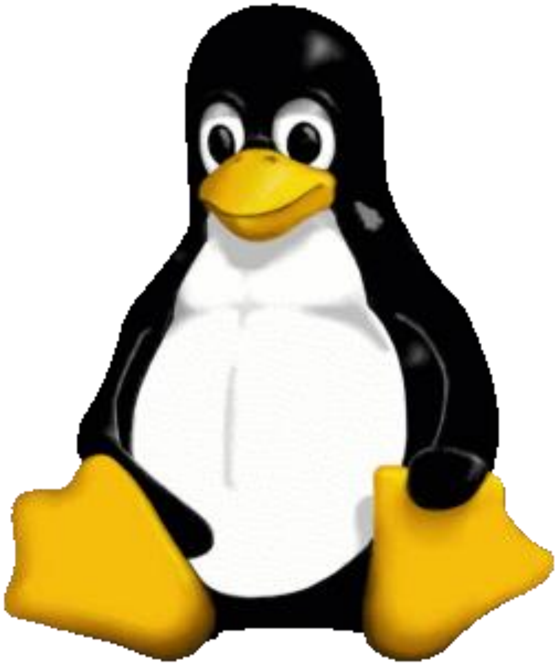
GNU/Linux

Lesson 2 / 3

By Dr. Amir



Today's technical words

- Directory
- Path

man

- Type man followed by a command (for which you want help) and start reading. Press q to quit the manpage. Some man pages contain examples (near the end).

```
amir@laika:~$ man whois
Reformatting whois(1), please wait...
```

□man -k (apropos)

man -k (or apropos) shows a list of man pages containing a string.

```
amir@laika:~$ man -k syslog
lm-syslog-setup (8)          - configure laptop mode to switch syslog.conf …
Logger (1)                             - a shell command interface to the syslog(3) ...
syslog-facility (8)          - Setup and remove LOCALx facility for sysklogd
syslog.conf (5)              - syslogd(8) configuration file
Syslogd (8)                            - Linux system logging utilities.
Syslogd-listfiles (8)        - list system logfiles
```

1 - Executable programs or shell commands
2 - System calls (functions provided by the kernel)
3 - Library calls (functions within program libraries)
4 - Special files (usually found in /dev)
5 - File formats and conventions eg /etc/passwd
6 - Games
7 - Miscellaneous (including macro packages and conventions), e.g. man(7)
8 - System administration commands (usually only for root)
9- Kernel routines [Non standard]

whatis

To see just the description of a manual page, use whatis followed by a string.

```
amir@u810:~$ whatis route
Route (8)                    - show / manipulate the IP routing table
```

whereis

The location of a manpage can be revealed with whereis.

