student@localhost lab]$
```

mkdir Command Example

mkdir -m	To control the permissions of new directories
-----------------	--

File Edit View Search Terminal Help

```
[student@localhost Documents]$ ls -l
total 8
drwxrwxr-x. 4 student student 4096 Mar 24 14:07 lab
drwxrwxr-x. 2 student student 4096 Mar 24 00:18 lab-1
[student@localhost Documents]$ mkdir -m 777 lab-2
[student@localhost Documents]$ ls -l
total 12
drwxrwxr-x. 4 student student 4096 Mar 24 14:07 lab
drwxrwxr-x. 2 student student 4096 Mar 24 00:18 lab-1
drwxrwxrwx. 2 student student 4096 Mar 24 14:12 lab-2
[student@localhost Documents]$
```

rmdir Command

- This command removes empty directories from your filesystem.

- **Syntax :**

`rmdir [-OPTION] DIRECTORY`

- **Example :**

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

rmdir Command Example

rmdir -p

Remove directory and its ancestors...

e.g., 'rmdir -p a/b/c' is similar to 'rmdir a/b/c a/b a'

File Edit View Search Terminal Help

```
[student@localhost Documents]$ ls
lab lab-1 lab-2 lab-3
[student@localhost Documents]$ rmdir lab-1
[student@localhost Documents]$ ls
lab lab-2 lab-3
[student@localhost Documents]$ rmdir lab-2 lab-3
[student@localhost Documents]$ ls
lab
[student@localhost Documents]$
```

File Edit View Search Terminal Help

```
[root@localhost ~]# mkdir -p /root/a/b/c
[root@localhost ~]# ls
a anaconda-ks.cfg install.log install.log.syslog
[root@localhost ~]# rmdir -p a/b/c
[root@localhost ~]# ls
anaconda-ks.cfg install.log install.log.syslog
[root@localhost ~]#
```

cp(copy) Command

- This command is used to copy files and directories.

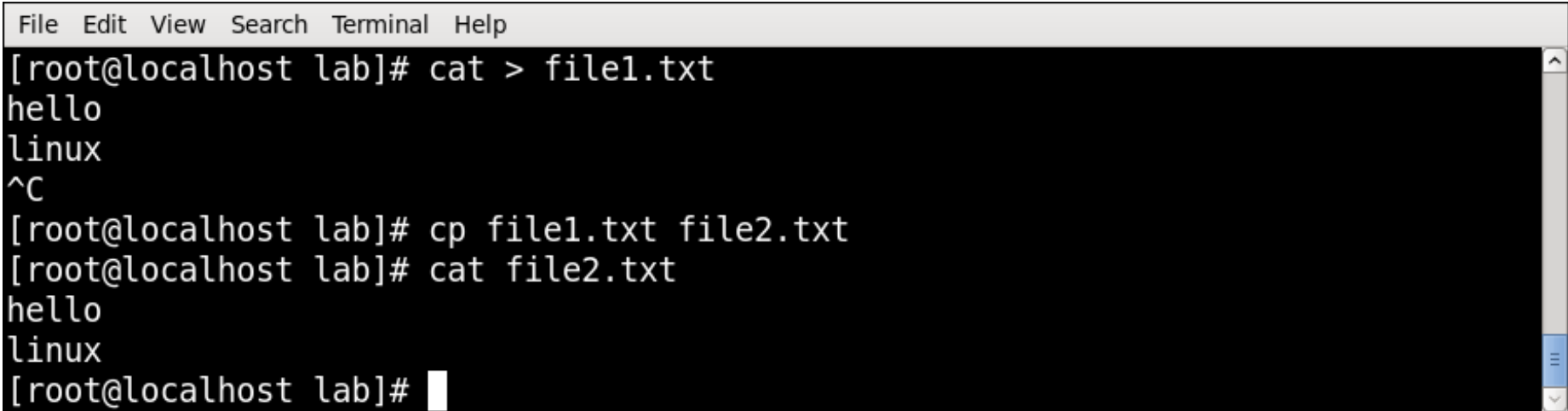
- **Syntax :**

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

cp Command Example

A terminal window with a menu bar (File, Edit, View, Search, Terminal, Help) and a dark background. The text is white. The commands and output are as follows:

```
[root@localhost lab]# cat > file1.txt
hello
linux
^C
[root@localhost lab]# cp file1.txt file2.txt
[root@localhost lab]# cat file2.txt
hello
linux
[root@localhost lab]#
```

The terminal window shows a sequence of commands and their outputs. First, the user runs `cat > file1.txt`, which prompts for input. The user enters `hello` and `linux`, then presses `^C` to stop the command. Next, the user runs `cp file1.txt file2.txt`, which copies the file. Finally, the user runs `cat file2.txt`, which displays the contents of the copied file: `hello` and `linux`. The prompt `[root@localhost lab]#` is shown at the end of each line.

cp Command Example

cp -i

Interactive - ask before overwrite

File Edit View Search Terminal Help

```
[root@localhost lab]# cp file1.txt file2.txt
[root@localhost lab]# cp -i file1.txt file2.txt
cp: overwrite `file2.txt'? yes
[root@localhost lab]# ls
file1.txt  file2.txt
[root@localhost lab]# cat file2.txt
hello
linux
[root@localhost lab]#
```

cp Command Example

cp -v

Print informative messages

File Edit View Search Terminal Help

```
[root@localhost lab]# cp -v file1.txt file3.txt
`file1.txt' -> `file3.txt'
[root@localhost lab]# cat file3.txt
hello
linux
[root@localhost lab]#
```

cp Command Example

cp -s

Make symbolic links instead of copying

File Edit View Search Terminal Help

```
[root@localhost lab]# cp -s file1.txt file3.txt
[root@localhost lab]# ls -l
total 8
-rw-r--r--. 1 root root 25 Mar 24 15:29 file1.txt
-rw-r--r--. 1 root root 25 Mar 24 15:30 file2.txt
lrwxrwxrwx. 1 root root 9 Mar 24 15:31 file3.txt -> file1.txt
[root@localhost lab]#
```

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

mv Command Example

```
File Edit View Search Terminal Help
[root@localhost lab]# ls
f1.txt  file1.txt  file2.txt
[root@localhost lab]# mv f1.txt ../lab-2/
[root@localhost lab]# ls
file1.txt  file2.txt
[root@localhost lab]# cd ..
[root@localhost Documents]# cd lab-2/
[root@localhost lab-2]# ls
f1.txt
[root@localhost lab-2]#
```


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

rm Command Example

rm -i

Prompt before every removal

File Edit View Search Terminal Help

```
[root@localhost lab]# ls
f1.txt f2.txt f5.txt file1.txt file2.txt files new.txt
[root@localhost lab]# rm f2.txt
rm: remove regular file `f2.txt'? y
[root@localhost lab]#
```

File Edit View Search Terminal Help

```
[root@localhost lab]# ls
file1.txt file2.txt file3.txt
[root@localhost lab]# rm -i *.txt
rm: remove regular file `file1.txt'? y
rm: remove regular file `file2.txt'? y
rm: remove regular file `file3.txt'? y
[root@localhost lab]# ls
[root@localhost lab]#
```

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 list of field. The default field to be used TAB. The default behavior can be overwritten by use of -d option
cut -d	Specifies a delimiter to be used as a field. Default field is TAB and this option overwrites this default behavior

cut Command Example

cut -c	Select only the characters from each line as specified in LIST
--------	--

```
[root@localhost ~]# cat data.txt
```

```
1 abc 12-12-2010 Rajkot
```

```
2 pqr 02-04-2011 Baroda
```

```
3 xyz 01-05-1998 Surat
```

```
[root@localhost ~]# cut -c 3 data.txt
```

```
a
```

```
p
```

```
x
```

```
[root@localhost ~]# cut -c 3-6 data.txt
```

```
abc
```

```
pqr
```

```
xyz
```

cut Command Example

cut -b	Select only the bytes from each line as specified in LIST
--------	---

```
[root@localhost ~]# cat data.txt
1 abc 12-12-2010 Rajkot
2 pqr 02-04-2011 Baroda
3 xyz 01-05-1998 Surat
[root@localhost ~]# cut -b 3 data.txt
a
p
x
```

cut Command Example

cut -f	Cuts the input file using list of field. The default field to be used TAB. The default behavior can be overwritten by use of -d option
cut -d	Specifies a delimiter to be used as a field. Default field is TAB and this option overwrites this default behavior

```
[root@localhost ~]# cat mydata.txt
1|abc|rajkot|20000
2|pqr|morbi|24000
3|xyz|surat|25000
[root@localhost ~]# cut -f 3 -d '|' mydata.txt
rajkot
morbi
surat
[root@localhost ~]# cut -f 2-3 -d '|' mydata.txt
abc|rajkot
pqr|morbi
xyz|surat
```

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

paste Command Example

paste -d	Reuse characters from LIST instead of tabs
----------	--

