

Purdue University - ITaP

Gladys Andino

Dan Dietz

Jieyu Gao

Lev Gorenstein

Erik Gough

Stephen Harrell

Randy Herban

Steve Kelley

Boyu Zhang

Xiao Zhu

rcac-help@purdue.edu

January 23 and 25, 2018

Slides available:

www.rcac.purdue.edu/training/unix101/

Acknowledgments

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

- The material in this workshop was prepared by the Purdue University ITaP Research Computing team.
- Special thanks to Eric Adams and Megan Dale for organizing the workshop sessions.
- We have drawn from documentation provided by the Purdue Bioinformatics Core used in the UNIX for Biologists workshop and Next-generation Transcriptome Analysis Workshop Manual provided by Professor Michael Gribskov and Professor Esperanza Torres.

Acknowledgments

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

- Please sign the attendance sheet
- We will have a couple short (10 min) breaks throughout the workshop
- Please fill out the evaluations we will send next week.
Your feedback allows us to improve this workshop!

Acknowledgments

Acknowledgments

Logging In

Files and Directories

First Command

Basic Commands

File Permissions

Compressing and Archiving

Redirects, Pipelines and Scripts

Conventions used in the workshop slides:

- Any text shown in a fixed width font refers to a command or file name
- Any text prefaced by a \$ is a command to be typed. Don't type the \$!
- Blocks of text in a box are output from the terminal:

```
# Our comments are in blue  
# Commands are again prefaced by a $, output follows.  
$ command  
This is the output of command!
```

Logging In

Acknowledgments

Logging In

Windows

Mac

Linux

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Logging In

■ Windows

■ Mac

■ Linux

Logging In

Acknowledgments

Logging In

Windows
Mac
Linux

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

- We will be using the Radon cluster
www.rcac.purdue.edu/compute/radon/
- Everyone has been given an account on the cluster for the duration of the workshop

Logging In

Acknowledgments

Logging In

Windows
Mac
Linux

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Logging into a remote UNIX based system requires a client based on the "SSH" or Secure Shell protocol.

- Encrypted
- Used on most UNIX systems
- Variety of clients for all platforms

Logging In

Windows

Acknowledgments

Logging In

Windows

Mac

Linux

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Many clients are available for Windows:

- We will use the PuTTY SSH client
- Download PuTTY, no install required
- www.chiark.greenend.org.uk/~sgtatham/putty/download.html
(or Google search *putty*)
- Download `putty.exe` for Intel x86 to your desktop

Logging In

Windows

Acknowledgments

Logging In

Windows

Mac

Linux

Files and
Directories

First
Command

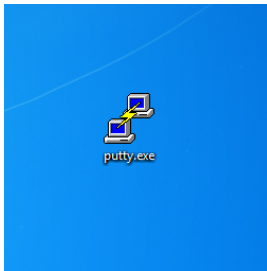
Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Go to desktop and double click downloaded file



Logging In

Windows

Acknowledgments

Logging In

Windows

Mac

Linux

Files and
Directories

First
Command

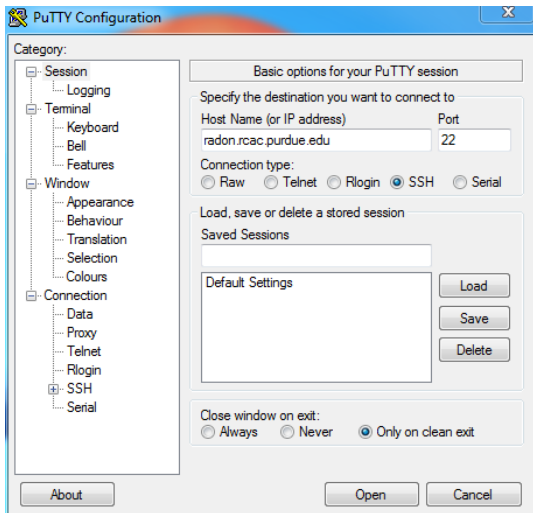
Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Host Name for Radon is `radon.rcac.purdue.edu`



Logging In

Windows

Acknowledgments

Logging In

Windows

Mac

Linux

Files and
Directories

First
Command

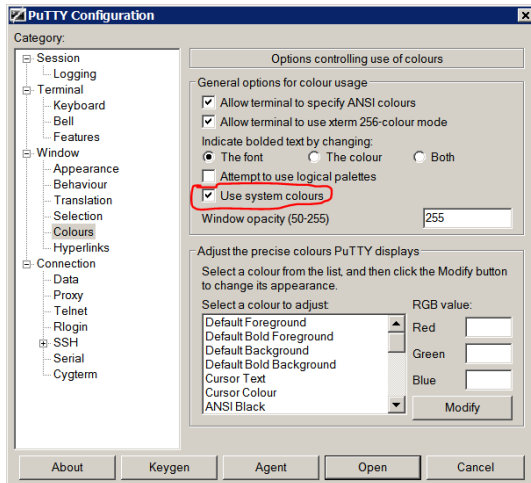
Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

One tweak: enable system colors in Appearance → Colours



Logging In

Windows

Acknowledgments

Logging In

Windows

Mac

Linux

Files and
Directories

First
Command

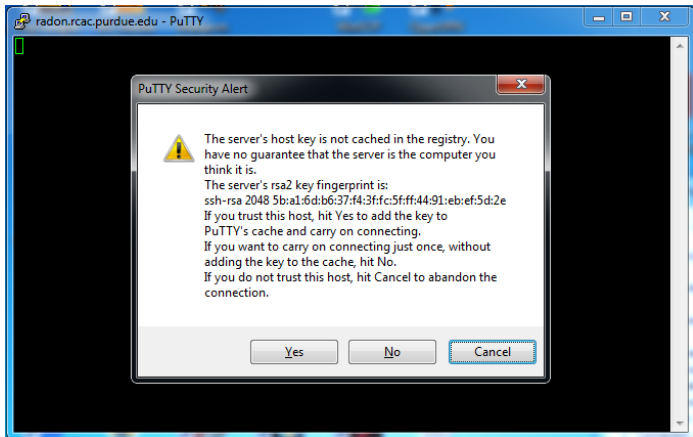
Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Accept server host key



Logging In

Windows

Acknowledgments

Logging In

Windows

Mac

Linux

Files and
Directories

First
Command

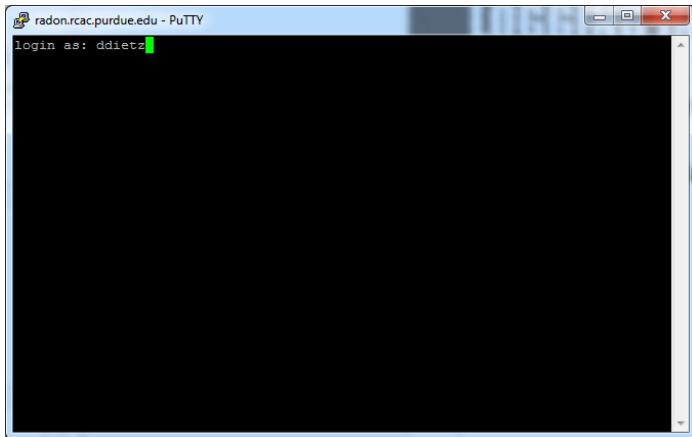
Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

PuTTY will prompt for your Purdue Career Account



Logging In

Windows

Acknowledgments

Logging In

Windows

Mac

Linux

Files and

Directories

First

Command

Basic

Commands

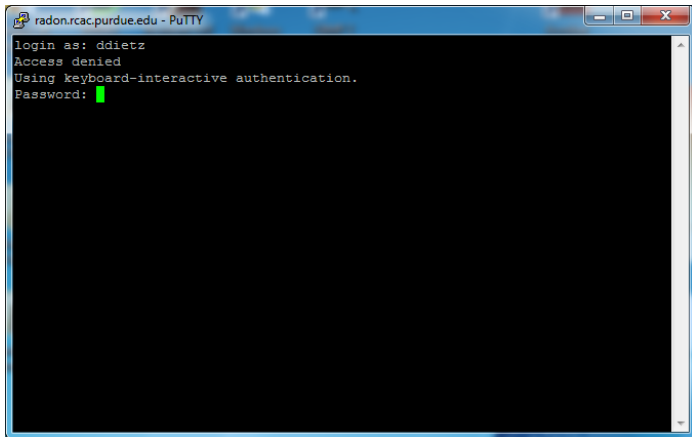
File

Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Use your Career Account password when prompted