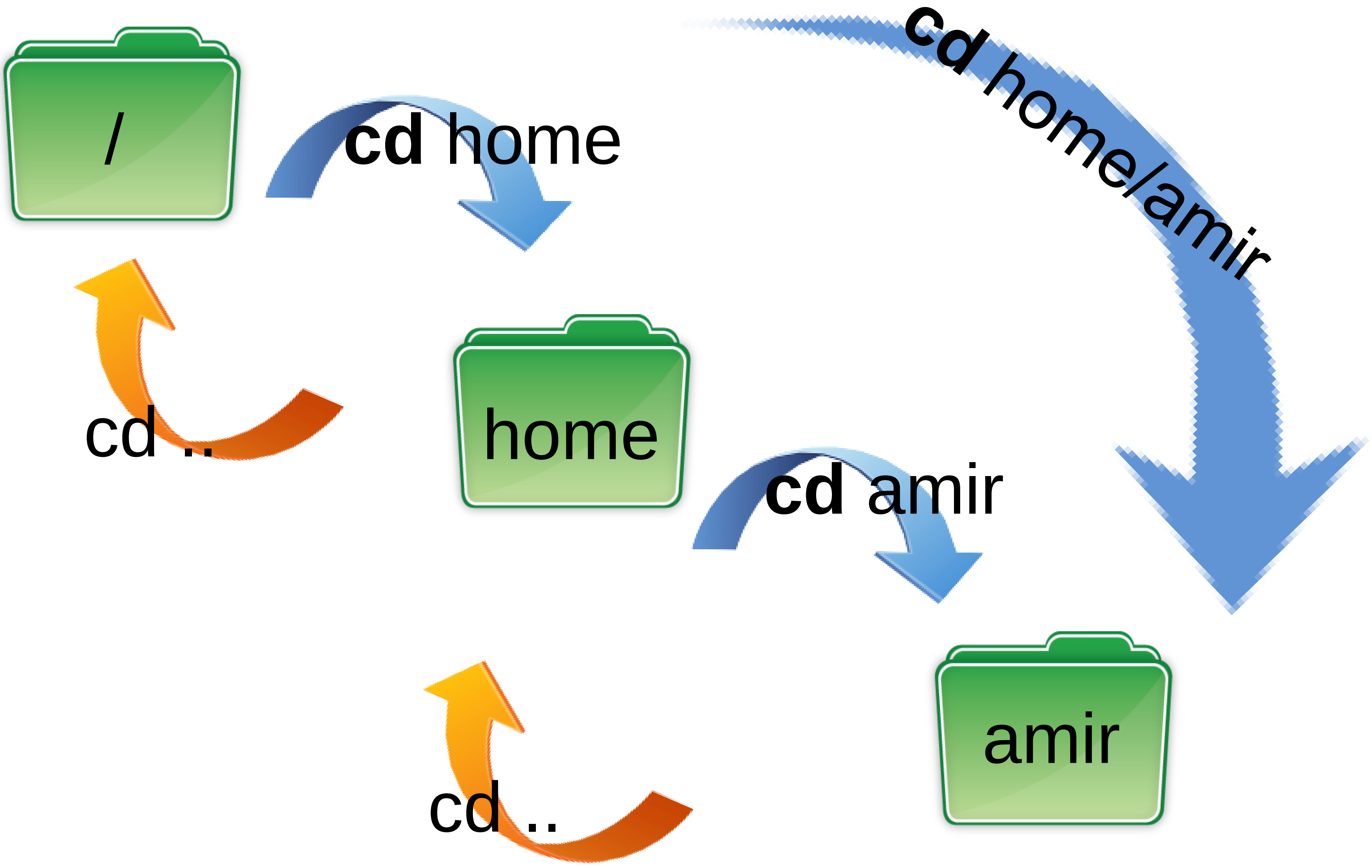
```
amir@laika:~$ whereis -m whois
whois:    /usr/share/man/man1/whois.1.gz
```

This file is directly readable by man.

```
amir@laika:~$ man /usr/share/man/man1/whois.1.gz
```

Moving between directories

/home/amir



Working with directories

Pwd (Print Working Directory)