```
[test1990@server-1 ~]$cat empID.txt
1
2
3
4
[test1990@server-1 ~]$cat empName.txt
abc
pqr
xyz
demo
[test1990@server-1 ~]$paste - - < empName.txt
abc      pqr
xyz      demo
[test1990@server-1 ~]$paste -d':' empID.txt empName.txt
1:abc
2:pqr
3:xyz
4:demo
[test1990@server-1 ~]$paste -d'\n' empID.txt empName.txt
1
abc
2
pqr
3
xyz
4
demo
```


paste Command Example

paste -s	Paste one file at a time instead of in parallel
----------	---

```
[test1990@server-1 ~]$cat empID.txt
```

```
1
```

```
2
```

```
3
```

```
4
```

```
[test1990@server-1 ~]$cat empName.txt
```

```
abc
```

```
pqr
```

```
xyz
```

```
demo
```

```
[test1990@server-1 ~]$paste empID.txt empName.txt
```

```
1      abc
```

```
2      pqr
```

```
3      xyz
```

```
4      demo
```

```
[test1990@server-1 ~]$paste -s empID.txt empName.txt
```

```
1      2      3      4  
abc    pqr    xyz    demo
```

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

more Command Example

File Edit View Search Terminal Help

```
[root@localhost lab]# more file1.txt
```

```
1 ab
```

```
2 cd
```

```
3 ef
```

```
4 pq
```

```
5 rs
```

```
6 tu
```

```
7 xy
```

```
8 abc
```

```
9 pqr
```

```
10 xyz
```

```
[root@localhost lab]#
```

more Command Example

more -number

To Specify how many lines are printed in the screen for a given file

File Edit View Search Terminal Help

```
[root@localhost lab]# more -4 file1.txt
```

```
1 ab
```

```
2 cd
```

```
3 ef
```

```
4 pq
```

```
--More-- (37%)
```

File Edit View Search Terminal Help

```
[root@localhost lab]# more +4 file1.txt
```

```
4 pq
```

```
5 rs
```

```
6 tu
```

```
7 xy
```

```
8 abc
```

```
9 pqr
```

```
10 xyz
```

```
[root@localhost lab]#
```

more Command Example

more -c

Clear screen before displaying

File Edit View Search Terminal Help

```
[root@localhost lab]# more +4 file1.txt
```

```
4 pq
```

```
5 rs
```

```
6 tu
```

```
7 xy
```

```
8 abc
```

```
9 pqr
```

```
10 xyz
```

```
[root@localhost lab]# more -c file1.txt
```

```
1 ab
```

```
2 cd
```

```
3 ef
```

```
4 pq
```

```
5 rs
```

```
6 tu
```

```
7 xy
```

```
8 abc
```

```
9 pqr
```

```
10 xyz
```

```
[root@localhost lab]#
```

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]]]`

cmp Command 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
cmp -n	Compare at most LIMIT bytes
cmp -l	Print byte position and byte value for all differing bytes

```
File Edit View Search Terminal Help
[root@localhost lab]# cat > file1.txt
hi good morning
how r u
^C
[root@localhost lab]# cat > file2.txt
hello good morning
how r u
^C
[root@localhost lab]# cmp file1.txt file2.txt
file1.txt file2.txt differ: byte 2, line 1
[root@localhost lab]#
```

cmp Command Example

cmp -b	Print differing bytes
---------------	------------------------------

File Edit View Search Terminal Help

```
[root@localhost lab]# cat file1.txt
```

```
hi good morning
```

```
how r u
```

```
[root@localhost lab]# cat file2.txt
```

```
hello good morning
```

```
how r u
```

```
[root@localhost lab]# cmp -b file1.txt file2.txt
```

```
file1.txt file2.txt differ: byte 2, line 1 is 151 i 145 e
```

```
[root@localhost lab]#
```


cmp Command Example

cmp -i

Skip a particular number of initial bytes from both the files

File Edit View Search Terminal Help

```
[root@localhost lab]# cat file1.txt
hi linux good morning
[root@localhost lab]# cat file2.txt
hello hi good morning
[root@localhost lab]# cmp -i 8 file1.txt file2.txt
[root@localhost lab]# cmp -i 7 file1.txt file2.txt
file1.txt file2.txt differ: byte 1, line 1
[root@localhost lab]#
```

cmp Command Example

cmp -s

Do not print anything; only return an exit status indicating whether the files differ

File Edit View Search Terminal Help

