Linux files and directories

Tushar B. Kute,
http://tusharkute.com

Linux files

- You can name a file using any letters, underscores, and numbers. You can also include periods and commas. Except in certain special cases, you should never begin a filename with a period.
- Other characters, such as slashes, question marks, or asterisks, are reserved for use as special characters by the system and should not be part of a filename.
- Filenames can be as long as 256 characters. Filenames can also include spaces, though to reference such filenames from the command line, be sure to encase them in quotes. On a desktop like GNOME or KDE, you do not need quotes.

Linux files

- Special initialization files are also used to hold shell configuration commands.
- These are the hidden, or dot, files, which begin with a period. Dot files used by commands and applications have predetermined names, such as the .mozilla directory used to hold your Mozilla data and configuration files.
- Recall that when you use **`ls`** to display your filenames, the dot files will not be displayed. To include the dot files, you need to use ls with the -a option.

Linux files

- The ls -l command displays detailed information about a file.
- First the permissions are displayed, followed by the number of links, the owner of the file, the name of the group the user belongs to, the file size in bytes, the date and time the file was last modified, and the name of the file.
- Permissions indicate who can access the file: the user, members of a group, or all other users.

Linux files

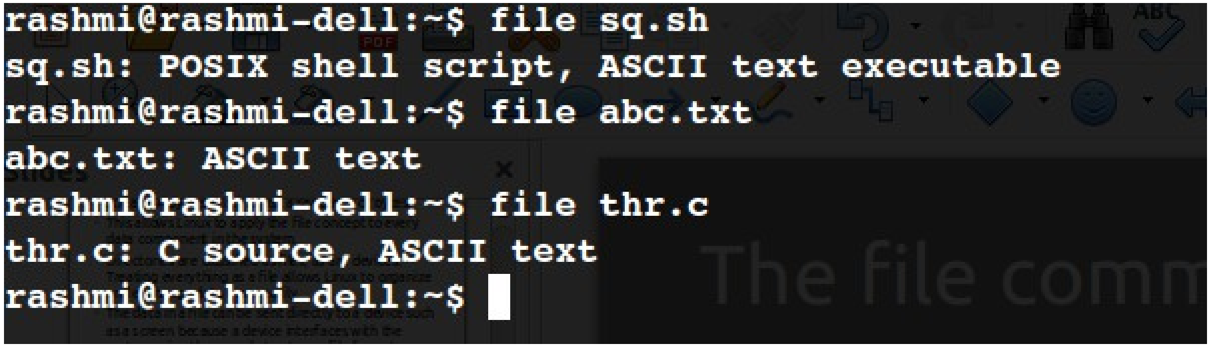
- All files in Linux have one physical format—a byte stream. A byte stream is just a sequence of bytes.
- This allows Linux to apply the file concept to every data component in the system.
- Directories are classified as files, as are devices. Treating everything as a file allows Linux to organize and exchange data more easily.
- The data in a file can be sent directly to a device such as a screen because a device interfaces with the system using the same byte-stream file format as regular files.

Linux files types

- Five common types of files exist:
  - ordinary files,
  - directory files,
  - first-in first-out pipes,
  - character device files, and
  - block device files.

The file command

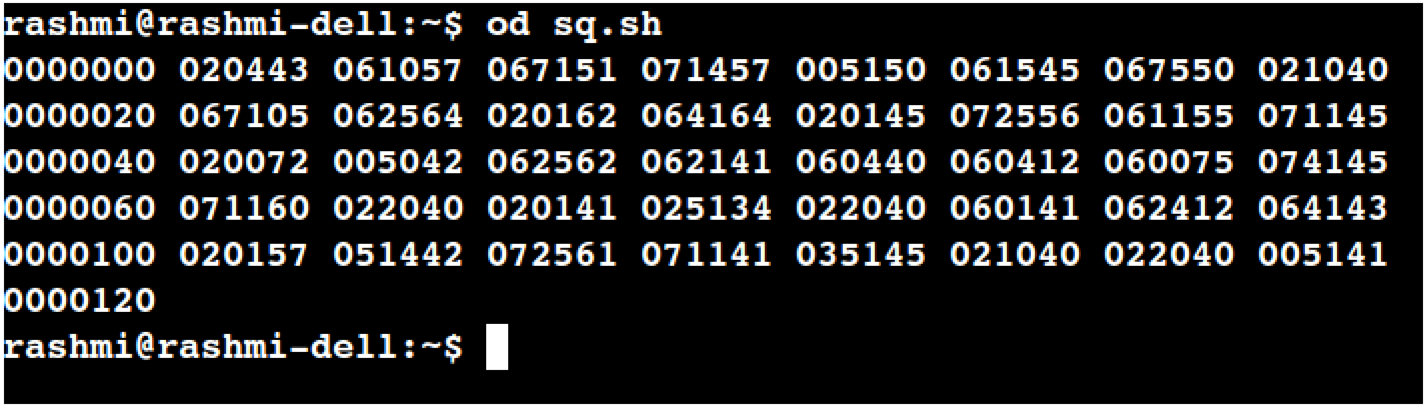
- The Linux file command helps you determine what a file is used for.
- It examines the first few lines of a file and tries to determine a classification for it.
- The file command looks for special keywords or special numbers in those first few lines, but it is not always accurate.



```
rashmi@rashmi-dell:~$ file sq.sh
sq.sh: POSIX shell script, ASCII text executable
rashmi@rashmi-dell:~$ file abc.txt
abc.txt: ASCII text
rashmi@rashmi-dell:~$ file thr.c
thr.c: C source, ASCII text
rashmi@rashmi-dell:~$ 
```

The od command

- If you need to examine the entire file byte by byte, you can do so with the od (octal dump) command.
- The od command performs a dump of a file. By default, it prints every byte in its octal representation. However, you can also specify a character, decimal, or hexadecimal representation.
- The od command is helpful when you need to detect any special character in your file or if you want to display a binary file.

