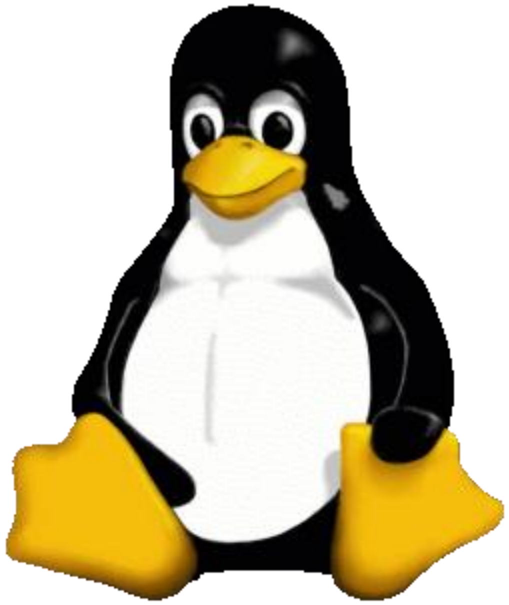
Lesson 6

By Dr. Amir

GNU/Linux

Linux file system hierarchy



Linux file system hierarchy

‘/’ root

/bin

The /bin directory contains binaries for use by all users.

/sbin

/sbin contains binaries to configure the operating system. Many of the system binaries require root privilege to perform certain tasks.

/lib

Binaries found in /bin and /sbin often use shared libraries located in /lib. Below is a screenshot of the partial contents of /lib.

/lib/modules

Typically, the Linux kernel loads kernel modules from /lib/modules/$kernel-version/.

This directory is discussed in detail in the Linux kernel chapter.

/opt

The purpose of /opt is to store optional software. In many cases this is software from outside

the distribution repository. You may find an empty /opt directory on many systems.

/boot

The /boot directory contains all files needed to boot the computer. These files don't change very often.

/etc

All of the machine-specific configuration files should be located in /etc. Historically /etc stood for etcetera, today people often use the Editable Text Configuration backronym.

/home

Users can store personal or project data under /home. It is common (but not mandatory by the fhs) practice to name the users home directory after the user name in the format /home/$USERNAME.

/root

On many systems /root is the default location for personal data and profile of the root user.

If it does not exist by default, then some administrators create it.

/srv

You may use /srv for data that is served by your system. The FHS allows locating cvs,

rsync, ftp and www data in this location.

/media

The /media directory serves as a mount point for removable media devices such as CD-ROM's, digital cameras, and various usb attached devices. Since /media is rather new in the Unix world, you could very well encounter systems running without this directory.

/mnt

The /mnt directory should be empty and should only be used for temporary mount points

(according to the FHS).

/tmp

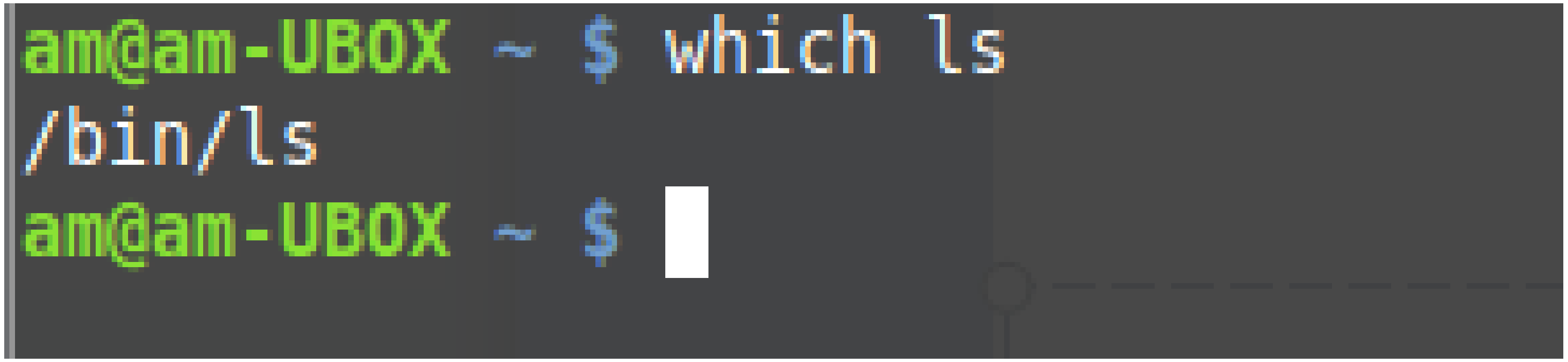
Applications and users should use /tmp to store temporary data when needed. Data stored in /tmp may use either disk space or RAM.