```
[root@localhost lab]# cat file1.txt
hi linux good morning
[root@localhost lab]# cat file2.txt
hello hi good morning
[root@localhost lab]# cmp -s file1.txt file2.txt
[root@localhost lab]#
```

cmp Command Example

cmp -n	Compare at most LIMIT bytes
---------------	------------------------------------

File Edit View Search Terminal Help

```
[root@localhost lab]# cat f1.txt
hi hello linux
[root@localhost lab]# cat f2.txt
hi hello linux
[root@localhost lab]# cmp -n 3 f1.txt f2.txt
[root@localhost lab]#
```

cmp Command Example

cmp -l	Print byte position and byte value for all differing bytes
---------------	---

File Edit View Search Terminal Help

```
[root@localhost lab]# cmp -l file1.txt file2.txt
```

```
2 151 145
```

```
3 40 154
```

```
5 151 157
```

```
6 156 40
```

```
7 165 150
```

```
8 170 151
```

```
[root@localhost lab]# cmp -l f1.txt f2.txt
```

```
[root@localhost lab]#
```

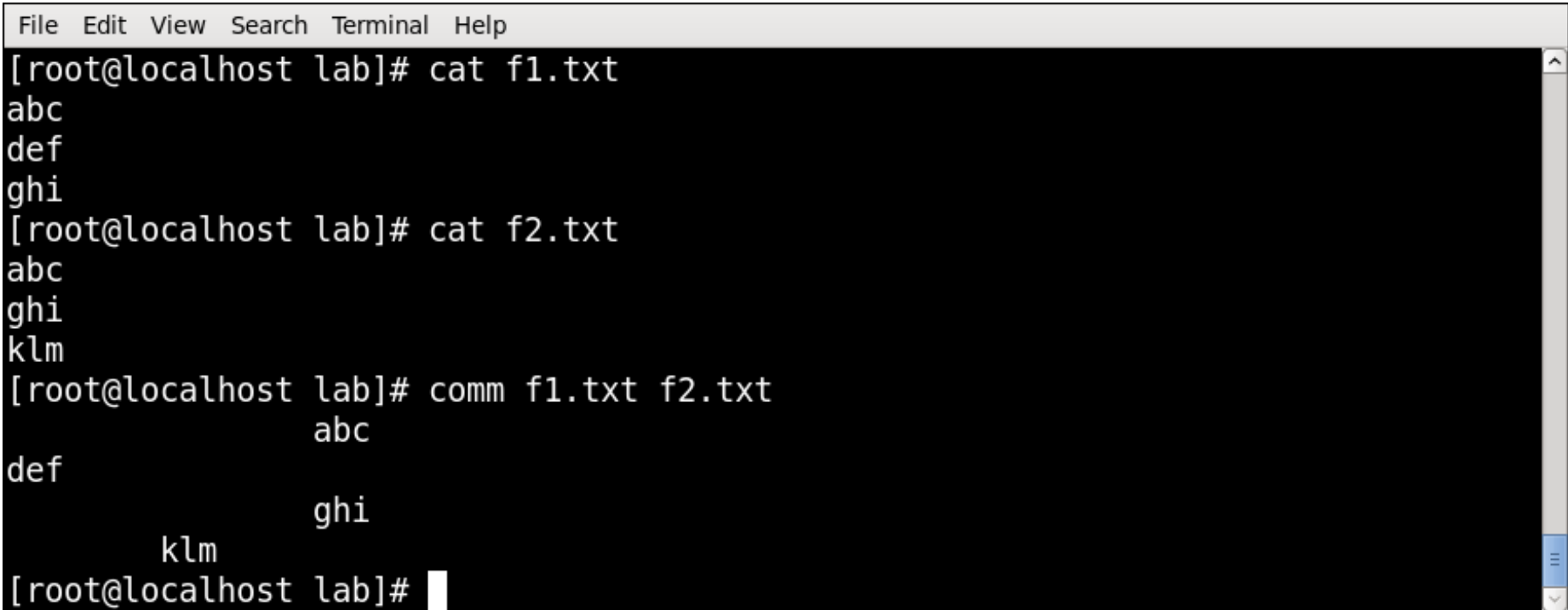
comm Command

- Compare two sorted files line by line.

- **Syntax :**

`comm [OPTION]... FILE1 FILE2`

- **Example :**

A terminal window with a menu bar (File, Edit, View, Search, Terminal, Help) and a dark background. It shows the execution of the 'comm' command to compare two files, f1.txt and f2.txt. The output shows lines unique to each file and lines common to both.