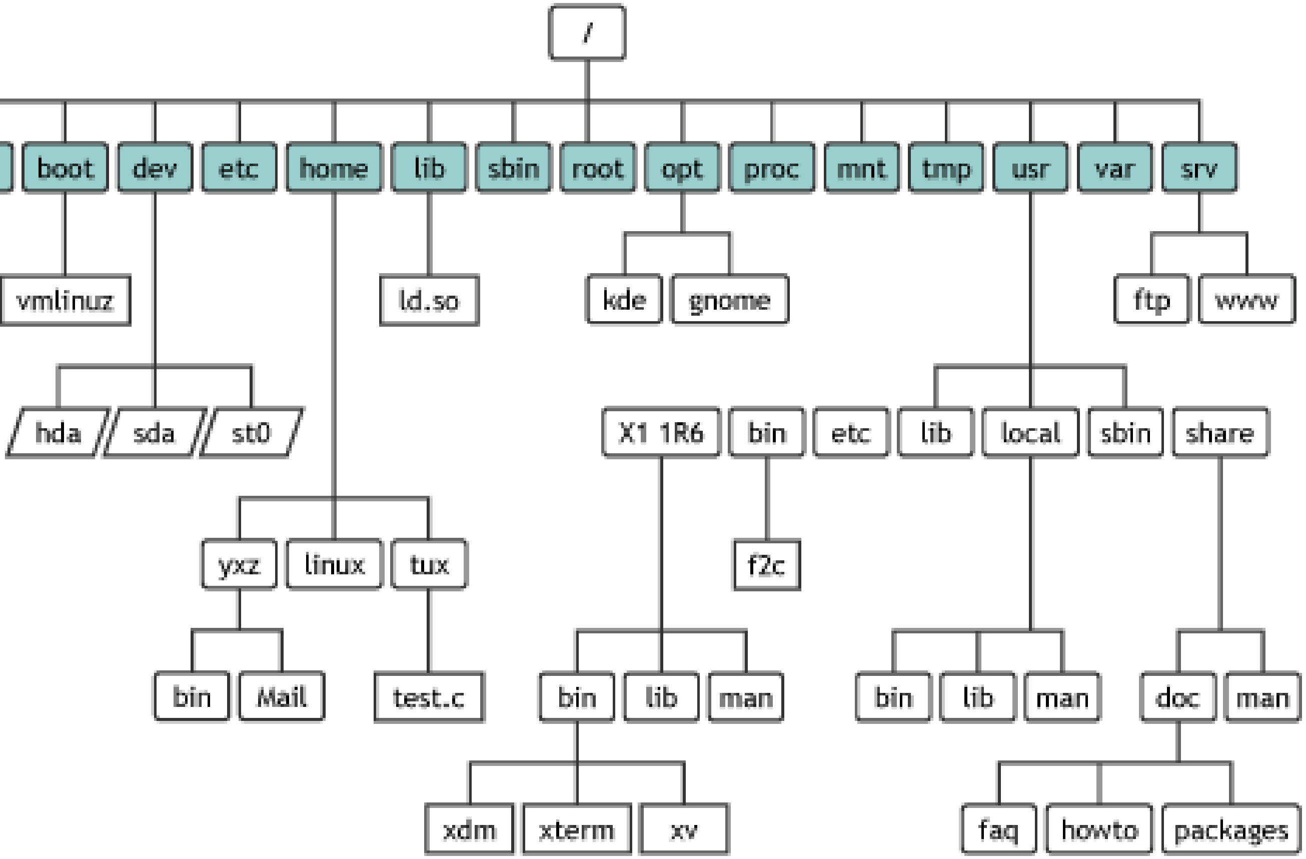


```
rashmi@rashmi-dell:~$ od sq.sh
0000000 020443 061057 067151 071457 005150 061545 067550 021040
0000020 067105 062564 020162 064164 020145 072556 061155 071145
0000040 020072 005042 062562 062141 060440 060412 060075 074145
0000060 071160 022040 020141 025134 022040 060141 062412 064143
0000100 020157 051442 072561 071141 035145 021040 022040 005141
0000120
rashmi@rashmi-dell:~$ 
```

The file structure

- Linux organizes files into a hierarchically connected set of directories.
- Each directory may contain either files or other directories. In this respect, directories perform two important functions.
- A directory holds files, much like files held in a file drawer, and a directory connects to other directories, much as a branch in a tree is connected to other branches.
- Because of the similarities to a tree, such a structure is often referred to as a tree structure.

The file structure



Home directory

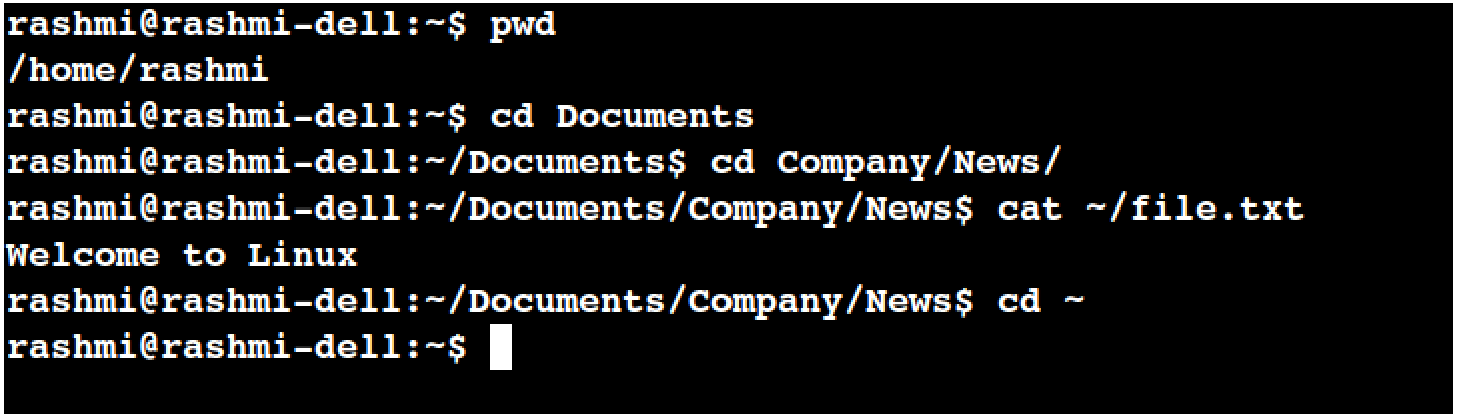
- When you log in to the system, you are placed within your home directory.
- The name given to this directory by the system is the same as your login name.
- Any files you create when you first log in are organized within your home directory.
- Within your home directory, however, you can create more directories.
- You can change the working directory by using the cd command to designate another directory as the working directory.