man hier

For more information about filing system of your Linux operating system

which *<command>*

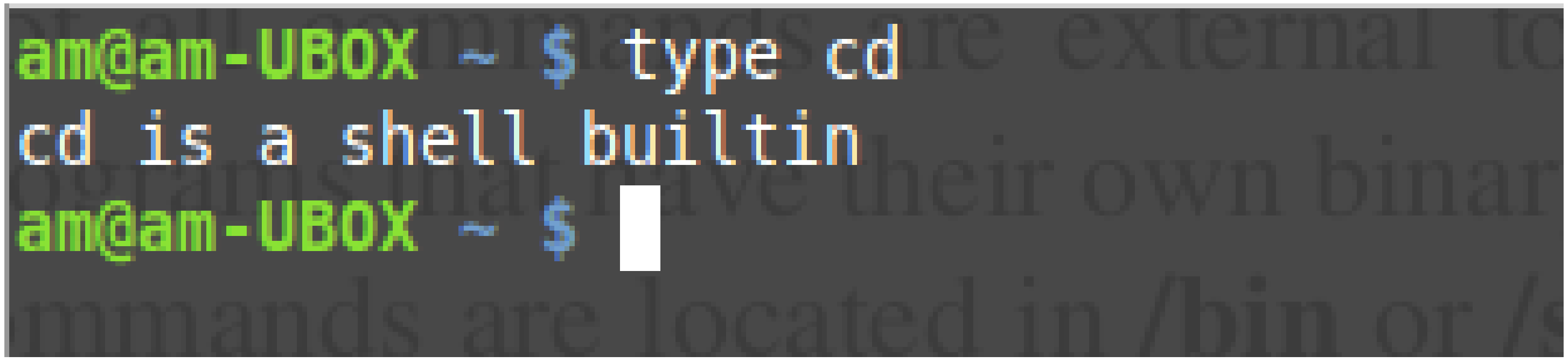
Returns the address for a command



```
am@am-UBOX ~ $ which ls
/bin/ls
am@am-UBOX ~ $ 
```

type *<command>*

Will tell you that a command is external or internal



```
am@am-UBOX ~ $ type cd
cd is a shell builtin
am@am-UBOX ~ $ 
```

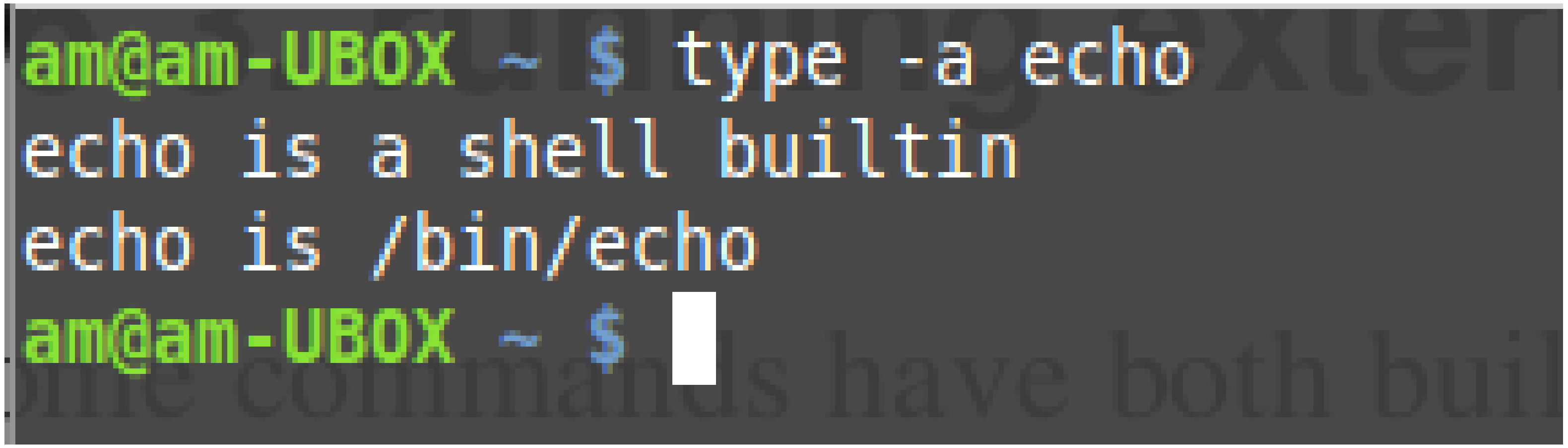
External and Internal commands

Some commands are extterrnal and some are part of the Linux shell.