Logging In

Windows

Acknowledgments

Logging In

Windows

Mac

Linux

Files and

Directories

First

Command

Basic

Commands

File

Permissions

Compressing

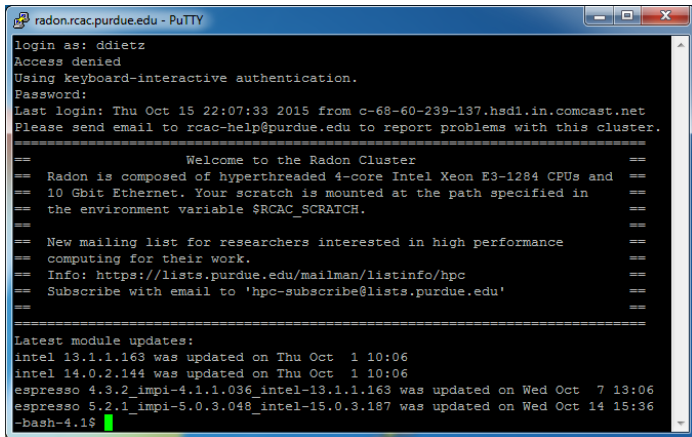
and Archiving

Redirects,

Pipelines and

Scripts

Now you should be logged in!



```
radon.rcac.purdue.edu - PuTTY
login as: ddietz
Access denied
Using keyboard-interactive authentication.
Password:
Last login: Thu Oct 15 22:07:33 2015 from c-68-60-239-137.hsd1.in.comcast.net
Please send email to rcac-help@purdue.edu to report problems with this cluster.

=====
                        Welcome to the Radon Cluster                        ==
==  Radon is composed of hyperthreaded 4-core Intel Xeon E3-1284 CPUs and ==
==  10 Gbit Ethernet. Your scratch is mounted at the path specified in ==
==  the environment variable $RCAC_SCRATCH.                               ==
==                                                                           ==
==  New mailing list for researchers interested in high performance ==
==  computing for their work.                                             ==
==  Info: https://lists.purdue.edu/mailman/listinfo/hpc                  ==
==  Subscribe with email to 'hpc-subscribe@lists.purdue.edu'            ==
==                                                                           ==
=====
Latest module updates:
intel 13.1.1.163 was updated on Thu Oct  1 10:06
intel 14.0.2.144 was updated on Thu Oct  1 10:06
espresso 4.3.2_impi-4.1.1.036_intel-13.1.1.163 was updated on Wed Oct  7 13:06
espresso 5.2.1_impi-5.0.3.048_intel-15.0.3.187 was updated on Wed Oct 14 15:36
-bash-4.1$
```

Logging In

Mac

Acknowledgments

Logging In

Windows

Mac

Linux

Files and Directories

First Command

Basic Commands

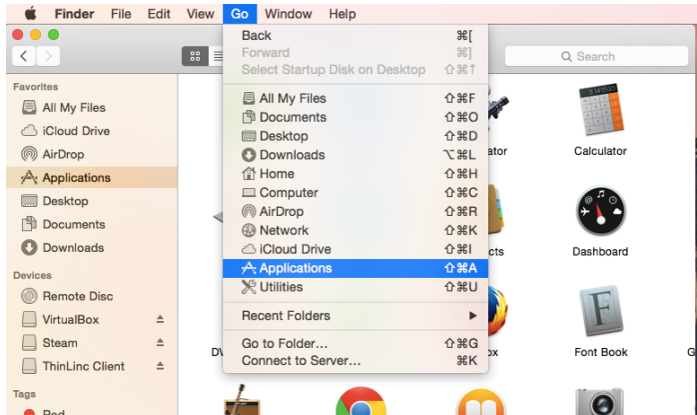
File Permissions

Compressing and Archiving

Redirects, Pipelines and Scripts

Mac OS X has built in Terminal app that can use SSH

Open Finder and Go to Applications



Logging In

Mac

Acknowledgments

Logging In

Windows

Mac

Linux

Files and
Directories

First
Command

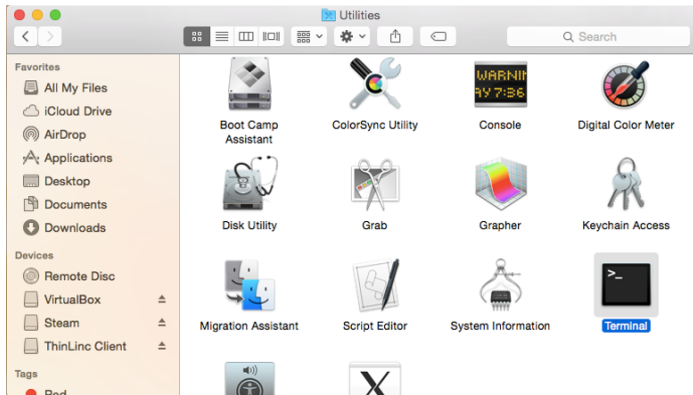
Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Find Utilities folder, open it, and find Terminal app



Logging In

Mac

Acknowledgments

Logging In

Windows

Mac

Linux

Files and

Directories

First

Command

Basic

Commands

File

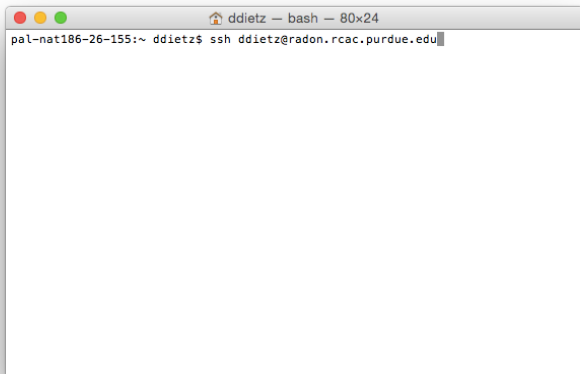
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Connect using:

```
ssh myusername@radon.rcac.purdue.edu
```



Logging In

Mac

Acknowledgments

Logging In

Windows

Mac

Linux

Files and

Directories

First

Command

Basic

Commands

File

Permissions

Compressing

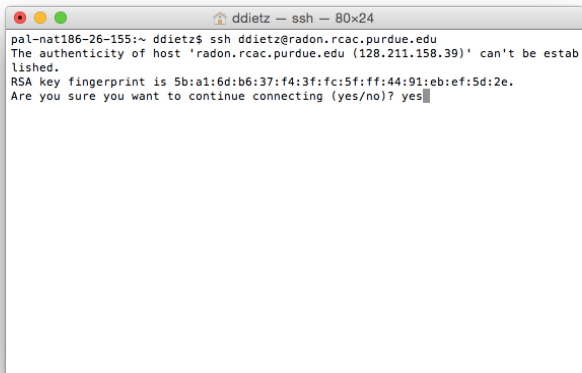
and Archiving

Redirects,

Pipelines and

Scripts

Accept host key by typing yes

A terminal window titled "ddietz — ssh — 80x24" showing an SSH session. The prompt is "pal-nat186-26-155:~ ddietz\$". The user enters "ssh ddietz@radon.rcac.purdue.edu". The terminal displays a warning about the host's authenticity and its RSA key fingerprint. The user responds with "yes".

```
pal-nat186-26-155:~ ddietz$ ssh ddietz@radon.rcac.purdue.edu
The authenticity of host 'radon.rcac.purdue.edu (128.211.158.39)' can't be estab
lished.
RSA key fingerprint is 5b:a1:6d:b6:37:f4:3f:fc:5f:ff:44:91:eb:ef:5d:2e.
Are you sure you want to continue connecting (yes/no)? yes
```

Logging In

Mac

Acknowledgments

Logging In

Windows

Mac

Linux

Files and

Directories

First

Command

Basic

Commands

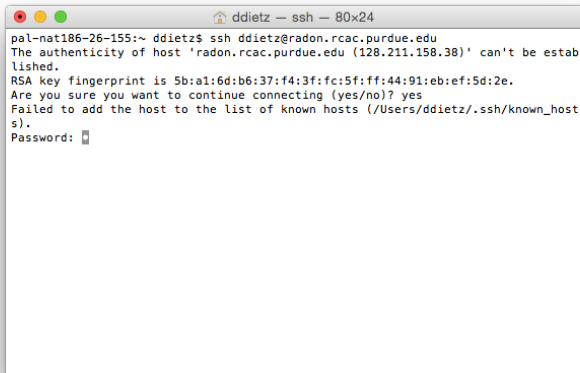
File

Permissions

Compressing and Archiving

Redirects, Pipelines and Scripts

Type in Career Account password when prompted



```
pal-nat186-26-155:~ ddietz$ ssh ddietz@radon.rcac.purdue.edu
The authenticity of host 'radon.rcac.purdue.edu (128.211.158.38)' can't be estab
lished.
RSA key fingerprint is 5b:a1:6d:b6:37:f4:3f:fc:5f:ff:44:91:eb:ef:5d:2e.
Are you sure you want to continue connecting (yes/no)? yes
Failed to add the host to the list of known hosts (/Users/ddietz/.ssh/known_host
s).
Password: 
```