Pathnames

- The name you give to a directory or file when you create it is not its full name.
- The full name of a directory is its pathname.
- In Linux, you write a pathname by listing each directory in the path separated from the last by a forward slash.
- A slash preceding the first directory in the path represents the root.
- The pathname for the rashmi directory is /home/rashmi. The pathname for the reports directory is /home/rashmi/reports.

Pathnames

- The absolute pathname from the root to your home directory can be especially complex and, at times, even subject to change by the system administrator.
- To make it easier to reference, you can use a special character, the tilde (~), which represents the absolute pathname of your home directory.

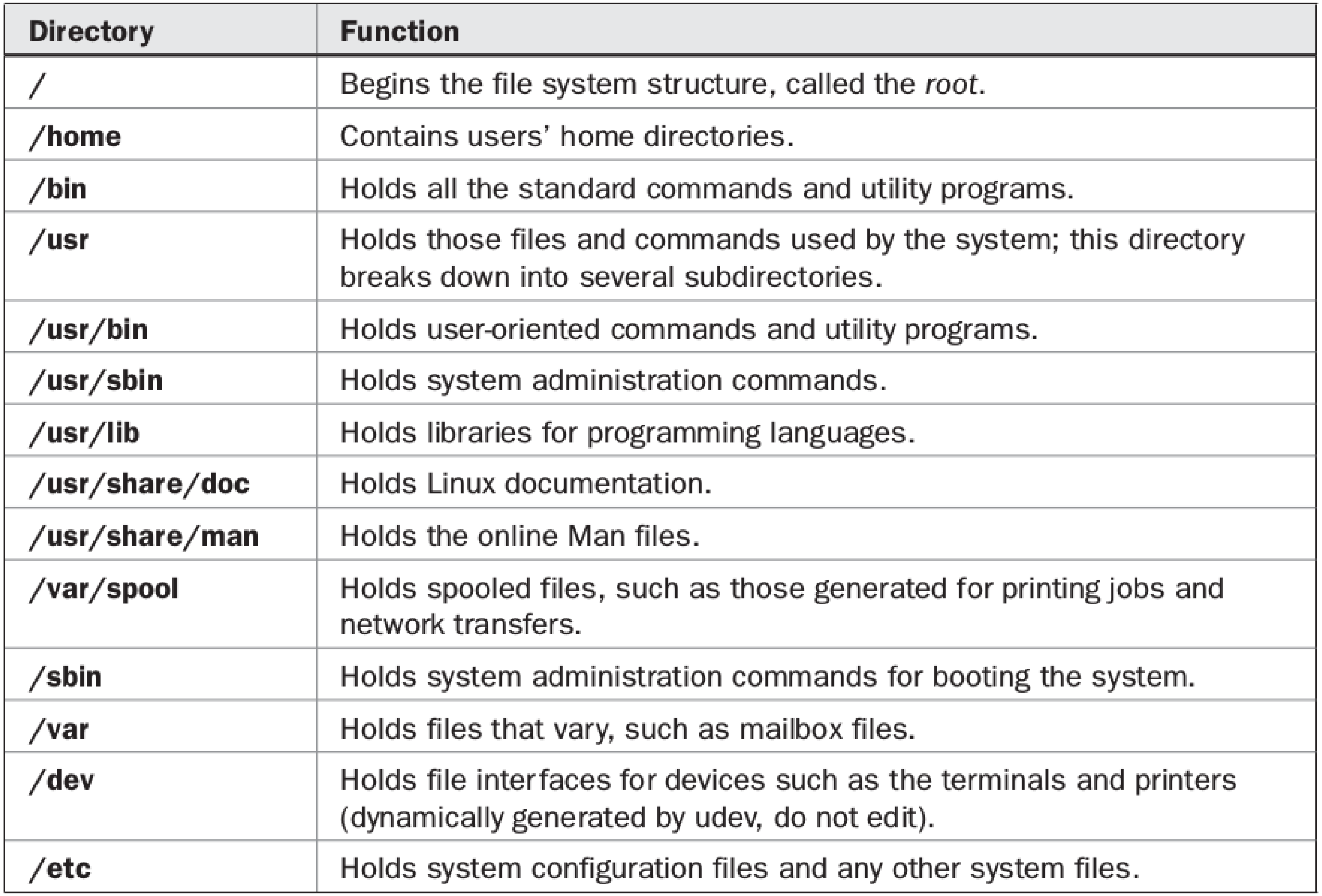


```
rashmi@rashmi-dell:~$ pwd
/home/rashmi
rashmi@rashmi-dell:~$ cd Documents
rashmi@rashmi-dell:~/Documents$ cd Company/News/
rashmi@rashmi-dell:~/Documents/Company/News$ cat ~/file.txt
Welcome to Linux
rashmi@rashmi-dell:~/Documents/Company/News$ cd ~
rashmi@rashmi-dell:~$ 
```

System Directories

- The root directory that begins the Linux file structure contains several system directories.
- The system directories contain files and programs used to run and maintain the system.
- Many contain other subdirectories with programs for executing specific features of Linux.
- For example, the directory /usr/bin contains the various Linux commands that users execute, such as lpl.
- The directory /bin holds system-level commands.