If such a command is called, the internal one will be executed. To call external, you must use full address.

type –a *<command>*

To see both external command and the shell built-in commands:

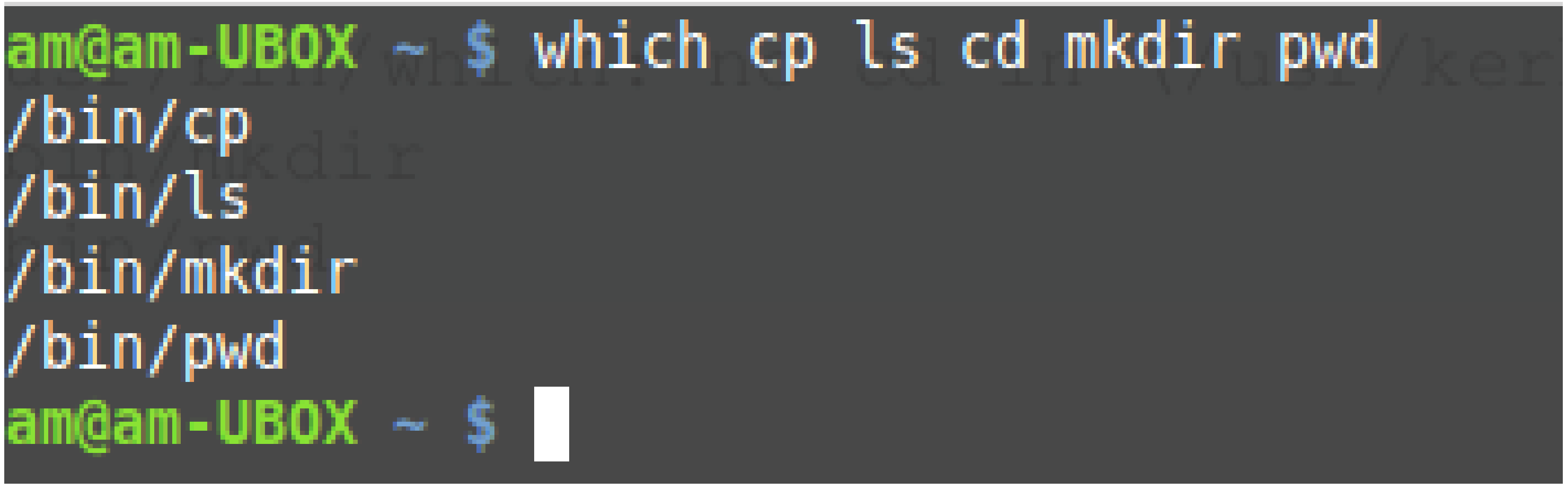


```
am@am-UBOX ~ $ type -a echo
echo is a shell builtin
echo is /bin/echo
am@am-UBOX ~ $ 
```

which *<command>*

Which will return only external commands path.

In this example cd is a shell built-in command.



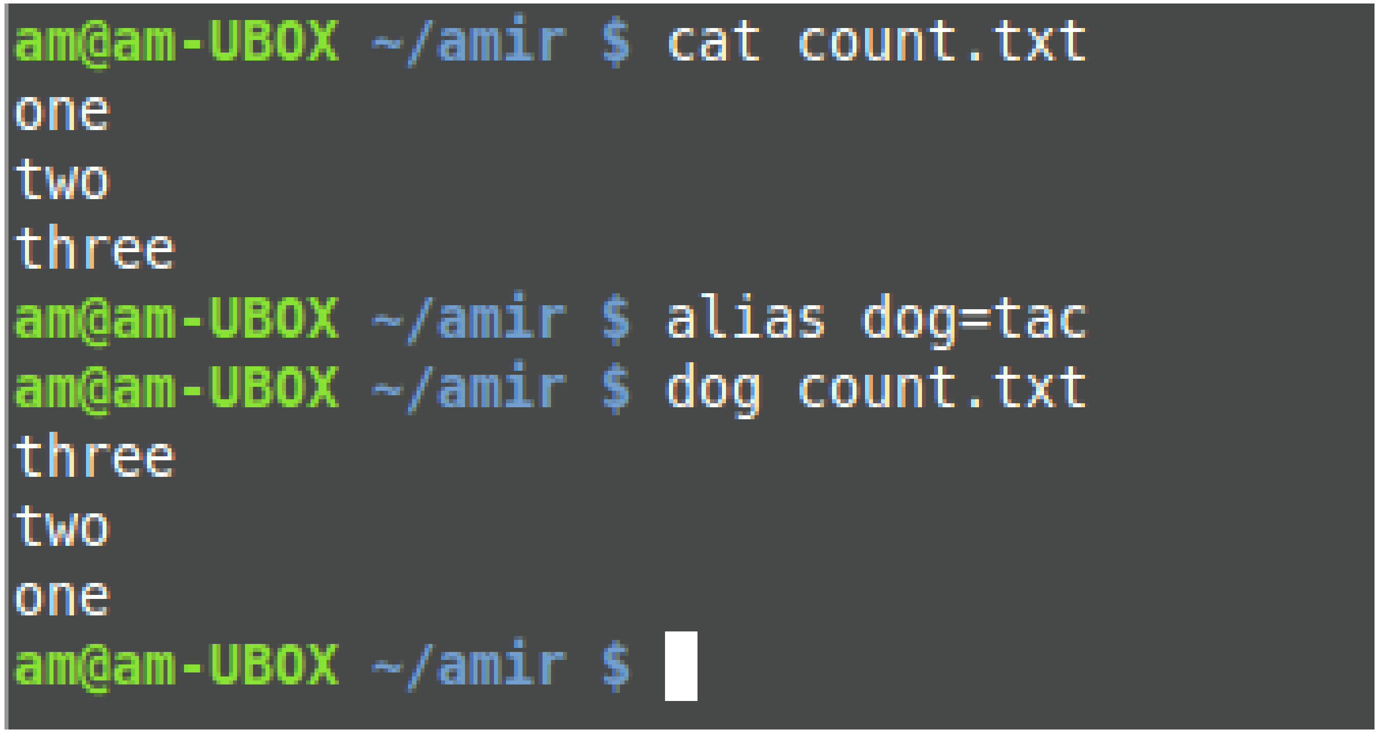
```
am@am-UBOX ~ $ which cp ls cd mkdir pwd
/bin/cp
/bin/ls
/bin/mkdir
/bin/pwd
am@am-UBOX ~ $
```