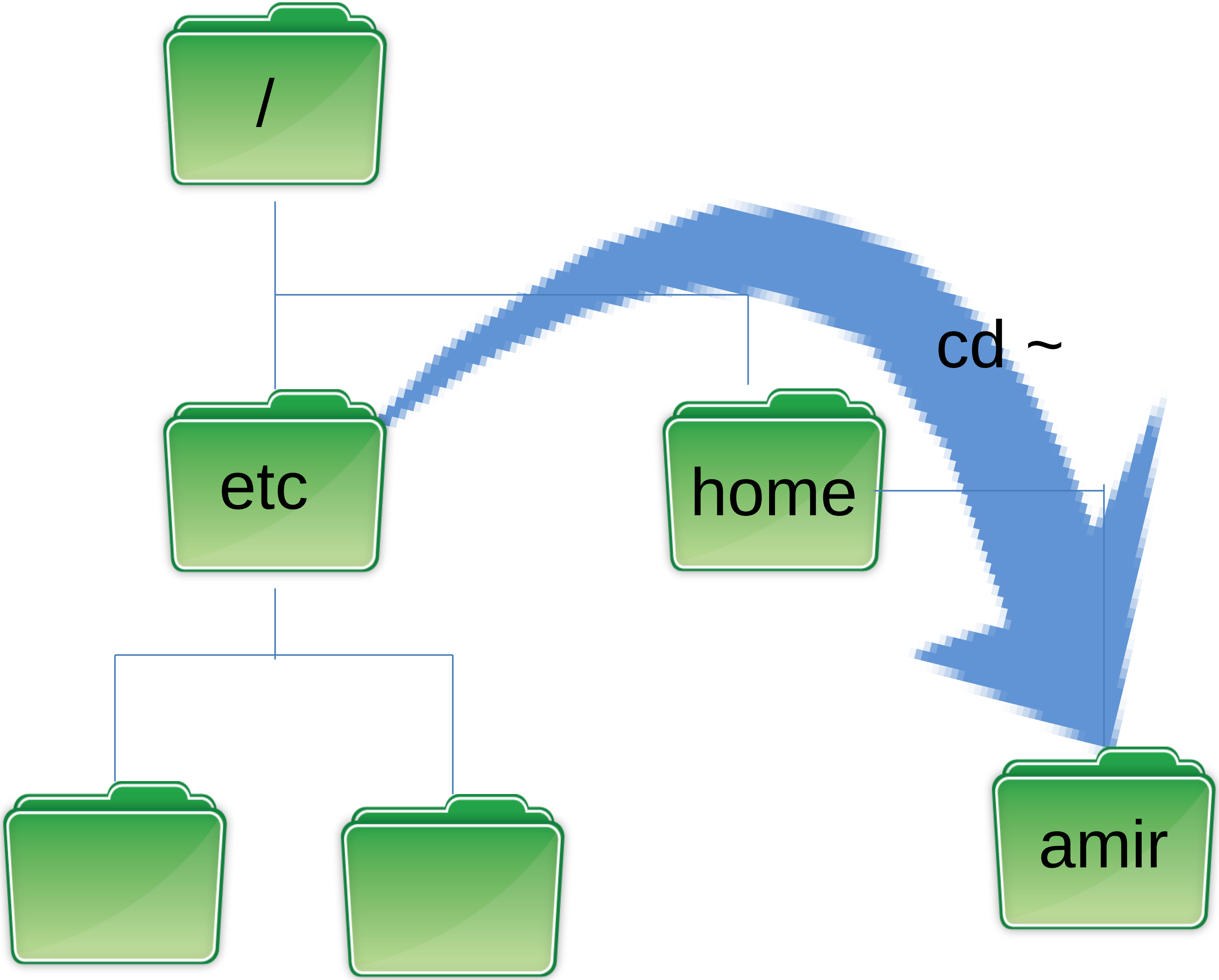
The **you are here** sign can be displayed with the pwd command (Print Working Directory). Go ahead, try it: Open a command line interface (also called a terminal, console or xterm) and type **pwd**. The tool displays your current directory.

```
amir@debian8:~$ pwd
/home/amir
```

cd (change directory)

You can change your current directory with the cd command (Change Directory).

```
amir@debian8$ cd /etc
amir@debian8$ pwd
/etc
amir@debian8$ cd /bin
amir@debian8$ pwd
/bin
amir@debian8$ cd /home/amir/
amir@debian8$ pwd
/home/amir
```