Logging In

Mac

Acknowledgments

Logging In

Windows

Mac

Linux

Files and

Directories

First

Command

Basic

Commands

File

Permissions

Compressing

and Archiving

Redirects,

Pipelines and

Scripts

You should now be logged in!

```
ddietz — ssh — 80x24
Are you sure you want to continue connecting (yes/no)? yes
Failed to add the host to the list of known hosts (/Users/ddietz/.ssh/known_hosts).
Password:
Last login: Thu Oct 15 22:08:48 2015 from c-68-60-239-137.hsd1.in.comcast.net
Please send email to rcac-help@purdue.edu to report problems with this cluster.
=====
==                               Welcome to the Radon Cluster                               ==
== Radon is composed of hyperthreaded 4-core Intel Xeon E3-1284 CPUs and ==
== 10 Gbit Ethernet. Your scratch is mounted at the path specified in ==
== the environment variable $RCAC_SCRATCH. ==
==                               ==
== New mailing list for researchers interested in high performance ==
== computing for their work. ==
== Info: https://lists.purdue.edu/mailman/listinfo/hpc ==
== Subscribe with email to 'hpc-subscribe@lists.purdue.edu' ==
==                               ==
=====
Latest module updates:
intel 13.1.1.163 was updated on Thu Oct  1 10:06
intel 14.0.2.144 was updated on Thu Oct  1 10:06
espresso 4.3.2_impi-4.1.1.036_intel-13.1.1.163 was updated on Wed Oct  7 13:06
espresso 5.2.1_impi-5.0.3.048_intel-15.0.3.187 was updated on Wed Oct 14 15:36
-bash-4.1$
```

Logging In

Linux

Acknowledgments

Logging In

Windows

Mac

Linux

Files and

Directories

First

Command

Basic

Commands

File

Permissions

Compressing

and Archiving

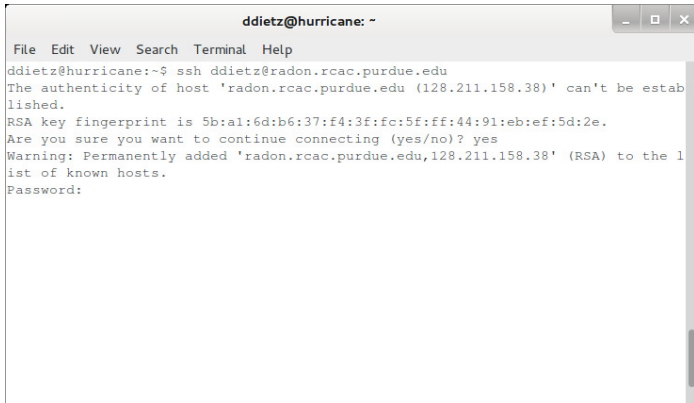
Redirects,

Pipelines and

Scripts

Linux also has a built in terminal client, similar to Mac:

```
ssh myusername@radon.rcac.purdue.edu
```



The screenshot shows a terminal window titled "ddietz@hurricane: ~". The window has a menu bar with "File", "Edit", "View", "Search", "Terminal", and "Help". The terminal content shows the user typing "ssh ddietz@radon.rcac.purdue.edu". The output indicates a connection attempt to 128.211.158.38, which failed to establish. It displays the RSA key fingerprint "5b:a1:6d:b6:37:f4:3f:fc:5f:ff:44:91:eb:ef:5d:2e" and asks for confirmation to continue. The user responds "yes". A warning message states: "Warning: Permanently added 'radon.rcac.purdue.edu,128.211.158.38' (RSA) to the list of known hosts." The prompt "Password:" is shown at the bottom of the visible text.

```
ddietz@hurricane: ~  
File Edit View Search Terminal Help  
ddietz@hurricane:~$ ssh ddietz@radon.rcac.purdue.edu  
The authenticity of host 'radon.rcac.purdue.edu (128.211.158.38)' can't be estab  
lished.  
RSA key fingerprint is 5b:a1:6d:b6:37:f4:3f:fc:5f:ff:44:91:eb:ef:5d:2e.  
Are you sure you want to continue connecting (yes/no)? yes  
Warning: Permanently added 'radon.rcac.purdue.edu,128.211.158.38' (RSA) to the l  
ist of known hosts.  
Password:
```

Files and Directories

Acknowledgments

Logging In

**Files and
Directories**

Files

Directories

File paths

First

Command

Basic

Commands

File

Permissions

Compressing

and Archiving

Redirects,

Pipelines and

Scripts

Files and Directories

- Files
- Directories
- File paths

Files and Directories

Acknowledgments

Logging In

Files and
Directories

Files
Directories
File paths

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Files and directories are two important constructs in UNIX (and most operating systems).

- Contain your documents, images, code, programs, OS, etc.
- Everything in UNIX is built on files and directories!
- A filesystem is a collection of files and directories stored on a single physical device
 - Often called drives in Windows

Files and Directories

Files

Acknowledgments

Logging In

Files and
Directories

Files
Directories
File paths

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Files store some sort of information or data.

- Two basic types of files:
 - Text (documents, code)
 - Binary (images, executables)
- Have metadata associated with them
 - Name, timestamps, permissions

Files and Directories

Directories

Acknowledgments

Logging In

Files and
Directories

Files
Directories
File paths

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Directories are collections of files and directories

- Analogous and interchangeable with folders
- Have metadata associated with them
 - Name, timestamps, permissions

Files and Directories

File paths

Acknowledgments

Logging In

Files and
Directories

Files
Directories
File paths

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

In UNIX all files and directories have a "path", which is the "path" of directories you must follow to get to the file. Directories in the "path" to a file are separated by a "/" .

Examples:

- `/home/ddietz`
- `/home/ddietz/`
- `/home/ddietz/file.txt`

File extensions don't matter in UNIX. Good practice is to use standard extensions to quickly identify a file type.

Files and Directories

File paths

Acknowledgments

Logging In

Files and
Directories

Files
Directories
File paths

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Absolute paths

- The path to a file starting at the root of the system
- Begins with / to denote the path starts at the root
- Guaranteed to get you there

Relative paths

- The path to a file starting at the current location
- Indicate current directory with .
- and parent directory as ..
- Can break if you start in the wrong place!

Files and Directories

File paths

Acknowledgments

Logging In

Files and
Directories

Files
Directories
File paths

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Examples:

Assume current location is: `/home/ddietz/`

Relative Path	Absolute Path
<code>file.txt</code>	<code>/home/ddietz/file.txt</code>
<code>./file.txt</code>	<code>/home/ddietz/file.txt</code>
<code>files/file.txt</code>	<code>/home/ddietz/files/file.txt</code>
<code>../gandino/files/</code>	<code>/home/gandino/files/</code>
<code>../../depot/</code>	<code>/depot/</code>
<code>../gandino/../../home/ddietz/file.txt</code>	<code>/home/ddietz/file.txt</code>

First Command

Acknowledgments

Logging In

Files and
Directories

**First
Command**

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

First Command

First Command

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

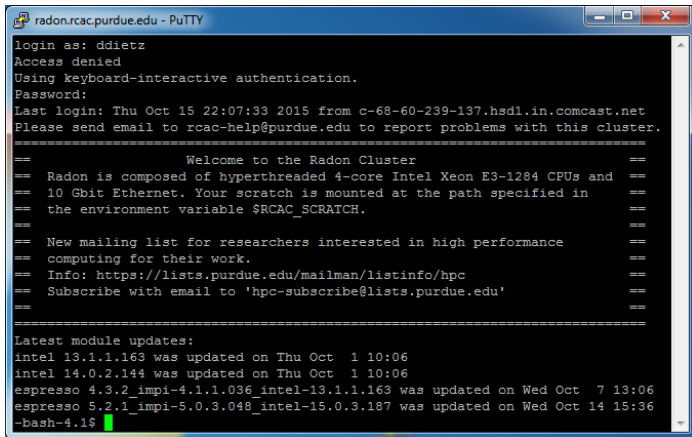
File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

You will be greeted with a Message of the Day (MOTD)

Command prompt: `myusername@hostname: $`



```
radon.rcac.purdue.edu - PuTTY
login as: ddietz
Access denied
Using keyboard-interactive authentication.
Password:
Last login: Thu Oct 15 22:07:33 2015 from c-68-60-239-137.hsd1.in.comcast.net
Please send email to rcac-help@purdue.edu to report problems with this cluster.
=====
                Welcome to the Radon Cluster                =====
==  Radon is composed of hyperthreaded 4-core Intel Xeon E3-1284 CPUs and  ==
==  10 Gbit Ethernet. Your scratch is mounted at the path specified in  ==
==  the environment variable $RCAC_SCRATCH.                               ==
==                                                                    ==
==  New mailing list for researchers interested in high performance      ==
==  computing for their work.                                           ==
==  Info: https://lists.purdue.edu/mailman/listinfo/hpc                 ==
==  Subscribe with email to 'hpc-subscribe@lists.purdue.edu'           ==
==                                                                    ==
=====
Latest module updates:
intel 13.1.1.163 was updated on Thu Oct  1 10:06
intel 14.0.2.144 was updated on Thu Oct  1 10:06
espresso 4.3.2_impi-4.1.1.036_intel-13.1.1.163 was updated on Wed Oct  7 13:06
espresso 5.2.1_impi-5.0.3.048_intel-15.0.3.187 was updated on Wed Oct 14 15:36
-bash-4.1$
```