```
File Edit View Search Terminal Help
[root@localhost lab]# cat f1.txt
abc
def
ghi
[root@localhost lab]# cat f2.txt
abc
ghi
klm
[root@localhost lab]# comm f1.txt f2.txt
      abc
def
      ghi
      klm
[root@localhost lab]#
```

comm Command Example

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

```
File Edit View Search Terminal Help
[root@localhost lab]# cat f1.txt
abc
def
ghi
[root@localhost lab]# cat f2.txt
abc
ghi
klm
[root@localhost lab]# comm -1 f1.txt f2.txt
    abc
    ghi
klm
[root@localhost lab]# comm -2 f1.txt f2.txt
    abc
def
    ghi
[root@localhost lab]# comm -3 f1.txt f2.txt
def
    klm
[root@localhost lab]#
```

diff(difference) Command

- This command is used to display the differences in the files by comparing the files line by line
- diff analyzes 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

diff Command Example

```
File Edit View Search Terminal Help
[root@localhost lab]# cat > f1.txt
hello
good morning
all
^C
[root@localhost lab]# cat > f2.txt
hello
good morning
^C
[root@localhost lab]# diff f1.txt f2.txt
3d2
< all
[root@localhost lab]#
```

Special symbols are:

- a : add
- c : change
- d : delete

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

chmod(change mode) Command

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

chmod Command Example

- Each write, read, and execute permissions have following number value:

u	owner	r (read)	4
g	group	w (write)	2
o	others	x (execute)	1
a	all	no permissions	0

```
[root@localhost ~]# chmod 764 f2.txt
[root@localhost ~]# chmod u=rwx,g=rw,o=r f3.txt
[root@localhost ~]# ls -l
total 20
drwxrwxrwx    3 root    root          163 Aug 21  2011 dos
-rwxrw----    1 root    root           12 Jan 13  15:35 f1.txt
-rwxrw-r--    1 root    root           13 Jan 13  15:36 f2.txt
-rwxrw-r--    1 root    root           16 Jan 13  15:54 f3.txt
-rw-r--r--    1 root    root          242 Jul 15  2017 hello.c
```

chmod Command Example

```
[root@localhost ~]# ls -l
total 20
drwxr-xr-x    3 root    0           163 Aug 21  2011 dos
-rw-r--r--    1 root    0           12 Jan 13 15:35 f1.txt
-rw-r--r--    1 root    0           13 Jan 13 15:36 f2.txt
-rw-r--r--    1 root    0           16 Jan 13 15:54 f3.txt
-rw-r--r--    1 root    0          242 Jul 15  2017 hello.c
[root@localhost ~]# chmod 777 dos
[root@localhost ~]# ls -l
total 20
drwxrwxrwx    3 root    0           163 Aug 21  2011 dos
-rw-r--r--    1 root    0           12 Jan 13 15:35 f1.txt
-rw-r--r--    1 root    0           13 Jan 13 15:36 f2.txt
-rw-r--r--    1 root    0           16 Jan 13 15:54 f3.txt
-rw-r--r--    1 root    0          242 Jul 15  2017 hello.c
[root@localhost ~]# chmod u=rwx f1.txt
[root@localhost ~]# ls -l
total 20
drwxrwxrwx    3 root    0           163 Aug 21  2011 dos
-rwxr--r--    1 root    0           12 Jan 13 15:35 f1.txt
-rw-r--r--    1 root    0           13 Jan 13 15:36 f2.txt
-rw-r--r--    1 root    0           16 Jan 13 15:54 f3.txt
-rw-r--r--    1 root    0          242 Jul 15  2017 hello.c
```

chown(change owner) Command

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

- **Syntax :**

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

chown Command Example