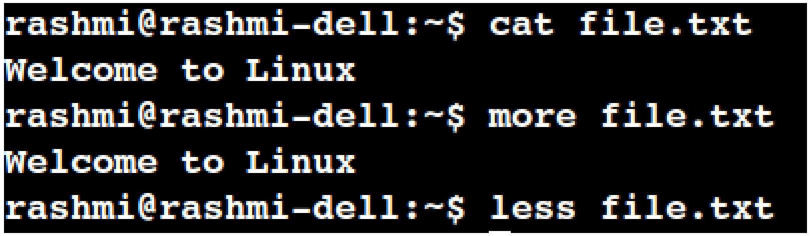
Standard system directories



Directory	Function
/	Begins the file system structure, called the *root*.
/home	Contains users' home directories.
/bin	Holds all the standard commands and utility programs.
/usr	Holds those files and commands used by the system; this directory breaks down into several subdirectories.
/usr/bin	Holds user-oriented commands and utility programs.
/usr/sbin	Holds system administration commands.
/usr/lib	Holds libraries for programming languages.
/usr/share/doc	Holds Linux documentation.
/usr/share/man	Holds the online Man files.
/var/spool	Holds spooled files, such as those generated for printing jobs and network transfers.
/sbin	Holds system administration commands for booting the system.
/var	Holds files that vary, such as mailbox files.
/dev	Holds file interfaces for devices such as the terminals and printers (dynamically generated by udev, do not edit).
/etc	Holds system configuration files and any other system files.

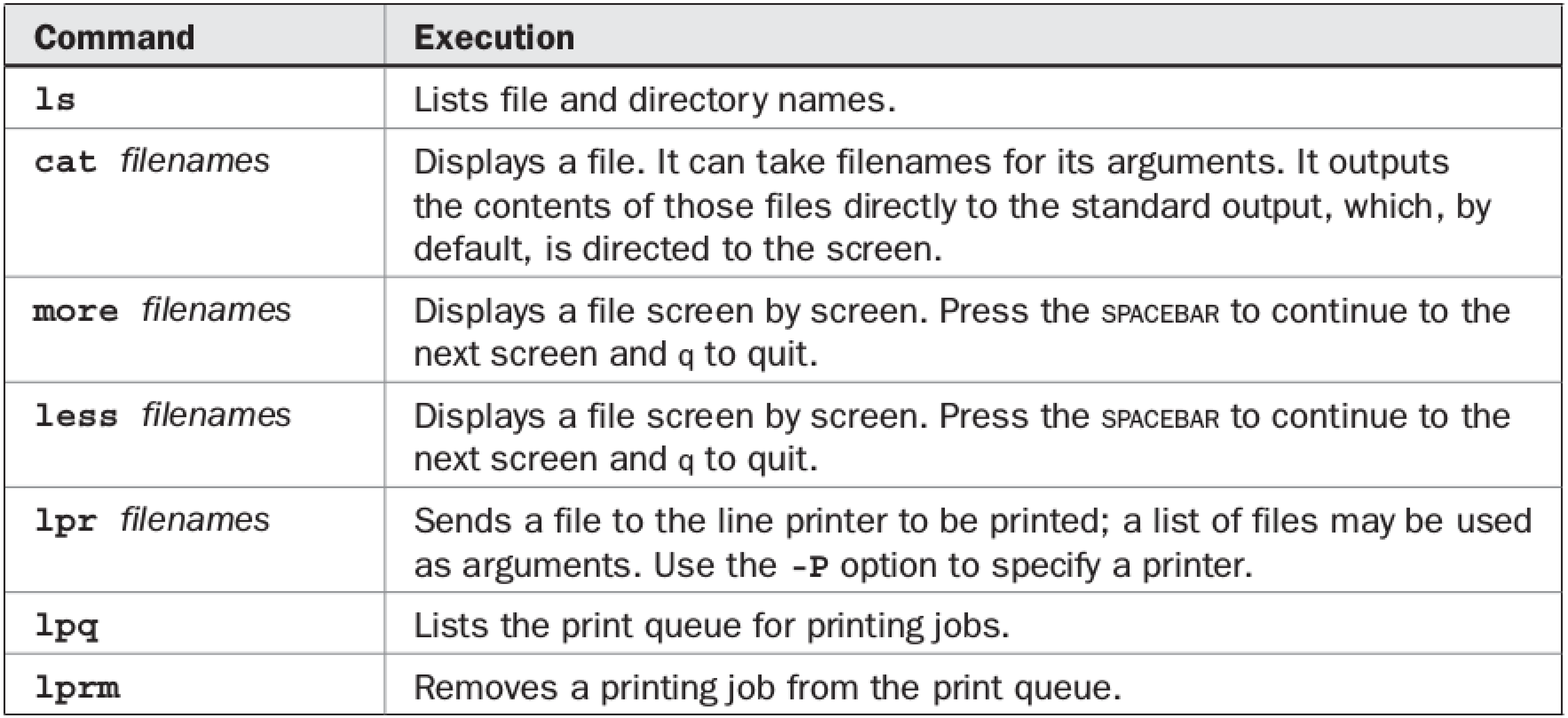
Displaying file contents

- The **`cat`** and **`more`** commands display the contents of a file on the screen. The name cat stands for concatenate.
- The cat command outputs the entire text of a file to the screen at once. This presents a problem when the file is large because its text quickly speeds past on the screen.
- The more and less commands are designed to overcome this limitation by displaying one screen of text at a time.
- You can then move forward or backward in the text at your leisure.



```
rashmi@rashmi-dell:~$ cat file.txt
Welcome to Linux
rashmi@rashmi-dell:~$ more file.txt
Welcome to Linux
rashmi@rashmi-dell:~$ less file.txt
```

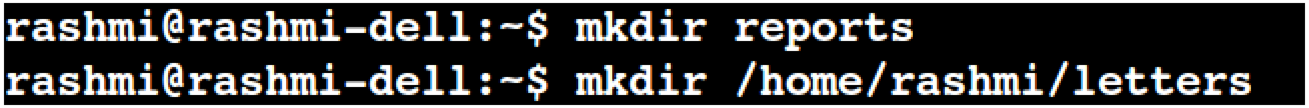
Listing, Displaying and Printing files



Command	Execution
`ls`	Lists file and directory names.
`cat` *filenames*	Displays a file. It can take filenames for its arguments. It outputs the contents of those files directly to the standard output, which, by default, is directed to the screen.
`more` *filenames*	Displays a file screen by screen. Press the SPACEBAR to continue to the next screen and q to quit.
`less` *filenames*	Displays a file screen by screen. Press the SPACEBAR to continue to the next screen and q to quit.
`lpr` *filenames*	Sends a file to the line printer to be printed; a list of files may be used as arguments. Use the `-P` option to specify a printer.
`lpq`	Lists the print queue for printing jobs.
`lprm`	Removes a printing job from the print queue.