man hier

Returns the file hierarchy of your Linux system

Alias name= *<command>*

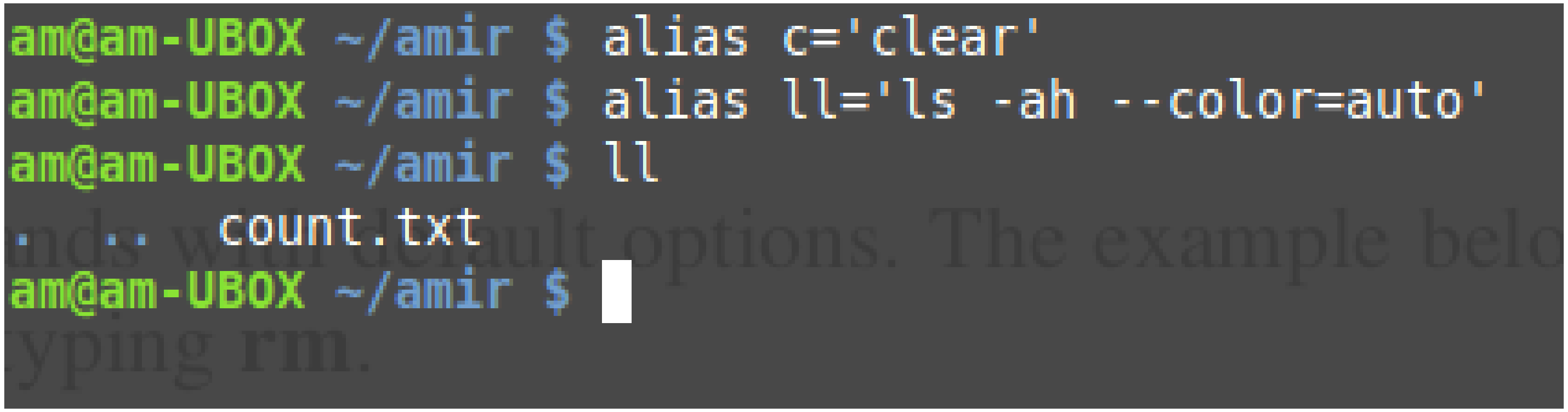
The shell allows you to create aliases.
Aliases are often used to create an easier to remember name for an existing command or to easily supply parameters.



```
am@am-UBOX ~/amir $ cat count.txt
one
two
three
am@am-UBOX ~/amir $ alias dog=tac
am@am-UBOX ~/amir $ dog count.txt
three
two
one
am@am-UBOX ~/amir $ 
```

Alias

An alias can also be useful to abbreviate an existing command.

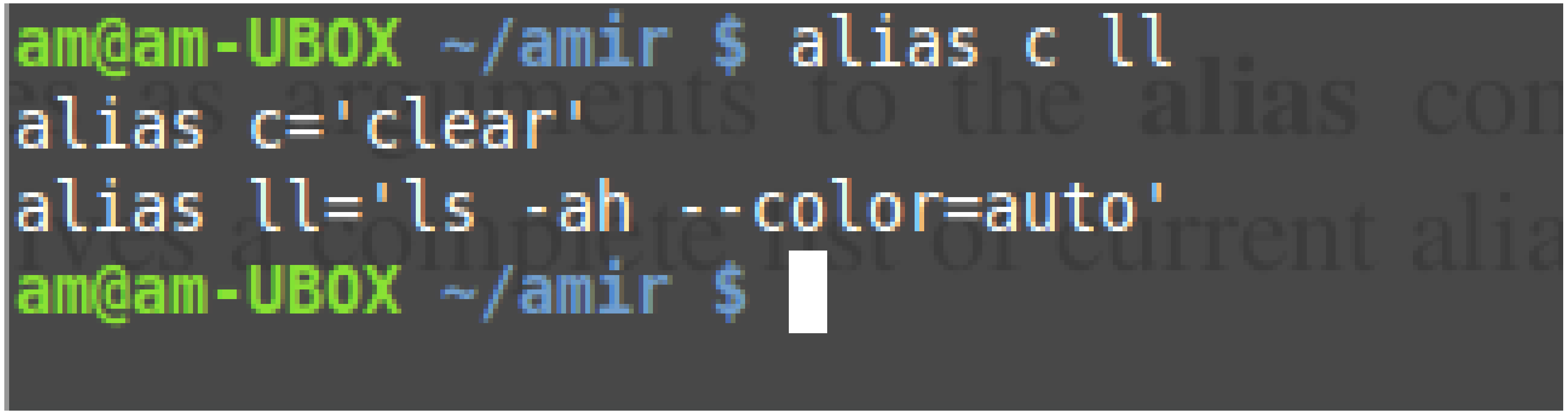


```
am@am-UBOX ~/amir $ alias c='clear'
am@am-UBOX ~/amir $ alias ll='ls -ah --color=auto'
am@am-UBOX ~/amir $ ll
.  ..  count.txt
am@am-UBOX ~/amir $ 
```