First Command

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

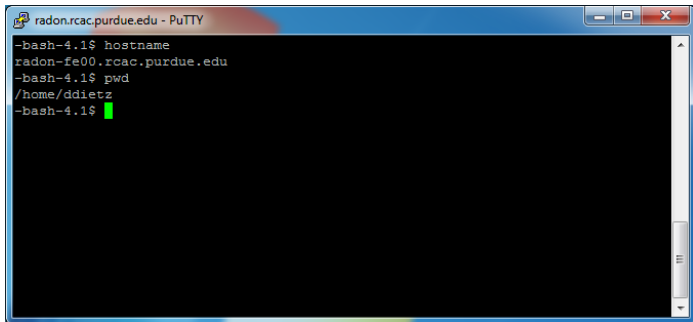
Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Always be aware of where you are in Unix

Get a couple of commands trained in muscle memory:

- `hostname` what machine am I on?
- `pwd` what directory am I in?



```
radon.rcac.purdue.edu - PuTTY
-bash-4.1$ hostname
radon-fe00.rcac.purdue.edu
-bash-4.1$ pwd
/home/ddietz
-bash-4.1$
```


First Command

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

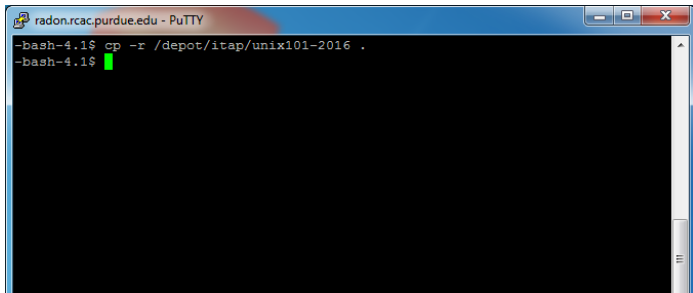
Redirects,
Pipelines and
Scripts

Lets run our first command and grab workshop files

```
$ cp -r /depot/itap/unix101 .
```

Spaces separate these parts! Breaking it down:

cp	-r	/depot/itap/unix101	.
Command	Flag	Argument	Argument



The screenshot shows a PuTTY terminal window titled "radon.rcac.purdue.edu - PuTTY". The terminal displays a prompt "-bash-4.1\$" followed by the command "cp -r /depot/itap/unix101-2016 .". The command has been executed, and the prompt is now "-bash-4.1\$" with a green cursor. The terminal window has a blue title bar and standard window controls (minimize, maximize, close) in the top right corner.

Basic Commands

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls
cd
Exercises
File Structure
mkdir
cp
mv
Exercises
Viewing and
Editing Files
Deleting files
Exercises
scp

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and

Basic Commands

- ls
- cd
- Exercises
- File Structure
- mkdir
- cp
- mv
- Exercises
- Viewing and Editing Files
- Deleting files
- Exercises
- scp

Basic Commands

Acknowledgments

Logging In

Files and Directories

First Command

Basic Commands

ls cd

Exercises File Structure

mkdir

cp mv

Exercises

Viewing and Editing Files

Deleting files Exercises

scp

File Permissions

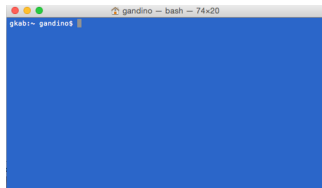
Compressing and Archiving

Redirects, Pipelines and

UNIX: is a text oriented operating system and is the primary operating system used at high performance computing facilities, as well as underlies the Mac OSX graphical operating system. You interact with the computer via a shell.

The shell is a program that interprets commands and acts as an intermediary between the user and the inner workings of the operating system.

- Navigation
- File structure
- Editing files
- Examining files
- File transfer



Basic Commands

Acknowledgments

Logging In

You should be logged in Radon:

Files and
Directories

```
ssh myusername@radon.rcac.purdue.edu
```

First
Command

`pwd` (**p**rint **w**orking **d**irectory): prints working directory

Basic
Commands

`/home/myusername` myusername is your Purdue username

ls
cd
Exercises
File Structure
mkdir

```
$ pwd  
/home/gandino
```

cp
mv
Exercises
Viewing and
Editing Files
Deleting files
Exercises
scp

Try a couple more commands:

File
Permissions

```
$ cp -r /depot/itap/unix101 ~/unix101/BACKUP  
$ cd unix101  
$ pwd  
/home/gandino/unix101
```

Compressing
and Archiving

Redirects,
Pipelines and

Basic Commands

ls

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls

cd

Exercises

File Structure

mkdir

cp

mv

Exercises

Viewing and
Editing Files

Deleting files

Exercises

scp

File

Permissions

Compressing
and Archiving

Redirects,
Pipelines and

The `ls` (**list**) command files and directories in a directory.

General syntax:

```
ls [OPTIONS] [FILENAME]
```

OPTIONS include:

- `-l` long listing, includes file date and size
- `-a` displays all files
- `-h` show file sizes in human readable terms
- `-t` show the newest files first

Example:

```
$ cd ~/unix101/
```

```
$ ls
```

```
BACKUP basic_commands data protein redirects regex scripts  
Shakespeare
```

Basic Commands

ls

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls

cd

Exercises

File Structure

mkdir

cp

mv

Exercises

Viewing and
Editing Files

Deleting files

Exercises

scp

File

Permissions

Compressing
and Archiving

Redirects,

Pipelines and

Long listing (permissions, link count, owner, group, bytes, date, name) - more on this later:

```
$ ls -l
drwxr-xr-x 6 gandino entm 111 Sep  8 15:31 BACKUP
drwxr-xr-x 2 gandino entm 157 May 19 09:37 basic_commands
drwxr-xr-x 2 gandino entm 163 May 19 09:37 protein
drwxr-xr-x 3 gandino entm 346 Jun  2 12:08 scripts
drwxr-xr-x 2 gandino entm 259 May 26 12:07 Shakespeare
...
```

Long listing + hidden files (any file starting with "." dot):

```
$ ls -la
drwxr-xr-x 7 gandino entm 135 Sep  8 15:31 .
drwx----- 63 gandino entm 4007 Sep  8 15:23 ..
drwxr-xr-x 6 gandino entm 111 Sep  8 15:31 BACKUP
drwxr-xr-x 2 gandino entm 157 May 19 09:37 basic_commands
drwxr-xr-x 2 gandino entm 163 May 19 09:37 protein
drwxr-xr-x 3 gandino entm 346 Jun  2 12:08 scripts
drwxr-xr-x 2 gandino entm 259 May 26 12:07 Shakespeare
...
```

Basic Commands

ls

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls
cd
Exercises
File Structure
mkdir

cp
mv
Exercises
Viewing and
Editing Files
Deleting files
Exercises
scp

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and

Long listing + hidden files + human readable file size (note how the second file is now 4.0K; especially useful for megabyte or gigabyte files which have a long number):

```
$ ls -lah
drwxr-xr-x  7 gandino entm  135 Sep  8 15:31 .
drwx----- 63 gandino entm 4.0K Sep  8 15:23 ..
drwxr-xr-x  6 gandino entm  111 Sep  8 15:31 BACKUP
drwxr-xr-x  2 gandino entm  157 May 19 09:37 basic_commands
drwxr-xr-x  2 gandino entm  163 May 19 09:37 protein
drwxr-xr-x  3 gandino entm  346 Jun  2 12:08 scripts
drwxr-xr-x  2 gandino entm  259 May 26 12:07 Shakespeare
...
```

Basic Commands

ls

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls

cd

Exercises

File Structure

mkdir

cp

mv

Exercises

Viewing and
Editing Files

Deleting files

Exercises

scp

File

Permissions

Compressing
and Archiving

Redirects,
Pipelines and

Long listing + hidden files + human readable + sort by date:

```
$ ls -laht
drwx----- 63 gandino entm 4.0K Sep  8 15:23 ..
drwxr-xr-x  6 gandino entm 111 Sep  8 15:31 BACKUP
drwxr-xr-x  7 gandino entm 135 Sep  8 15:31 .
drwxr-xr-x  3 gandino entm 346 Jun  2 12:08 scripts
drwxr-xr-x  2 gandino entm 259 May 26 12:07 Shakespeare
drwxr-xr-x  2 gandino entm 163 May 19 09:37 protein
drwxr-xr-x  2 gandino entm 157 May 19 09:37 basic_commands
...
```


Basic Commands

cd

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls
cd

Exercises
File Structure
mkdir

cp
mv

Exercises
Viewing and
Editing Files
Deleting files
Exercises
scp

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and

The `cd` (**c**hange **d**irectory) command is used to jump from one directory to another

General syntax:

```
cd [DIRECTORY]
```

Change your present location to the parent directory:

```
$ cd ..
```

Change your present location to your home directory:

```
$ cd
```

The directory which is up one level in the directory tree can be referred to as `..` (dot dot).