```
File Edit View Search Terminal Help
[root@localhost Documents]# ls -l
total 20
drwxr-xr-x. 2 root    root    4096 Mar 24 18:59 lab
drwxrwxr-x. 4 student student 4096 Mar 24 15:42 lab-1
drwxr-xr-x. 2 root    root    4096 Mar 24 15:44 lab-2
drwxr-xr-x. 2 root    root    4096 Mar 24 15:41 lab-3
drwxrwxr-x. 2 student student 4096 Mar 24 19:43 lab-4
[root@localhost Documents]# chown root lab-4
[root@localhost Documents]# ls -l
total 20
drwxr-xr-x. 2 root    root    4096 Mar 24 18:59 lab
drwxrwxr-x. 4 student student 4096 Mar 24 15:42 lab-1
drwxr-xr-x. 2 root    root    4096 Mar 24 15:44 lab-2
drwxr-xr-x. 2 root    root    4096 Mar 24 15:41 lab-3
drwxrwxr-x. 2 root    student 4096 Mar 24 19:43 lab-4
[root@localhost Documents]#
```

chgrp(change group) Command

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

- **Syntax :**

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

chgrp Command Example

```
File Edit View Search Terminal Help
[root@localhost Documents]# ls -l
total 20
drwxr-xr-x. 2 root    root    4096 Mar 24 18:59 lab
drwxrwxr-x. 4 student student 4096 Mar 24 15:42 lab-1
drwxr-xr-x. 2 root    root    4096 Mar 24 15:44 lab-2
drwxr-xr-x. 2 root    root    4096 Mar 24 15:41 lab-3
drwxrwxr-x. 2 root    student 4096 Mar 24 19:43 lab-4
[root@localhost Documents]# chgrp root lab-4
[root@localhost Documents]# ls -l
total 20
drwxr-xr-x. 2 root    root    4096 Mar 24 18:59 lab
drwxrwxr-x. 4 student student 4096 Mar 24 15:42 lab-1
drwxr-xr-x. 2 root    root    4096 Mar 24 15:44 lab-2
drwxr-xr-x. 2 root    root    4096 Mar 24 15:41 lab-3
drwxrwxr-x. 2 root    root    4096 Mar 24 19:43 lab-4
[root@localhost Documents]#
```

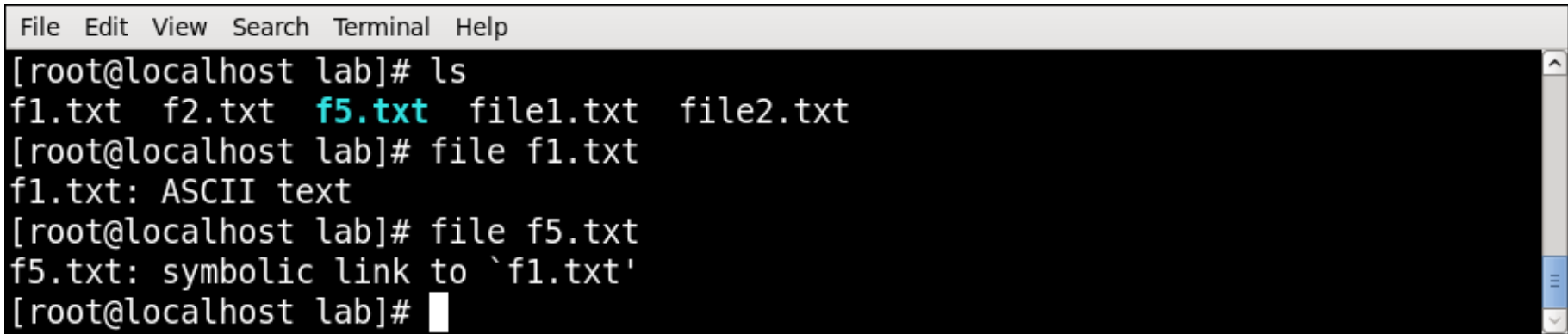

chgrp Command Example

File Edit View Search Terminal Help

```
[root@localhost lab]# ls -l
total 16
-rw-r--r--. 1 root root 23 Mar 24 19:33 f1.txt
-rw-r--r--. 1 student student 19 Mar 24 19:34 f2.txt
-rw-r--r--. 1 root root 22 Mar 24 18:16 file1.txt
-rw-r--r--. 1 root root 22 Mar 24 18:16 file2.txt
[root@localhost lab]# chown student f1.txt
[root@localhost lab]# chgrp student f1.txt
[root@localhost lab]# ls -l
total 16
-rw-r--r--. 1 student student 23 Mar 24 19:33 f1.txt
-rw-r--r--. 1 student student 19 Mar 24 19:34 f2.txt
-rw-r--r--. 1 root root 22 Mar 24 18:16 file1.txt
-rw-r--r--. 1 root root 22 Mar 24 18:16 file2.txt
[root@localhost lab]#
```

file Command

- The file command is used to determine a file's type.
- **Syntax :**
file [OPTIONS] file1 file2 ...
- **Example :**

A screenshot of a Linux terminal window. The window has a title bar with menu items: File, Edit, View, Search, Terminal, and Help. The terminal content shows a user at the root@localhost lab prompt. They run 'ls' and see a list of files: f1.txt, f2.txt, f5.txt (highlighted in cyan), file1.txt, and file2.txt. Then they run 'file f1.txt' and get the output 'f1.txt: ASCII text'. Next, they run 'file f5.txt' and get the output 'f5.txt: symbolic link to `f1.txt`'. The prompt is now ready for another command.