Linux command

View aliases

To view aliases we just call them

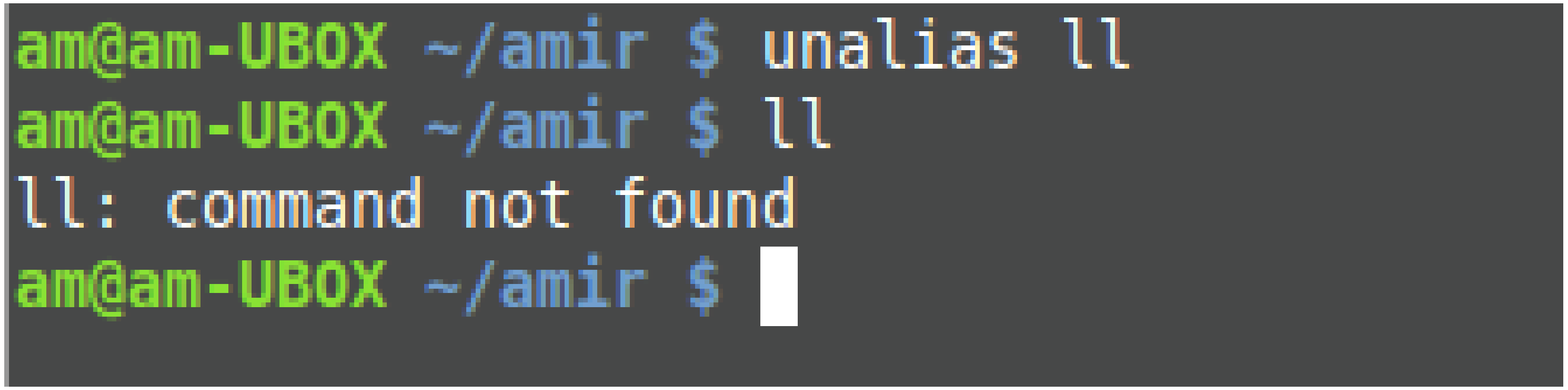


```
am@am-UBOX ~/amir $ alias c ll
alias c='clear'
alias ll='ls -ah --color=auto'
am@am-UBOX ~/amir $
```

Linux command

unalias *<command>*

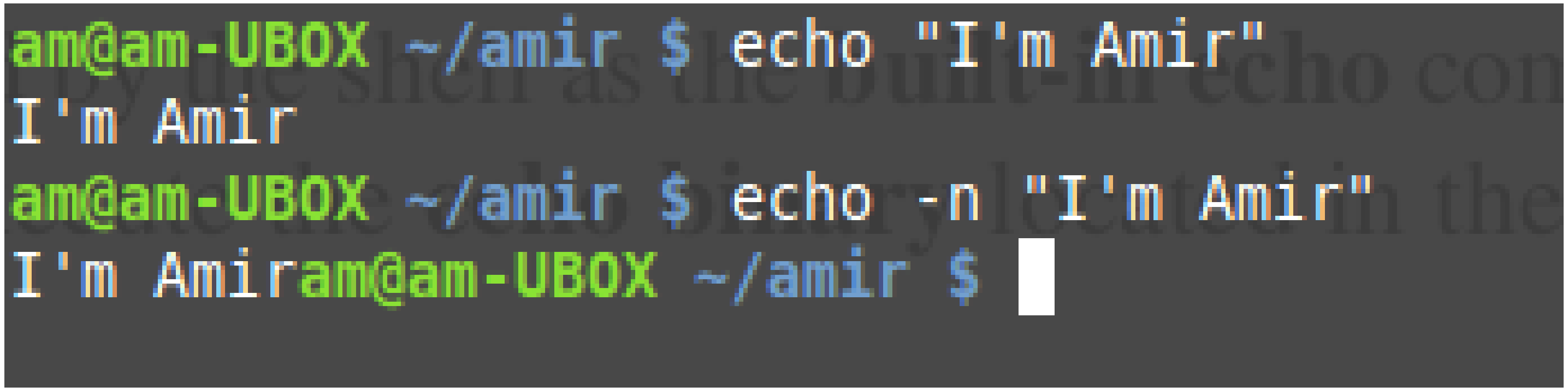
You can undo an alias with the unalias command.



```
am@am-UBOX ~/amir $ unalias ll
am@am-UBOX ~/amir $ ll
ll: command not found
am@am-UBOX ~/amir $
```