Basic Commands

cd

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls
cd

Exercises
File Structure
mkdir

cp
mv

Exercises
Viewing and
Editing Files

Deleting files
Exercises
scp

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and

Try yourself:

```
$ cd ~/unix101/basic_commands
$ pwd
/home/gandino/unix101/basic_commands/
$ cd ..
$ pwd      # where did that get you?
/home/gandino/unix101/
$ ls
BACKUP basic_commands  data  protein  redirects  regex  scripts
      Shakespeare
```

Let's try different ways to get to our home directory:

OR

```
$ cd /home/gandino/unix101
$ cd ~
$ pwd
/home/gandino
$ ls
unix101
```

```
$ cd /home/gandino/unix101
$ cd /home/gandino
$ pwd
/home/gandino
$ ls
unix101
```

Basic Commands

cd

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls
cd

Exercises
File Structure
mkdir

cp
mv

Exercises
Viewing and
Editing Files
Deleting files
Exercises
scp

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and

There are a few special shortcuts for `cd`:

<code>cd</code>	move to home directory
<code>cd ~</code>	move to home directory
<code>cd ..</code>	move up one directory
<code>cd -</code>	move back to the last directory you were in

Special directories:

<code>.</code>	current directory
<code>..</code>	parent directory
<code>~</code>	your home directory
<code>~someusername</code>	another user's home directory

Basic Commands

Exercises

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls
cd

Exercises
File Structure

mkdir
cp
mv

Exercises
Viewing and
Editing Files

Deleting files
Exercises

scp

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and

Try the following command sequence (**starting from your home directory!**):

1. `cd unix101/`
2. `pwd` `(/home/gandino/unix101)`
3. `ls -al`
4. `cd basic_commands/`
5. `pwd` `(/home/gandino/unix101/basic_commands)`
6. `cd .`
7. `pwd` `(/home/gandino/unix101/basic_commands)`
8. `cd ..`

Basic Commands

Exercises

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls
cd

Exercises
File Structure

mkdir
cp
mv

Exercises
Viewing and
Editing Files

Deleting files
Exercises

scp

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and

Try the following command sequence. Verify the output of each `pwd` with the path in red.

9. `pwd` (/home/gandino/unix101)
10. `ls -al`
11. `cd ..`
12. `pwd` (/home/gandino)
13. `cd ..`
14. `pwd` (/home)
15. `ls -al`
16. `cd ~/unix101/`

Basic Commands

Exercises

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls

cd

Exercises

File Structure

mkdir

cp

mv

Exercises

Viewing and
Editing Files

Deleting files

Exercises

scp

File

Permissions

Compressing
and Archiving

Redirects,
Pipelines and

Answers:

```
1) $ cd unix101/
2) $ pwd
/home/gandino/unix101
3) $ ls -al
drwxr-xr-x  7 gandino entm  135 Sep  8 15:31 .
drwx----- 63 gandino entm 4.0K Sep  8 23:25 ..
drwxr-xr-x  6 gandino entm  111 Sep  8 15:31 BACKUP
drwxr-xr-x  2 gandino entm  157 May 19 09:37 basic_commands
drwxr-xr-x  2 gandino entm  163 May 19 09:37 protein
drwxr-xr-x  3 gandino entm  346 Jun  2 12:08 scripts
drwxr-xr-x  2 gandino entm  259 May 26 12:07 Shakespeare
...
4) $ cd basic_commands/
5) $ pwd
/home/gandino/unix101/basic_commands
```

Basic Commands

Exercises

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls
cd

Exercises
File Structure
mkdir

cp
mv

Exercises
Viewing and
Editing Files

Deleting files
Exercises

scp

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and

Answers:

```
6) $ cd .
7) $ pwd
/home/gandino/unix101/basic_commands
8) $ cd ..
9) $ pwd
/home/gandino/unix101
10) $ ls -al
drwxr-xr-x  7 gandino entm   135 Sep  8 15:31 .
drwx----- 63 gandino entm  4.0K Sep  8 23:25 ..
drwxr-xr-x  6 gandino entm   111 Sep  8 15:31 BACKUP
drwxr-xr-x  2 gandino entm   157 May 19 09:37 basic_commands
drwxr-xr-x  2 gandino entm   163 May 19 09:37 protein
drwxr-xr-x  3 gandino entm   346 Jun  2 12:08 scripts
drwxr-xr-x  2 gandino entm   259 May 26 12:07 Shakespeare
...
```

Basic Commands

Exercises

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls
cd

Exercises

File Structure

mkdir

cp

mv

Exercises

Viewing and
Editing Files

Deleting files

Exercises

scp

File

Permissions

Compressing
and Archiving

Redirects,
Pipelines and

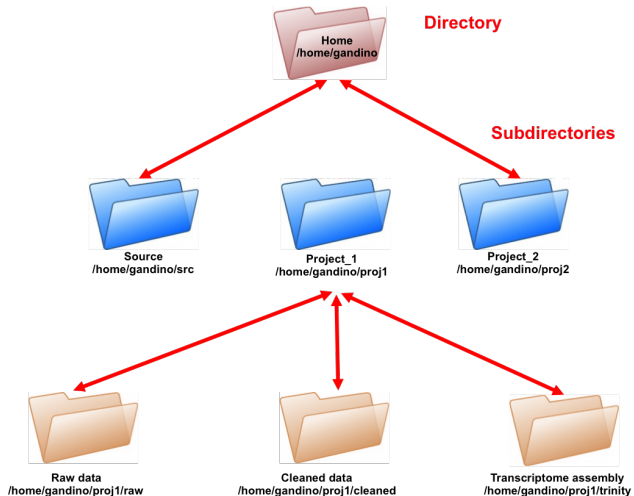
Answers:

```
11) $ cd ..
12) $ pwd
/home/gandino
13) $ cd ..
14) $ pwd
/home
15) $ ls -al # what happens now?
ls: cannot open directory .: Permission denied
16) $ cd ~/unix101/
```


Basic Commands

File Structure

Organize your work!



Basic Commands

mkdir

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls
cd

Exercises
File Structure

mkdir

cp

mv

Exercises

Viewing and
Editing Files

Deleting files

Exercises

scp

File

Permissions

Compressing
and Archiving

Redirects,
Pipelines and

The `mkdir` (**make directory**) command creates a new directory.

General syntax:

```
mkdir [OPTIONS] DIRECTORY
```

OPTIONS include:

- `-p` create parent directories (can create several levels of directories at once)

Example:

```
$ cd ~/unix101/basic_commands
$ mkdir NEW_DIRECTORY
$ ls -lh
total 296M
-rwxr-xr-x 1 gandino entm 2.1K Sep  8 15:29 intro_basic-unix.txt
drwxr-xr-x 2 gandino entm    0 Sep  9 16:53 NEW_DIRECTORY
...
```

Basic Commands

cp

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls
cd

Exercises
File Structure
mkdir

cp
mv

Exercises
Viewing and
Editing Files

Deleting files
Exercises
scp

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and

The `cp` (**copy**) command is used to copy a file or directory.

General syntax:

`cp [OPTIONS] SOURCE DESTINATION`

OPTIONS include:

- `-r` recursively copy a directory, all files and subdirectories inside it

Example:

```
$ cd ~/unix101/basic_commands/NEW_DIRECTORY
$ ls      # this directory should be empty!
$ pwd
/home/gandino/unix101/basic_commands/NEW_DIRECTORY
$ cp ../intro-basic-unix.txt . # specify both source & destination!!
                                # note the . at the end!

$ ls
intro-basic-unix.txt
```

Basic Commands

cp

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls

cd

Exercises

File Structure

mkdir

cp

mv

Exercises

Viewing and
Editing Files

Deleting files

Exercises

scp

File

Permissions

Compressing
and Archiving

Redirects,
Pipelines and

When copying directories make sure to use the option `-r` to copy directories recursively. This means that every file and subdirectory inside that directory will be copied.

Example:

```
$ pwd
/home/gandino/unix101/basic_commands/NEW_DIRECTORY
$ cp -r ../../BACKUP/ . # try one or the other
$ cp -r ../../BACKUP/ BACKUP_FILES
$ ls -lh
total 28K
drwxr-xr-x 6 gandino entm 111 Sep  9 18:02 BACKUP_FILES
-rwxr-xr-x 1 gandino entm 2.1K Sep  9 18:01 intro_basic-unix.txt
```

Basic Commands

mv

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls
cd

Exercises
File Structure

mkdir
cp

mv

Exercises
Viewing and
Editing Files

Deleting files
Exercises

scp

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and

The `mv` (**m**ove) command is used to move or rename a file or directory.