Creating directory

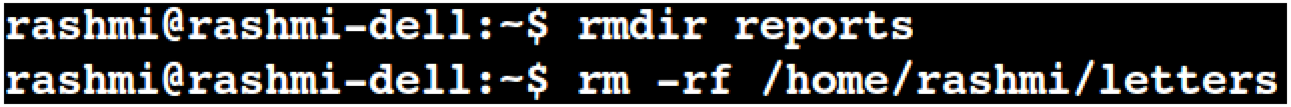
- You create and remove directories with the mkdir and rmdir commands.
- In either case, you can also use pathnames for the directories.
- In the next example, the user creates the directory *reports*. Then the user creates the directory *letters* using a pathname:



```
rashmi@rashmi-dell:~$ mkdir reports
rashmi@rashmi-dell:~$ mkdir /home/rashmi/letters
```

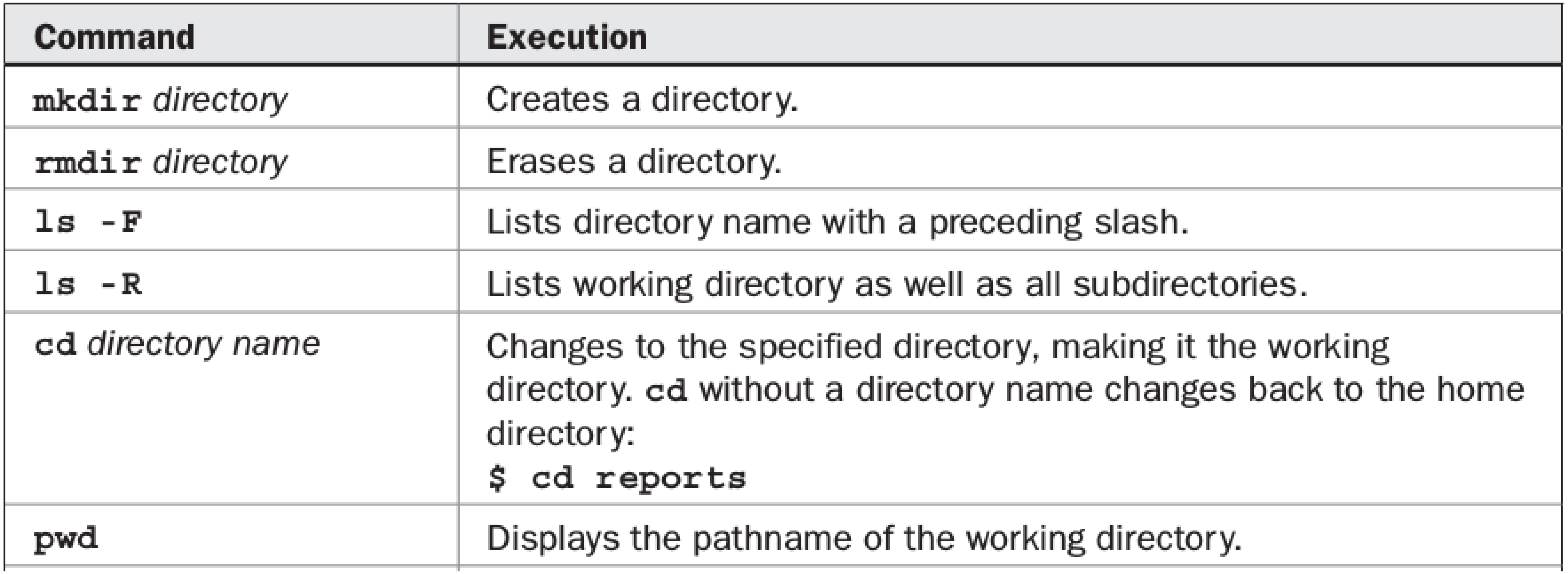
Removing directory

- You can remove a directory with the **`rmdir`** command followed by the directory name.
- To remove a directory and all its subdirectories, you use the **`rm`** command with the **`-r`** option.
- This is a very powerful command and can easily be used to erase all your files. If your rm command is aliased as **`rm -i`** (interactive mode), you will be prompted for each file.
- To simply remove all files and subdirectories without prompts, add the **`-f`** option.