Echo -n ‘text’

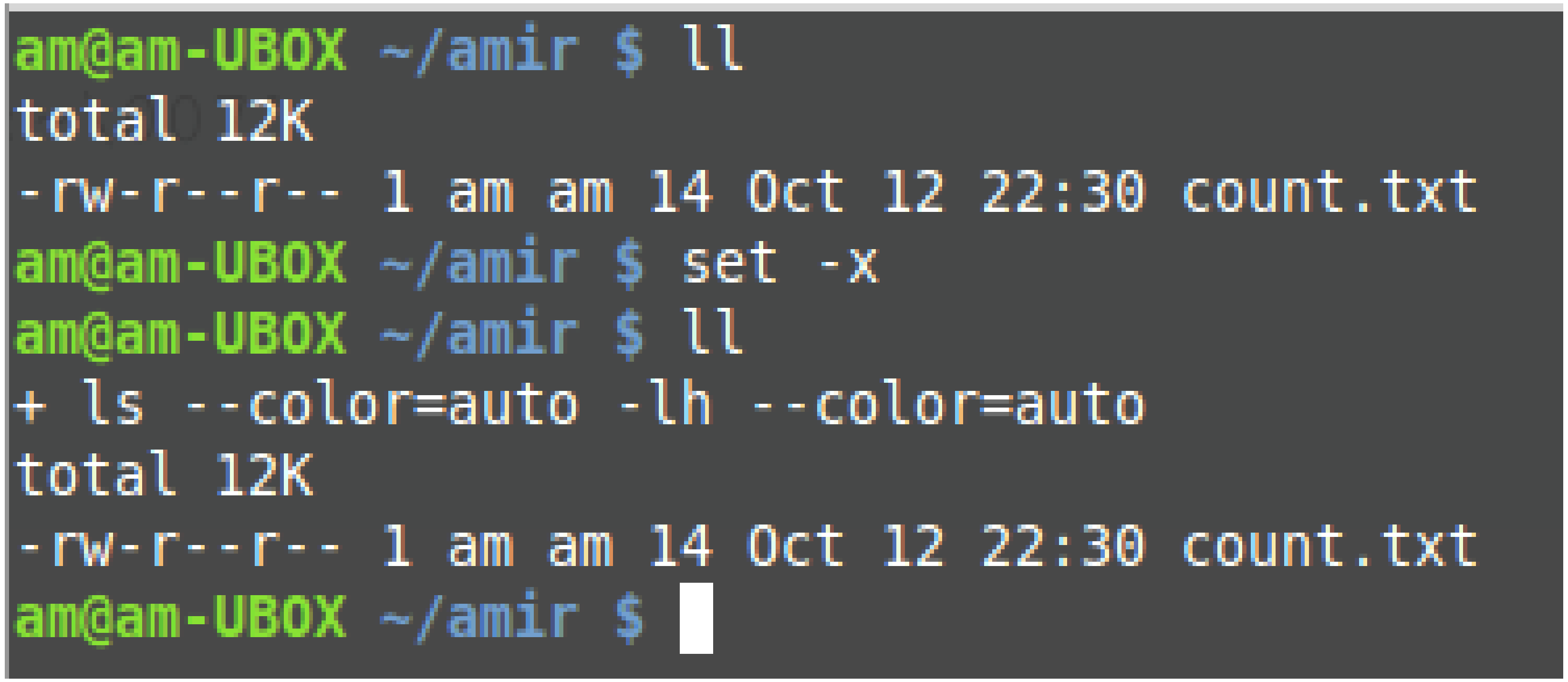
-n option prevent echo of inserting a new line.



```
am@am-UBOX ~/amir $ echo "I'm Amir"
I'm Amir
am@am-UBOX ~/amir $ echo -n "I'm Amir"
I'm Amiram@am-UBOX ~/amir $ 
```

Set -x

To display shell expansion.