General syntax:

`mv SOURCE DESTINATION`

Example:

```
$ pwd
/home/gandino/unix101/basic_commands/NEW_DIRECTORY
$ cd ..
$ pwd
/home/gandino/unix101/basic_commands
$ ls -lh # pay attention to the list of files
$ mv intro-basic-unix.txt NEW_DIRECTORY/ # file inside directory
# be sure to specify both source and destination!
# notice we just overwrote this file!
$ ls -lh NEW_DIRECTORY/ # now the file should be in here
```

Basic Commands

mv

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls

cd

Exercises

File Structure

mkdir

cp

mv

Exercises

Viewing and
Editing Files

Deleting files

Exercises

scp

File

Permissions

Compressing
and Archiving

Redirects,
Pipelines and

mv can also be used to rename files and directories

mv oldname newname

Example:

```
$ pwd
/home/gandino/unix101/basic_commands
$ mv SP_R1.list list_of_reads.txt # instead of destination directory
                                   # provide new name for file
$ ls -lh
total 296M
drwxr-xr-x 3 gandino entm 157 Sep 11 19:43 .
drwxr-xr-x 7 gandino entm 135 Sep  8 15:31 ..
-rwxr-xr-x 1 gandino entm 197M Sep  8 15:29 list_of_reads.txt
drwxr-xr-x 3 gandino entm  68 Sep  9 18:10 NEW_DIRECTORY
-rwxr-xr-x 1 gandino entm 46M Sep  8 15:29 sequences.fasta
-rwxr-xr-x 1 gandino entm 1.6M Sep  8 15:29 SP_R1.fastq
-rwxr-xr-x 1 gandino entm 1.6M Sep  8 15:29 SP_R2.fastq
```

Basic Commands

Exercises

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls
cd
Exercises
File Structure
mkdir

cp
mv
Exercises

Viewing and
Editing Files
Deleting files
Exercises
scp

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and

Go into the subdirectory called NEW_DIRECTORY:

```
$ cd ~/unix101/basic_commands/NEW_DIRECTORY
```

Note: Verify all exercises by doing `ls -al`

1. Copy `intro_basic-unix.txt` to `intro_basic-unix.txt.copy`
2. Rename `intro_basic-unix.txt.copy` to `intro_basic-unix.2`
3. Create a new directory `new2` in the `NEW_DIRECTORY` directory.
4. Move `intro_basic-unix.2` into the `new2` directory.
5. Move `intro_basic-unix.txt` into the `new2` directory.
6. Move `intro_basic-unix.txt` in the `new2` directory back into the `NEW_DIRECTORY` directory and rename to `"intro_basic-unix.old"`
7. Change directory to `~/unix101/basic_commands/`

Basic Commands

Exercises

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls
cd

Exercises

File Structure
mkdir

cp
mv

Exercises

Viewing and
Editing Files

Deleting files
Exercises

scp

File

Permissions

Compressing
and Archiving

Redirects,
Pipelines and

1. Copy intro_basic-unix.txt to intro_basic-unix.txt.copy

```
$ cp intro_basic-unix.txt intro_basic-unix.txt.copy
$ ls -al
total 65
drwxr-xr-x 3 gandino entm 111 Sep 11 20:49 .
drwxr-xr-x 3 gandino entm 157 Sep 11 19:43 ..
drwxr-xr-x 6 gandino entm 111 Sep  9 18:02 BACKUP_FILES
-rwxr-xr-x 1 gandino entm 2145 Sep  8 15:29 intro_basic-unix.txt
-rwxr-xr-x 1 gandino entm 2145 Sep 11 20:49 intro_basic-unix.txt.
      copy
```


Basic Commands

Exercises

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls
cd

Exercises
File Structure
mkdir

cp
mv

Exercises
Viewing and
Editing Files
Deleting files
Exercises
scp

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and

2. Rename intro_basic-unix.txt.copy to intro_basic-unix.2

```
$ mv intro_basic-unix.txt.copy intro_basic-unix.2
$ ls -al
total 89
drwxr-xr-x 3 gandino entm 104 Sep 11 20:53 .
drwxr-xr-x 3 gandino entm 157 Sep 11 19:43 ..
drwxr-xr-x 6 gandino entm 111 Sep  9 18:02 BACKUP_FILES
-rwxr-xr-x 1 gandino entm 2145 Sep 11 20:49 intro_basic-unix.2
-rwxr-xr-x 1 gandino entm 2145 Sep  8 15:29 intro_basic-unix.txt
```

Basic Commands

Exercises

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls
cd

Exercises
File Structure
mkdir

cp
mv

Exercises
Viewing and
Editing Files
Deleting files
Exercises
scp

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and

3. Create a new directory new2 in the NEW_DIRECTORY directory.

```
$ mkdir new2
$ ls -al
total 91
drwxr-xr-x 4 gandino entm 126 Sep 11 20:55 .
drwxr-xr-x 3 gandino entm 157 Sep 11 19:43 ..
drwxr-xr-x 6 gandino entm 111 Sep  9 18:02 BACKUP_FILES
-rwxr-xr-x 1 gandino entm 2145 Sep 11 20:49 intro_basic-unix.2
-rwxr-xr-x 1 gandino entm 2145 Sep  8 15:29 intro_basic-unix.txt
drwxr-xr-x 2 gandino entm 4096 Sep 11 20:55 new2
```

Basic Commands

Exercises

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls

cd

Exercises

File Structure

mkdir

cp

mv

Exercises

Viewing and
Editing Files

Deleting files

Exercises

scp

File

Permissions

Compressing
and Archiving

Redirects,
Pipelines and

4. Move `intro_basic-unix.2` into the `new2` directory.

```
$ mv intro_basic-unix.2 new2/
$ ls new2/
intro_basic-unix.2
$ ls -al new2/
total 30
drwxr-xr-x 2 gandino entm  36 Sep 11 20:58 .
drwxr-xr-x 4 gandino entm  90 Sep 11 20:58 ..
-rwxr-xr-x 1 gandino entm 2145 Sep 11 20:49 intro_basic-unix.2
```

Basic Commands

Exercises

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls
cd
Exercises
File Structure
mkdir

cp
mv

Exercises
Viewing and
Editing Files
Deleting files
Exercises
scp

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and

5. Move `intro_basic-unix.txt` into the `new2` directory.

```
$ mv intro_basic-unix.txt new2/
$ ls
BACKUP_FILES  new2 # note file is gone!
$ ls -al new2/ # check list of file in new2 directory!
total 55
drwxr-xr-x 2 gandino entm   74 Sep 11 21:05 .
drwxr-xr-x 4 gandino entm   52 Sep 11 21:05 ..
-rwxr-xr-x 1 gandino entm 2145 Sep 11 20:49 intro_basic-unix.2
-rwxr-xr-x 1 gandino entm 2145 Sep  8 15:29 intro_basic-unix.txt
```

Basic Commands

Exercises

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls

cd

Exercises

File Structure

mkdir

cp

mv

Exercises

Viewing and
Editing Files

Deleting files

Exercises

scp

File

Permissions

Compressing
and Archiving

Redirects,
Pipelines and

6. Move `intro_basic-unix.txt` in the `new2` directory back into the `NEW_DIRECTORY` directory and rename to `intro_basic-unix.old`

```
$ mv new2/intro_basic-unix.txt intro_basic-unix.old
$ ls -al # note new intro_basic-unix.old file
total 66
drwxr-xr-x 4 gandino entm   90 Sep 11 22:26 .
drwxr-xr-x 3 gandino entm  157 Sep 11 19:43 ..
drwxr-xr-x 6 gandino entm  111 Sep  9 18:02 BACKUP_FILES
-rwxr-xr-x 1 gandino entm 2145 Sep  8 15:29 intro_basic-unix.old
drwxr-xr-x 2 gandino entm   36 Sep 11 22:26 new2
$ ls -al new2/ # note file intro_basic-unix.txt gone!
total 30
drwxr-xr-x 2 gandino entm   36 Sep 11 22:26 .
drwxr-xr-x 4 gandino entm   90 Sep 11 22:26 ..
-rwxr-xr-x 1 gandino entm 2145 Sep 11 20:49 intro_basic-unix.2
```

Basic Commands

Exercises

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls

cd

Exercises

File Structure

mkdir

cp

mv

Exercises

Viewing and
Editing Files

Deleting files

Exercises

scp

File

Permissions

Compressing
and Archiving

Redirects,
Pipelines and

7. Change directory to ~/unix101/basic_commands/

```
$ cd ~/unix101/basic_commands/
$ pwd
/home/gandino/unix101/basic_commands
$ ls -al
total 501312
drwxr-xr-x  3 gandino entm      4096 Feb  6 11:16 .
drwxr-xr-x 10 gandino entm      4096 Feb  3 18:14 ..
-rwxr-xr-x  1 gandino entm 205953971 Feb  3 18:13 list_of_reads.txt
drwxr-xr-x  4 gandino entm      4096 Feb  7 16:39 NEW_DIRECTORY
-rw-r--r--  1 gandino entm  47387929 Feb  3 18:13 sequences.fasta
-rw-r--r--  1 gandino entm   1612210 Feb  3 18:13 SP_R1.fastq
-rw-r--r--  1 gandino entm   1612210 Feb  3 18:13 SP_R2.fastq
```

Basic Commands

Viewing and Editing Files

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls
cd

Exercises
File Structure
mkdir

cp
mv

Exercises
Viewing and
Editing Files

Deleting files
Exercises
scp

File
Permissions

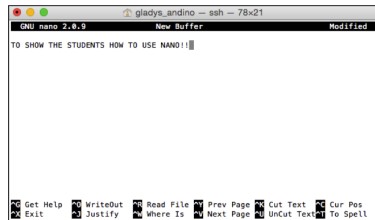
Compressing
and Archiving

Redirects,
Pipelines and

Files can be created and edited using one of several command line text editors. `nano` is one such editor:

`nano` FILENAME

- Type `nano` at the command prompt to start the editor
- Type your commands into the screen
- Move around with the arrow keys
- Save your file with `Ctrl+O`, and provide a name for the file
- Quit the editor with `Ctrl+X`



Basic Commands

Viewing and Editing Files

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls

cd

Exercises

File Structure

mkdir

cp

mv

Exercises

Viewing and
Editing Files

Deleting files

Exercises

scp

File

Permissions

Compressing
and Archiving

Redirects,

Pipelines and

The `less` command displays file contents on the screen with line scrolling.