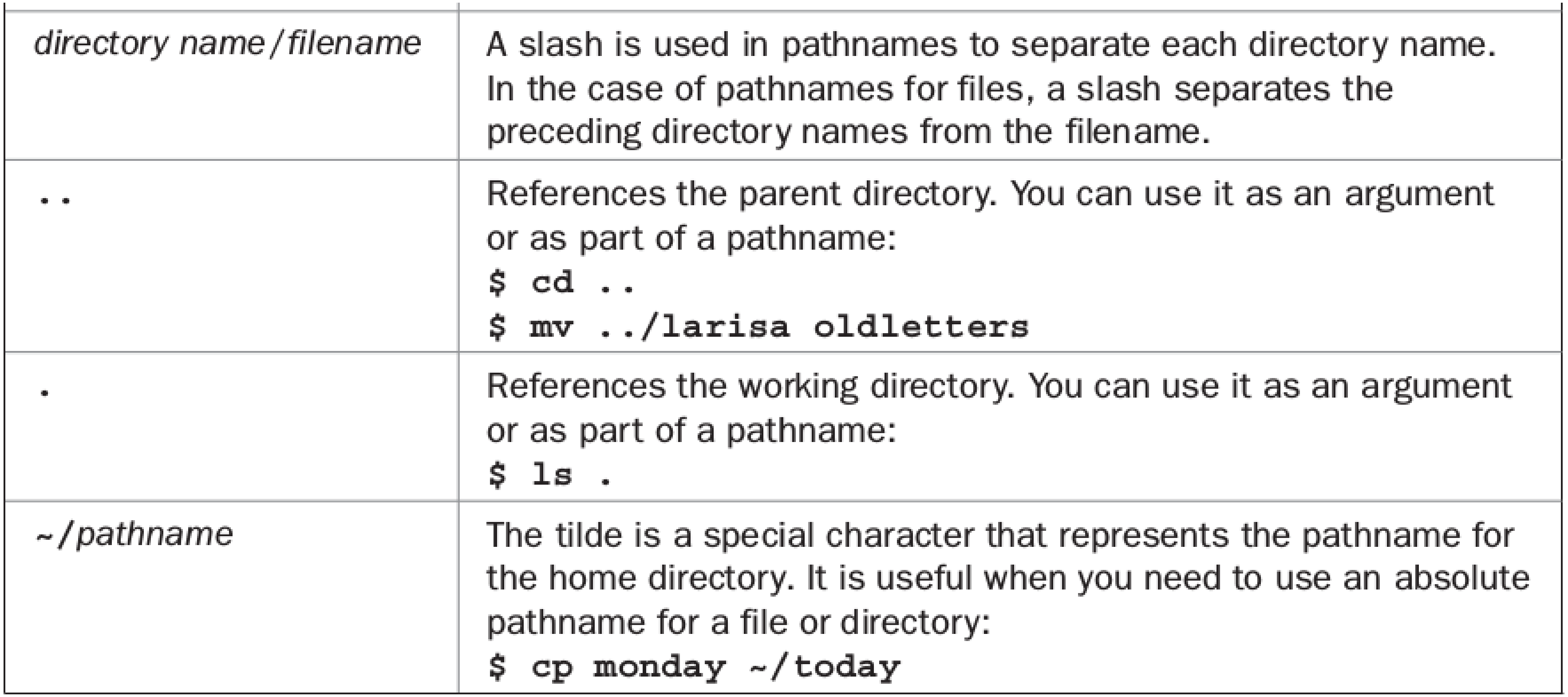
```
rashmi@rashmi-dell:~$ rmdir reports
rashmi@rashmi-dell:~$ rm -rf /home/rashmi/letters
```

Directory Commands



Command	Execution
`mkdir` *directory*	Creates a directory.
`rmdir` *directory*	Erases a directory.
`ls -F`	Lists directory name with a preceding slash.
`ls -R`	Lists working directory as well as all subdirectories.
`cd` *directory name*	Changes to the specified directory, making it the working directory. `cd` without a directory name changes back to the home directory: `$ cd reports`
`pwd`	Displays the pathname of the working directory.

Directory Commands



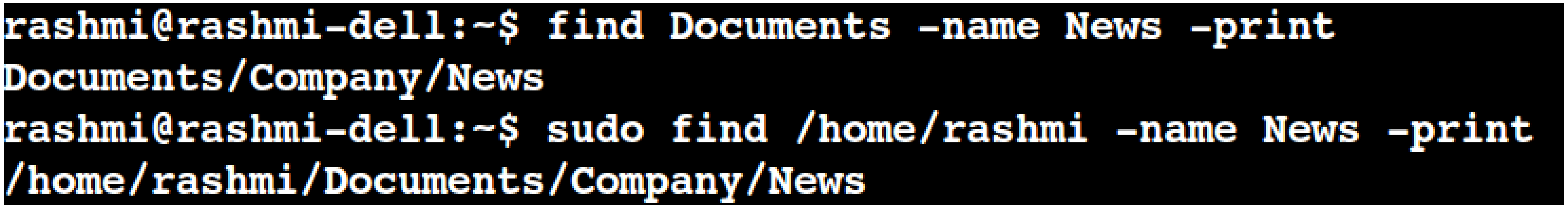
directory name/filename	A slash is used in pathnames to separate each directory name. In the case of pathnames for files, a slash separates the preceding directory names from the filename.

`..`	References the parent directory. You can use it as an argument or as part of a pathname:
	`$ cd ..`
	`$ mv ../larisa oldletters`

`.`	References the working directory. You can use it as an argument or as part of a pathname: `$ ls .`
~/pathname	The tilde is a special character that represents the pathname for the home directory. It is useful when you need to use an absolute pathname for a file or directory: `$ cp monday ~/today`