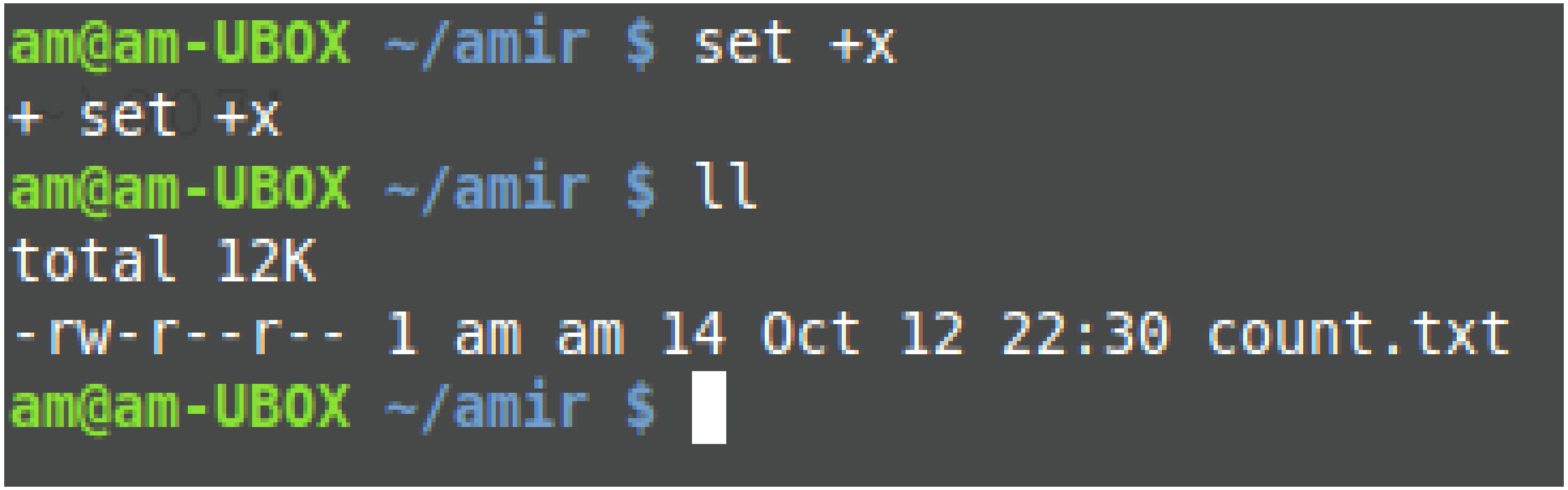


```
am@am-UBOX ~/amir $ ll
total 12K
-rw-r--r-- 1 am am 14 Oct 12 22:30 count.txt
am@am-UBOX ~/amir $ set -x
am@am-UBOX ~/amir $ ll
+ ls --color=auto -lh --color=auto
total 12K
-rw-r--r-- 1 am am 14 Oct 12 22:30 count.txt
am@am-UBOX ~/amir $ 
```

Linux command

Set +x

To disable (turn off) shell expansion.



```
am@am-UBOX ~/amir $ set +x
+ set +x
am@am-UBOX ~/amir $ ll
total 12K
-rw-r--r-- 1 am am 14 Oct 12 22:30 count.txt
am@am-UBOX ~/amir $ 
```

Exercises

1. How many arguments are in this line (not counting the command itself).

```
touch '/etc/cron/cron.allow' 'file 42.txt' "file 33.txt"
```

three

Exercises

2. Is ‘tac’ a shell command?

```
type tac
```

Exercises

3. Is there an existing alias for rm ?

```
alias rm
```

Exercises

4. Read the man page of rm, make sure you understand the -i option of rm. Create and remove a file to test the -i option.

```
man rm
touch testfile
rm -i testfile
```

Exercises

5. Execute: alias rm='rm -i' . Test your alias with a test file. Does this work as expected ?

touch testfile
rm testfile (should ask for confirmation)

Exercises

6. List all current aliases.

```
alias
```

7a. Create an alias called 'city' that echoes your hometown.

```
alias city=‘echo Manchester’
```

Exercises

7b. Use your alias to test that it works.

city

Exercises

9- Test the functionality of set -x by executing your city and rm aliases.

```
set –x
city
```

Exercises

10 Execute set +x to stop displaying shell expansion.

Set +X

Exercises

11. Remove your city alias.

```
unalias city
```

Exercises

12. What is the location of the cat and the passwd commands ?

```
which cat

which passwd
```

13. Explain the difference between the following commands:

echo

/bin/echo

Echo is a shell built-in command

/bin/echo is an external command

14. Explain the difference between the following commands:

```
echo Hello
echo -n Hello
```

The -n option of the echo command will prevent echo from echoing a trailing newline.
Echo Hello will echo six characters in total, **echo -n** hello only echoes five characters.

15. Display A B C with two spaces between B and C.

```
echo "A B  C"
```

16- Complete the following command (do not use spaces) to display exactly the following output:

```
4+4         =8
10+14       =24
```

```
echo -e "4+4\t=8" ; echo -e "10+14\t=24"
```