```
File Edit View Search Terminal Help
[root@localhost lab]# ls
f1.txt f2.txt f5.txt file1.txt file2.txt
[root@localhost lab]# file f1.txt
f1.txt: ASCII text
[root@localhost lab]# file f5.txt
f5.txt: symbolic link to `f1.txt`
[root@localhost lab]#
```

file Command Example

file -i

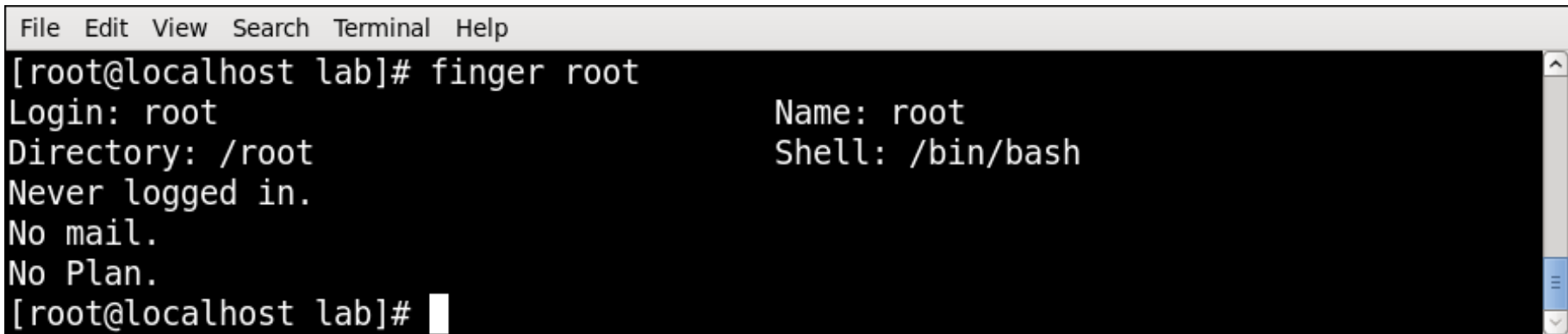
To view the mime type of a file rather than the human readable format

File Edit View Search Terminal Help

```
[root@localhost lab]# file -i f1.txt
f1.txt: text/plain; charset=us-ascii
[root@localhost lab]# file -i f5.txt
f5.txt: application/x-symlink; charset=binary
[root@localhost lab]#
```

finger Command

- finger looks up and displays information about system users.
- **Syntax :**
finger [-option] [username]
- **Example :**



```
File Edit View Search Terminal Help
[root@localhost lab]# finger root
Login: root                               Name: root
Directory: /root                         Shell: /bin/bash
Never logged in.
No mail.
No Plan.
[root@localhost lab]#
```

A terminal window with a menu bar (File, Edit, View, Search, Terminal, Help) and a dark background. The command `finger root` has been executed, showing user information for 'root' in two columns. The output includes login name, directory, login status, mail status, and plan status. The prompt `[root@localhost lab]#` is visible at the end of the line.

finger Command Example

finger -m	Match arguments only on user name (not first or last name)
------------------	---

File Edit View Search Terminal Help

```
[student@localhost ~]$ finger -m
Login      Name      Tty      Idle  Login Time   Office   Office Phone
Host
student    student   tty1      Mar 30 10:03
(:0)
student    student   pts/0      Mar 30 10:03
(:0.0)
[student@localhost ~]$
```

finger Command Example

finger -l	Force long output format
------------------	---------------------------------

File Edit View Search Terminal Help