General syntax:

`less FILENAME`

To scroll you can use arrow keys, PgUp/PgDn keys, space bar, or enter key. **When you are done press 'q' to exit.**

Example:

```
$ cd ~/unix101/basic_commands
$ pwd
/home/gandino/unix101/basic_commands
# copy this file back to its original name and location
$ cp NEW_DIRECTORY/intro_basic-unix.old intro_basic-unix.txt
$ less intro_basic-unix.txt
```


Basic Commands

Viewing and Editing Files

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls

cd

Exercises

File Structure

mkdir

cp

mv

Exercises

Viewing and
Editing Files

Deleting files

Exercises

scp

File

Permissions

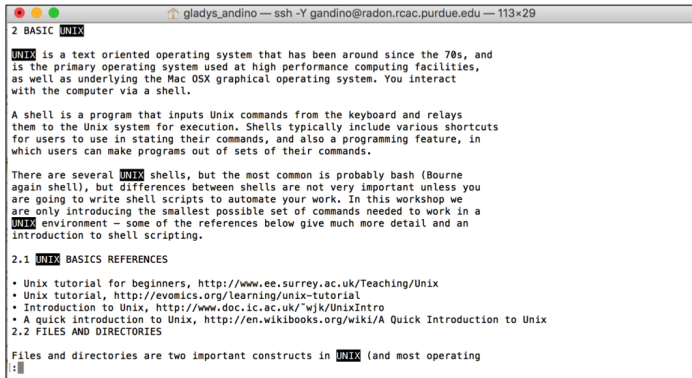
Compressing
and Archiving

Redirects,

Pipelines and

Files can be searched using `less`. Type `/` immediately followed by a search term. Press enter key to search. Navigate through matches with `'n'`.

Example (`/UNIX`):



```
gladys_andino — ssh -Y gandino@radon.rcac.purdue.edu — 113x29
2 BASIC UNIX
UNIX is a text oriented operating system that has been around since the 70s, and
is the primary operating system used at high performance computing facilities,
as well as underlying the Mac OSx graphical operating system. You interact
with the computer via a shell.

A shell is a program that inputs Unix commands from the keyboard and relays
them to the Unix system for execution. Shells typically include various shortcuts
for users to use in stating their commands, and also a programming feature, in
which users can make programs out of sets of their commands.

There are several UNIX shells, but the most common is probably bash (Bourne
again shell), but differences between shells are not very important unless you
are going to write shell scripts to automate your work. In this workshop we
are only introducing the smallest possible set of commands needed to work in a
UNIX environment – some of the references below give much more detail and an
introduction to shell scripting.

2.1 UNIX BASICS REFERENCES
• Unix tutorial for beginners, http://www.ee.surrey.ac.uk/Teaching/Unix
• Unix tutorial, http://evomics.org/learning/unix-tutorial
• Introduction to Unix, http://www.doc.ic.ac.uk/~wjk/UnixIntro
• A quick introduction to Unix, http://en.wikibooks.org/wiki/A\_Quick\_Introduction\_to\_Unix
2.2 FILES AND DIRECTORIES

Files and directories are two important constructs in UNIX (and most operating
systems)
```

Basic Commands

Viewing and Editing Files

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls

cd

Exercises

File Structure

mkdir

cp

mv

Exercises

Viewing and
Editing Files

Deleting files

Exercises

scp

File

Permissions

Compressing
and Archiving

Redirects,
Pipelines and

The cat (**cat**enate) command displays the entire file on your screen.

General syntax:

cat FILENAME

Simplest form of displaying contents. It prints the entire contents of the file on the screen. In case of large files, entire file will scroll on the screen without pausing.

Example:

```
$ cat intro_basic-unix.txt
2 BASIC UNIX
```

UNIX is a text oriented operating system that has been around since the 70s, and is the primary operating system used at high...

Basic Commands

Viewing and Editing Files

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls
cd
Exercises
File Structure
mkdir
cp
mv
Exercises
Viewing and
Editing Files
Deleting files
Exercises
scp

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and

The `head` command displays the starting lines of a file.

General syntax:

```
head [OPTIONS] FILENAME
```

OPTIONS include:

- `-n N` print first N lines

The default is first ten lines. But, any number of lines can be displayed using `-n` option (followed by required number of lines).

Example:

```
$ head -n 1 intro_basic-unix.txt
2 BASIC UNIX
```

Basic Commands

Viewing and Editing Files

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls
cd
Exercises
File Structure
mkdir

cp
mv
Exercises

**Viewing and
Editing Files**

Deleting files
Exercises
scp

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and

The `tail` command displays the last lines of a file.

General syntax:

```
tail [OPTIONS] FILENAME
```

OPTIONS include:

- `-n N` print last N lines

Similar to `head`, but displays the last 10 lines. Again `-n` option can be used to change this.

Example:

```
$ tail -n 1 intro_basic-unix.txt  
files, directories, and the details of this command in more detail  
later.
```

Basic Commands

Viewing and Editing Files

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls

cd

Exercises

File Structure

mkdir

cp

mv

Exercises

Viewing and
Editing Files

Deleting files

Exercises

scp

File

Permissions

Compressing
and Archiving

Redirects,
Pipelines and

The `grep` command is one of the most commonly used commands in UNIX and it is commonly used to filter a file, line by line, against a pattern (e.g., to print each line which matches the pattern).

General syntax:

```
grep [OPTIONS] PATTERN FILENAME
```

OPTIONS include:

- `-c` count lines that match PATTERN
- `-i` ignore case for PATTERN
- `-v` select lines that do not match (invert match)

We will only briefly cover `grep` in this workshop. It will be covered in much more detail in Part 2.

Basic Commands

Viewing and Editing Files

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls
cd

Exercises

File Structure

mkdir

cp

mv

Exercises

**Viewing and
Editing Files**

Deleting files

Exercises

scp

File

Permissions

Compressing
and Archiving

Redirects,
Pipelines and

A handy trick for bioinformaticians: how many sequences are in a FASTA-formatted file? By definition, each sequence record in a FASTA file has one line of description that always starts with `>` followed by multiple lines of sequence itself. Each sequence record ends when the next line starting with `>` appears:

Example:

```
$ pwd
/home/gandino/unix101/basic_commands/
$ grep -c '>' sequences.fasta
31925
```

Basic Commands

Deleting files

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls
cd
Exercises
File Structure
mkdir

cp
mv
Exercises
Viewing and
Editing Files
Deleting files
Exercises
scp

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and

The `rmdir` (**r**emove **d**irectory) is used to delete directories.

General syntax:

```
rmdir DIRECTORY
```

Directories must be empty before you use the `rmdir` command.

Basic Commands

Deleting files

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls
cd

Exercises
File Structure

mkdir
cp

mv
Exercises

Viewing and
Editing Files

Deleting files
Exercises

scp

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and

The `rm` (**r**emove) command is used to delete files and directories.

General syntax:

```
rm [OPTIONS] DIRECTORY
```

OPTIONS include:

- `-r` recursively delete files and directories

Individual files can be deleted with `rm` or you can recursively delete a directory (**CAREFUL!**) with the `-r` option. Unlike `rmdir` the `-r` option will delete directories even if they are not empty!