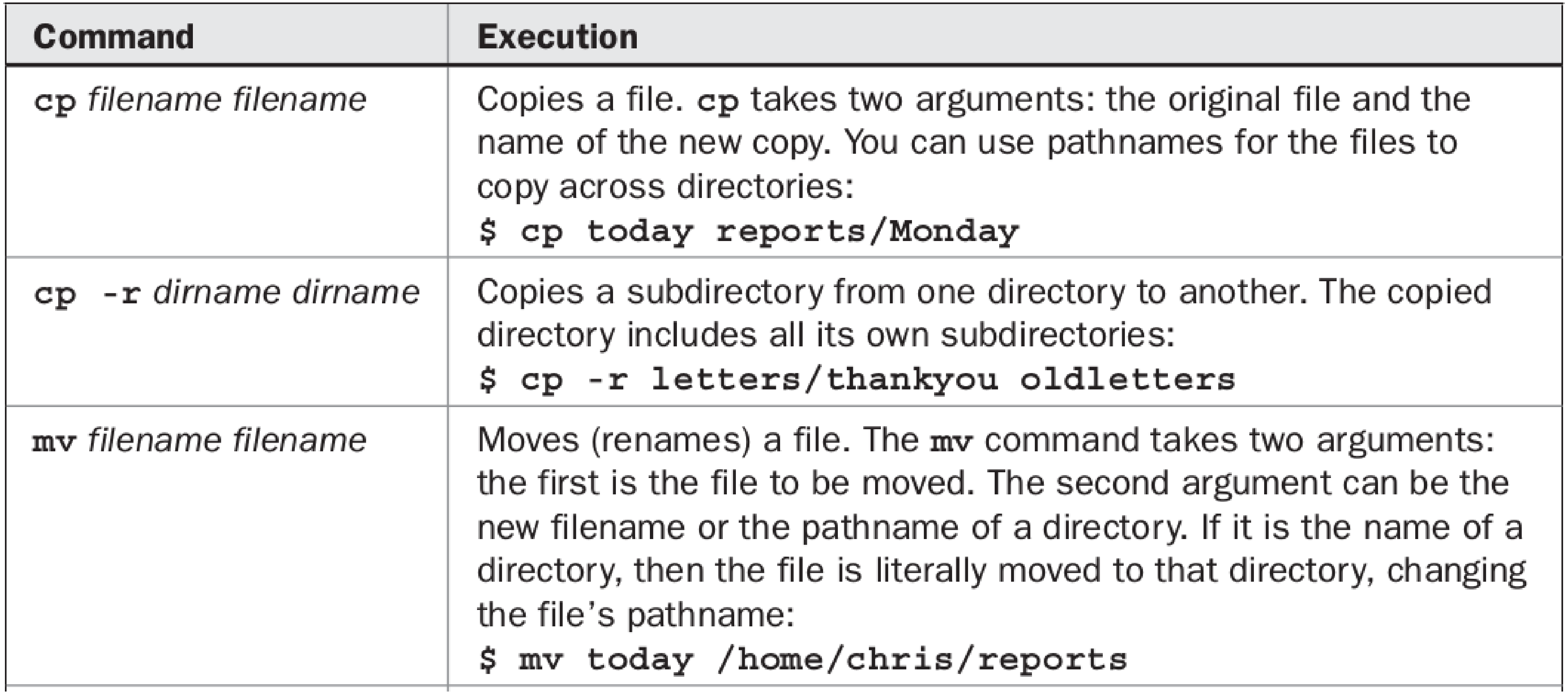
Searching directory: find

- Once you have a large number of files in many different directories, you may need to search them to locate a specific file, or files, of a certain type.
- The find command enables you to perform such a search from the command line.



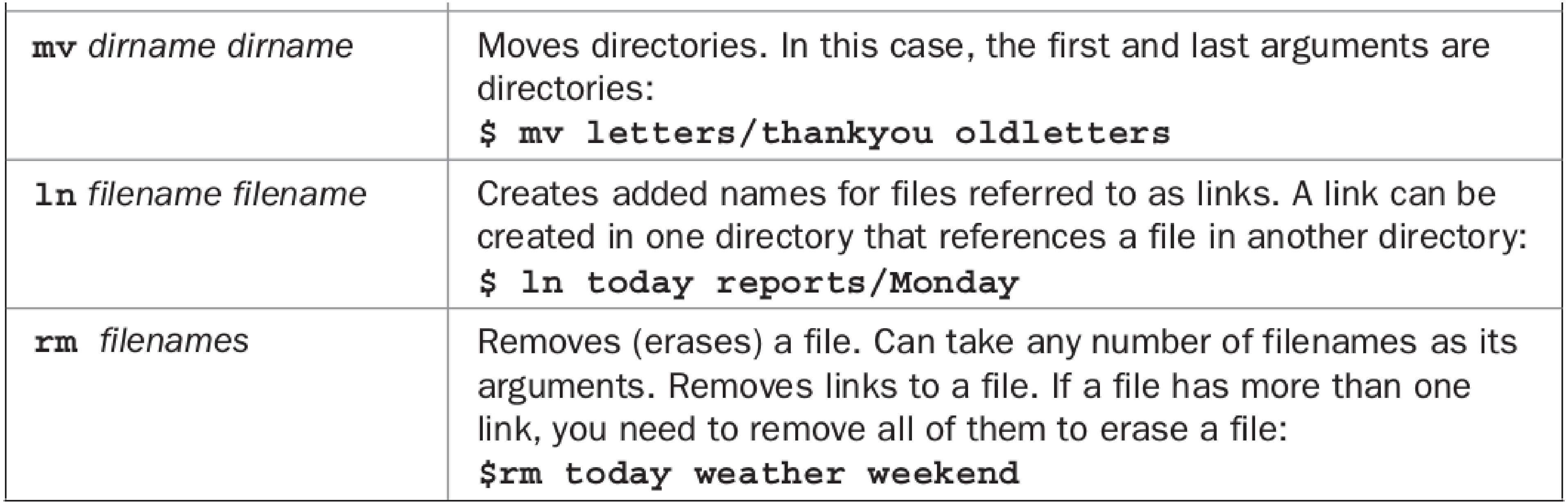
```
rashmi@rashmi-dell:~$ find Documents -name News -print
Documents/Company/News
rashmi@rashmi-dell:~$ sudo find /home/rashmi -name News -print
/home/rashmi/Documents/Company/News
```

Copy and move



Command	Execution
`cp` *filename filename*	Copies a file. `cp` takes two arguments: the original file and the name of the new copy. You can use pathnames for the files to copy across directories: `$ cp today reports/Monday`
`cp -r` *dirname dirname*	Copies a subdirectory from one directory to another. The copied directory includes all its own subdirectories: `$ cp -r letters/thankyou oldletters`
`mv` *filename filename*	Moves (renames) a file. The `mv` command takes two arguments: the first is the file to be moved. The second argument can be the new filename or the pathname of a directory. If it is the name of a directory, then the file is literally moved to that directory, changing the file's pathname: `$ mv today /home/chris/reports`