17. Use one echo command to display three words on three lines.

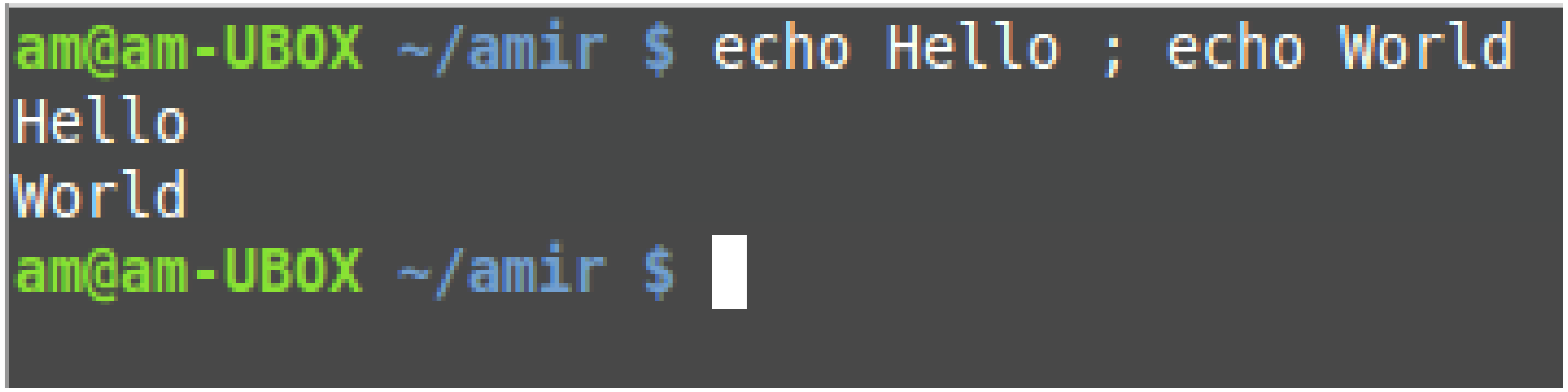
```
echo -e "one \ntwo \nthree"
```

Control Operators

Linux command

;

To enter more than one command on a line

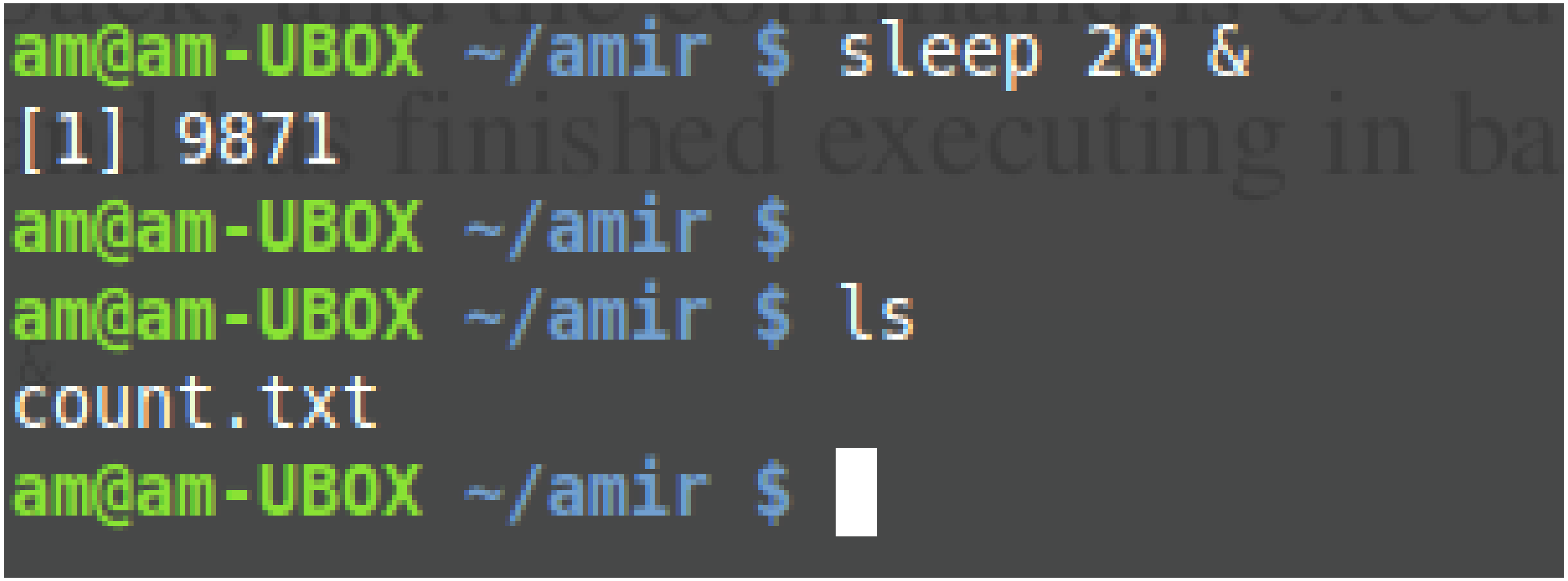


```
am@am-UBOX ~/amir $ echo Hello ; echo World
Hello
World
am@am-UBOX ~/amir $ 
```

&

& (ampersand)

When a line ends with an ampersand &, the shell will not wait for the command to finish.



```
am@am-UBOX ~/amir $ sleep 20 &
[1] 9871
am@am-UBOX ~/amir $
am@am-UBOX ~/amir $ ls
count.txt
am@am-UBOX ~/amir $
```

type *<command>*

Linux command

type *<command>*