```
[student@localhost ~]$ finger -l
Login: student                               Name: student
Directory: /home/student                     Shell: /bin/bash
On since Mon Mar 30 10:03 (IST) on tty1 from :0
      1 minute 18 seconds idle
On since Mon Mar 30 10:03 (IST) on pts/0 from :0.0
No mail.
No Plan.
[student@localhost ~]$
```

sleep Command

- The sleep command is used to delay for a specified amount of time.
- **Syntax :**
sleep NUMBER[SUFFIX]...
- **Example :**

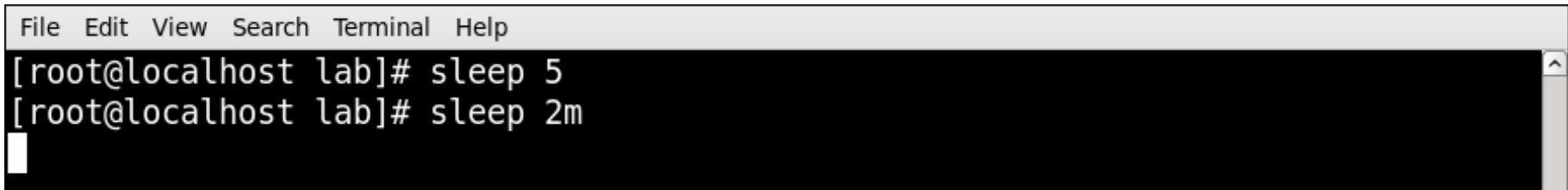
A terminal window with a menu bar (File, Edit, View, Search, Terminal, Help) and a dark background. The prompt is [root@localhost lab]#. The command sleep 5 has been entered, and a white cursor is visible on the line below.

```
File Edit View Search Terminal Help
[root@localhost lab]# sleep 5

```

sleep Command Example

- s for seconds; this is a default one if you don't specify any letter after the integer.
- m for minutes.
- h for hours.
- d for days.

A terminal window with a menu bar (File, Edit, View, Search, Terminal, Help) and a dark background. It shows two commands being entered: '[root@localhost lab]# sleep 5' and '[root@localhost lab]# sleep 2m'. A white cursor is visible on the line following the second command.

```
File Edit View Search Terminal Help
[root@localhost lab]# sleep 5
[root@localhost lab]# sleep 2m

```


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

ps Command Example

ps -e

Display every active process on a Linux system in generic (Unix/Linux) format

File Edit View Search Terminal Help

[root@localhost lab]# ps

PID	TTY	TIME	CMD
4176	pts/0	00:00:00	su
4185	pts/0	00:00:00	bash
4349	pts/0	00:00:00	ps

[root@localhost lab]#

File Edit View Search Terminal Help

[root@localhost lab]# ps -e

PID	TTY	TIME	CMD
1	?	00:00:00	init
2	?	00:00:00	kthreadd
3	?	00:00:00	ksoftirqd/0
4	?	00:00:00	migration/0
5	?	00:00:00	watchdog/0

ps Command Example

ps -x

View all processes owned by you

File Edit View Search Terminal Help

```
[root@localhost lab]# ps -x
```

```
Warning: bad syntax, perhaps a bogus '-'? See /usr/share/doc/procps-3.2.8/FAQ
```

PID	TTY	STAT	TIME	COMMAND
1	?	Ss	0:00	/sbin/init
2	?	S	0:00	[kthreadd]
3	?	S	0:00	[ksoftirqd/0]
4	?	S	0:00	[migration/0]
5	?	S	0:00	[watchdog/0]
6	?	S	0:02	[events/0]
7	?	S	0:00	[cpuset]
8	?	S	0:00	[khelper]

ps Command Example

ps -f

To provide more information on processes

File Edit View Search Terminal Help

```
[root@localhost lab]# ps -f
```

UID	PID	PPID	C	STIME	TTY	TIME	CMD
root	4176	4160	0	19:45	pts/0	00:00:00	su
root	4185	4176	0	19:45	pts/0	00:00:00	bash
root	4400	4185	0	20:03	pts/0	00:00:00	ps -f

```
[root@localhost lab]#
```

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

kill Command Example

```
File Edit View Search Terminal Help
[root@localhost Documents]# ps
  PID TTY          TIME CMD
 4454 pts/0        00:00:00 su
 4463 pts/0        00:00:00 bash
 4476 pts/0        00:00:00 ps
[root@localhost Documents]# kill 4454
[root@localhost Documents]#
Session terminated, killing shell... ..killed.
[student@localhost Documents]$  PID TTY          TIME CMD
 4160 pts/0        00:00:00 bash
 4477 pts/0        00:00:00 ps
[student@localhost Documents]$
```

kill Command Example

Kill -l	To display all the available signals
---------	--------------------------------------

File Edit View Search Terminal Help

```
[student@localhost ~]$ 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