Copy and move



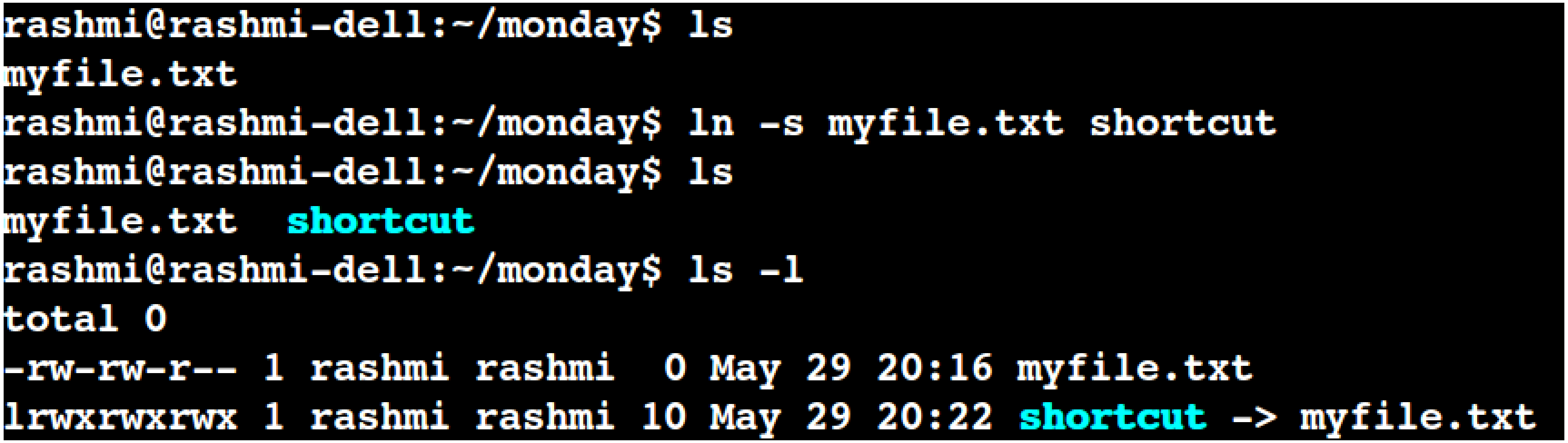
mv *dirname dirname*	Moves directories. In this case, the first and last arguments are directories: **$ mv letters/thankyou oldletters**
ln *filename filename*	Creates added names for files referred to as links. A link can be created in one directory that references a file in another directory: **$ ln today reports/Monday**
rm *filenames*	Removes (erases) a file. Can take any number of filenames as its arguments. Removes links to a file. If a file has more than one link, you need to remove all of them to erase a file: **$rm today weather weekend**

Symbolic Link

- You might want to reference a file using different filenames to access it from different directories. The added names are often referred to as links.
- Linux supports two different types of links, hard and symbolic.
- Hard links are literally another name for the same file, whereas symbolic links function like shortcuts referencing another file.
- Symbolic links are much more flexible and can work over many different file systems, whereas hard links are limited to your local file system.

Symbolic Link

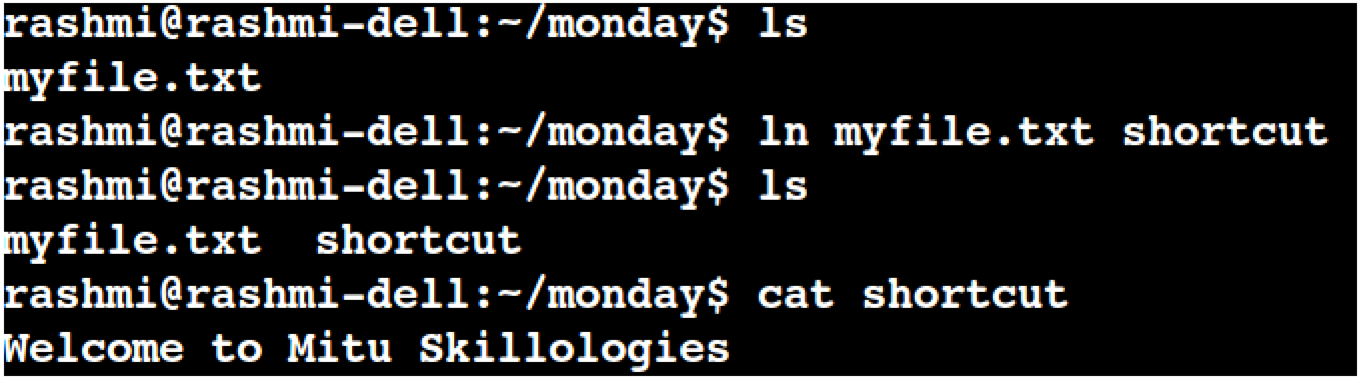
- To set up a symbolic link, you use the ln command with the -s option and two arguments: the name of the original file and the new, added filename.
- The ls operation lists both filenames, but only one physical file will exist.
- **`$ ln -s original-file-name added-file-name`**



```
rashmi@rashmi-dell:~/monday$ ls
myfile.txt
rashmi@rashmi-dell:~/monday$ ln -s myfile.txt shortcut
rashmi@rashmi-dell:~/monday$ ls
myfile.txt  shortcut
rashmi@rashmi-dell:~/monday$ ls -l
total 0
-rw-rw-r-- 1 rashmi rashmi  0 May 29 20:16 myfile.txt
lrwxrwxrwx 1 rashmi rashmi 10 May 29 20:22 shortcut -> myfile.txt
```

Hard Link

- You can give the same file several names by using the ln command on the same file many times.
- To set up a hard link, you use the ln command with no -s option and two arguments: the name of the original file and the new, added filename.
- The ls operation lists both filenames, but only one physical file will exist
- **`$ ln original-file-name added-file-name`**



```
rashmi@rashmi-dell:~/monday$ ls
myfile.txt
rashmi@rashmi-dell:~/monday$ ln myfile.txt shortcut
rashmi@rashmi-dell:~/monday$ ls
myfile.txt  shortcut
rashmi@rashmi-dell:~/monday$ cat shortcut
Welcome to Mitu Skillologies
```

Thank you

Web Resources
http://mitu.co.in
http://tusharkute.com

Blogs
http://digitallocha.blogspot.in
http://kyamputar.blogspot.in

tushar@tusharkute.com