/
etc
home
cd ~
amir

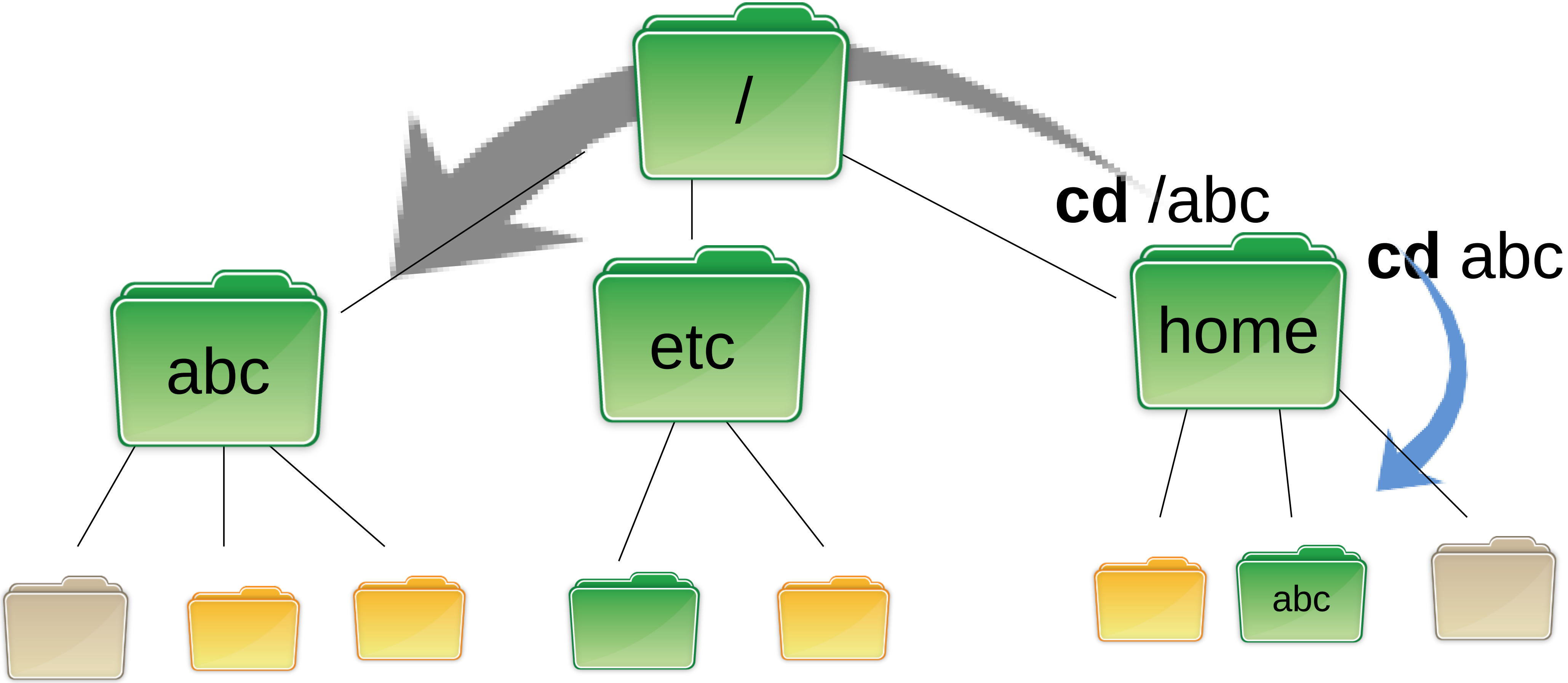
cd -

Another useful shortcut with cd is to just type cd - to go to the previous directory.

```
amir@debian8$ pwd
/home/amir
amir@debian8$ cd /etc
amir@debian8$ pwd
/etc
amir@debian8$ cd -
/home/amir
amir@debian8$ cd -
/etc
```

Absolute and relative path

You should be aware of absolute and relative paths in the file tree. When you type a path starting with a slash (/), then the root of the file tree is assumed. If you don't start your path with a slash, then the current directory is the assumed starting point.



path completion

The **tab key** can help you in typing a **path** without errors. Typing **cd /et** followed by the tab key will expand the command line to **cd /etc/.** When typing cd /Et followed by the **tab key**, nothing will happen because you typed the wrong **path** (upper case E).

ls

You can list the contents of a directory with ls.

```
amir@debian8:~$ ls
allfiles.txt dmesg.txt       services        stuff     summer.txt

amir@debian8:~$
```

ls -a

A frequently used option with ls is -a to show all files. Showing all files means including the hidden files. When a file name on a Linux file system starts with a dot, it is considered a hidden file and it doesn't show up in regular file listings.

```
amir@debian8:~$ ls -a
.          allfiles.txt     .bash_profile   dmesg.txt    .lesshst    stuff

..         .bash_history    .bashrc         services     .ssh        summer.txt

amir@debian8:~$
```

ls -l

Many times you will be using options with ls to display the contents of the directory in different formats or to display different parts of the directory. Typing just ls gives you a list of files in the directory. Typing ls -l (that is a letter L, not the number 1) gives you a long listing.

```
amir@debian8:~$ ls -l
total 17296
-rw-r--r--   1   amir   amir   17584442   Sep   17 00:03   allfiles.txt
-rw-r--r--   1   amir   amir   96650      Sep   17 00:03   dmesg.txt
-rw-r--r--   1   amir   amir   19558      Sep   17 00:04   services
drwxr-xr-x   2   amir   amir   4096       Sep   17 00:04   stuff
-rw-r--r--   1   amir   amir   0          Sep   17 00:04   summer.txt
```

ls -lh

Another frequently used ls option is -h. It shows the numbers (file sizes) in a more human readable format. Also shown below is some variation in the way you can give the options to ls. We will explain the details of the output later in this book.

```
amir@debian8:~$ ls -lh
total 17296
-rw-r--r--    1        amir      amir      17M      Sep       17 00:03                allfiles.txt
-rw-r--r--    1        amir      amir      95K      Sep       17 00:03                dmesg.txt
-rw-r--r--    1        amir      amir      20K      Sep       17 00:04                services
drwxr-xr-x  2        amir      amir      4.0K     Sep       17 00:04                stuff
-rw-r--r--    1        amir      amir      0        Sep       17 00:04                summer.txt
```

ls –lh   ls –hl   ls –l –h   ls –h –l

All of the above forms of using ‘l’ and ‘h’ are the same

mkdir

Walking around the Unix file tree is fun, but it is even more fun to create your own directories with **mkdir**. You have to give at least one parameter to **mkdir**, the name of the new directory to be created. Think before you type a leading / .

```
amir@debian8:~$ mkdir mydir
amir@debian8:~$ cd mydir
amir@debian8:~/mydir$ ls –al
total 8
drwxr-xr-x 2 amir amir 4096 Sep 17 00:07 .
drwxr-xr-x 48 amir amir 4096 Sep 17 00:07 ..
```

The following command will fail, because the parent directory of threedirsdeep does not exist.

```
amir@debian8:~$ mkdir mydir2/mysubdir2/threedirsdeep
mkdir: cannot create directory ‘mydir2/mysubdir2/threedirsdeep’: No such file or\
directory
```

mkdir -p

When given the option -p, then mkdir will create parent directories as needed.

```
amir@debian8:~$ mkdir -p mydir2/mysubdir2/threedirsdeep
amir@debian8:~$ cd mydir2
amir@debian8:~/mydir2$ ls -l
total 4
drwxr-xr-x 3 amir amir 4096 Sep 17 00:11 mysubdir2
amir@debian8:~/mydir2$ cd mysubdir2
amir@debian8:~/mydir2/mysubdir2$ ls -l
total 4
drwxr-xr-x 2 amir amir 4096 Sep 17 00:11 threedirsdeep
amir@debian8:~/mydir2/mysubdir2$ cd threedirsdeep/
amir@debian8:~/mydir2/mysubdir2/threedirsdeep$ pwd
/home/amir/mydir2/mysubdir2/threedirsdeep
```

rmdir

When a directory is empty, you can use **rmdir** to remove the directory.

```
paul@debian8:~/mydir$ ls -l
total 8
drwxr-xr-x 2 paul paul 4096 Sep 17 00:08 otherstuff
drwxr-xr-x 2 paul paul 4096 Sep 17 00:08 stuff
paul@debian8:~/mydir$ rmdir otherstuff
paul@debian8:~/mydir$ cd ..
paul@debian8:~$ rmdir mydir
rmdir: failed to remove ‘mydir’: Directory not empty
paul@debian8:~$ rmdir mydir/stuff
paul@debian8:~$ rmdir mydir
paul@debian8:~$
```

rmdir -p

And similar to the **mkdir -p** option, you can also use **rmdir** to recursively remove directories.

```
paul@debian8:~$ mkdir -p test42/subdir
paul@debian8:~$ rmdir -p test42/subdir
paul@debian8:~$
```