Basic Commands

Exercises

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls
cd
Exercises
File Structure
mkdir

cp
mv
Exercises
Viewing and
Editing Files
Deleting files
Exercises
scp

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and

Go into the subdirectory called `basic_commands`:

```
$ cd ~/unix101/basic_commands
```

1. Using `head` list only the first 4 lines of the file `list_of_reads.txt`
2. Using `tail` list only the last 15 lines of the file `list_of_reads.txt`
3. Using `grep` search the file `list_of_reads.txt` for occurrences of `OJOEACXX` and count them
4. Remove the file `intro_basic-unix.old` from the subdirectory `NEW_DIRECTORY`
5. Remove `NEW_DIRECTORY`

Basic Commands

Exercises

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls
cd

Exercises
File Structure
mkdir

cp
mv

Exercises
Viewing and
Editing Files
Deleting files

Exercises
scp

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and

1. Using head list only the first 4 lines of the file
`list_of_reads.txt`

```
$ head -n 4 list_of_reads.txt
@H-148:116:C0J0EACXX:5:1101:2712:1962
@H-148:116:C0J0EACXX:5:1101:5282:1935
@H-148:116:C0J0EACXX:5:1101:6288:1954
@H-148:116:C0J0EACXX:5:1101:6532:1940
```

Basic Commands

Exercises

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls
cd

Exercises

File Structure
mkdir

cp
mv

Exercises

Viewing and
Editing Files

Deleting files

Exercises

scp

File

Permissions

Compressing
and Archiving

Redirects,
Pipelines and

2. Using `tail` list only the last 15 lines of the file
`list_of_reads.txt`

```
$ tail -n 15 list_of_reads.txt
@H-148:116:C0JOEACXX:6:1316:14237:102029
@H-148:116:C0JOEACXX:6:1316:14254:102016
@H-148:116:C0JOEACXX:6:1316:14502:102016
...
@H-148:116:C0JOEACXX:6:1316:18548:102041
@H-148:116:C0JOEACXX:6:1316:19032:102009
@H-148:116:C0JOEACXX:6:1316:19177:102042
@H-148:116:C0JOEACXX:6:1316:19472:102043
```

Basic Commands

Exercises

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls

cd

Exercises

File Structure

mkdir

cp

mv

Exercises

Viewing and
Editing Files

Deleting files

Exercises

scp

File

Permissions

Compressing
and Archiving

Redirects,
Pipelines and

3. Using grep search the file `list_of_reads.txt` for occurrences of `0J0EACXX` and count them

```
$ grep -c "0J0EACXX" list_of_reads.txt  
5213684
```

Basic Commands

Exercises

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls
cd

Exercises
File Structure
mkdir

cp
mv

Exercises
Viewing and
Editing Files

Deleting files
Exercises

scp

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and

4. Remove the file `intro_basic-unix.old` from the subdirectory `NEW_DIRECTORY`

```
$ rm NEW_DIRECTORY/intro_basic-unix.old
$ ls -al NEW_DIRECTORY/
drwxr-xr-x 4 gandino entm   90 Sep 11 22:26 .
drwxr-xr-x 3 gandino entm  157 Sep 11 19:43 ..
drwxr-xr-x 6 gandino entm  111 Sep  9 18:02 BACKUP_FILES
drwxr-xr-x 2 gandino entm   36 Sep 11 22:26 new2
```

Basic Commands

Exercises

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls
cd
Exercises
File Structure
mkdir
cp
mv

Exercises
Viewing and
Editing Files
Deleting files
Exercises
scp

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and

5. Remove NEW_DIRECTORY

```
$ rm -r NEW_DIRECTORY/  
$ ls -la  
drwxr-xr-x 3 gandino entm 157 Sep 11 19:43 .  
drwxr-xr-x 7 gandino entm 135 Sep 8 15:31 ..  
-rwxr-xr-x 1 gandino entm 197M Sep 8 15:29 list_of_reads.txt  
-rwxr-xr-x 1 gandino entm 46M Sep 8 15:29 sequences.fasta  
-rwxr-xr-x 1 gandino entm 1.6M Sep 8 15:29 SP_R1.fastq  
-rwxr-xr-x 1 gandino entm 1.6M Sep 8 15:29 SP_R2.fastq  
-rwxr-xr-x 1 gandino entm 2145 Sep 8 15:29 intro_basic-unix.txt
```

Basic Commands

scp

Acknowledgments

Logging In

Files and Directories

First Command

Basic Commands

ls cd

Exercises File Structure

mkdir

cp mv

Exercises Viewing and Editing Files Deleting files Exercises

scp

File Permissions

Compressing and Archiving

Redirects, Pipelines and

We often have to transfer files between a compute server and a local server, or even from a personal laptop to a compute server. Many programs are available for this, but which ones you can use varies depending on your operating system.

Secure copy (scp) is convenient way to copy one or many files between a local computer and a UNIX system (such as Radon). The following examples assume you are running scp while logged into a local UNIX system (such as your Mac laptop).

Basic Commands

scp

Acknowledgments

Logging In

Files and Directories

First Command

Basic Commands

ls cd

Exercises File Structure

mkdir cp

mv Exercises

Viewing and Editing Files

Deleting files Exercises

scp

File Permissions

Compressing and Archiving

Redirects, Pipelines and

The scp (secure copy) command is used transfer files to and from two servers or local computers.

General syntax:

```
scp [OPTIONS] LOCAL REMOTE
```

```
scp [OPTIONS] REMOTE LOCAL
```

OPTIONS include:

- `-r` recursively copy files and directories
- `-p` preserve timestamps and permissions

File transfers are initiated by specifying a source and destination. **Order is dependent on where you are running the scp command and the direction you are transferring files.**

Basic Commands

scp

Acknowledgments

Logging In

Files and Directories

First Command

Basic Commands

ls

cd

Exercises

File Structure

mkdir

cp

mv

Exercises

Viewing and Editing Files

Deleting files

Exercises

scp

File

Permissions

Compressing and Archiving

Redirects, Pipelines and

Open a new terminal running on your laptop or lab computer. Copy a file from Radon into your laptop terminal's current directory:

```
$ scp myusername@radon.rcac.purdue.edu:~/unix101/basic_commands/  
intro_basic-unix.txt .
```

Copy a file from your laptop terminal's current directory to your Radon home directory:

```
$ scp intro_basic-unix.txt myusername@radon.rcac.purdue.edu:~/
```

If you are transferring to/from a personal/lab computer, you will almost always use `scp` from your computer and use the cluster as the remote host. It may be possible to initiate from the cluster, but requires you to specify your computer as the remote computer, your laptop being on the same network, and knowing your laptop's IP address.

Basic Commands

scp

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

ls
cd
Exercises
File Structure
mkdir

cp
mv
Exercises
Viewing and
Editing Files
Deleting files
Exercises

scp

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and

For Windows users, you can use the WinSCP program:

winscp.net/eng/download.php

This is a graphical drag and drop program for transferring files. Behind the scenes, it uses the scp protocol.

Mac users also have graphical options for transferring files, such as Cyberduck (also works on Windows):

cyberduck.io/

File Permissions

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Permission
Groups
Permission
Types
Changing
Permission

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

File Permissions

- Permission Groups
- Permission Types
- Changing Permission

File Permissions

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Permission
Groups
Permission
Types
Changing
Permission

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Permissions describe "who" (user, group, others, all) can do "what" (read, write, execute) to a file.

`ls -l` to view:

The diagram shows the output of the `ls -l` command for a file named `test`. The output is: `-rwxr-x--- 1 walbert support 0 Oct 31 11:06 test`. Annotations with arrows and brackets explain each part:

- File Type**: Points to the first character `-`.
- Permissions**: Points to the next nine characters `rwxr-x---`. Brackets below group them as `User` (`rwx`), `Other` (`---`), and `Group` (`r-x`).
- # of Hard Links**: Points to the number `1`.
- Owners**: Points to the user and group names `walbert` and `support`. Brackets below identify `walbert` as the `User` and `support` as the `Group`.
- File size**: Points to the number `0`.
- Last Modify Time**: Points to the date and time `Oct 31 11:06`.
- File name**: Points to the filename `test`.

Every user belongs to one or several groups. Every file belongs to one group.

File Permissions

Permission Groups

Acknowledgments

Logging In

Files and
Directories

First
Command

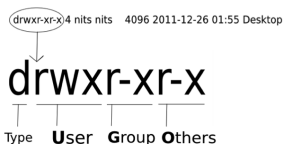
Basic
Commands

File
Permissions

Permission
Groups
Permission
Types
Changing
Permission

Compressing
and Archiving

Redirects,
Pipelines and
Scripts



Each file and directory has three user based permission groups:

user - Applies to only the owner of the file or directory.

group - Applies to the group that has been assigned to the file or directory.

others - Applies to all other users on the system, this is the permission group that you want to use the most caution with.

File Permissions

Permission Types

Acknowledgments

Logging In

Files and
Directories

First
Command

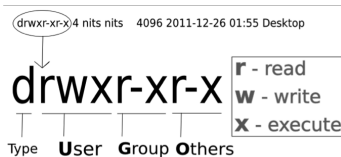
Basic
Commands

File
Permissions

Permission
Groups
Permission
Types
Changing
Permission

Compressing
and Archiving

Redirects,
Pipelines and
Scripts



Each file or directory has three basic permission types:

read - permission to read the contents of the file.

write - permission to write or modify a file or directory.

xecute - permission to execute a file or view the contents of a directory.

"x" has special meaning for directories (allows descending into). Often forgotten when trying to give a colleague access to your files (need to make files readable and all containing directories executable).

File Permissions

Changing Permission

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Permission
Groups
Permission
Types
Changing
Permission

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Permissions can be changed with a few different commands.

Common:

- change permissions: `chmod`

Rare:

- **change owner**: `chown`
- **change group**: `chgrp`

```
# change the ownership of a file named file1 to a new owner alice (  
    requires root):  
$ chown alice file1  
  
# change the owner of a file named file2 to user bob and change its  
    group ownership to group2 (requires root):  
$ chown bob:group2 file2  
  
# change the group ownership of the directory named dir1, and all  
    files and directories inside dir1, to the group group2:  
$ chgrp -R group2 dir1
```

File Permissions

Changing Permission

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Permission
Groups
Permission
Types
Changing
Permission

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

The `chmod` (**ch**ange **mo**de) command changes the permission levels for user, group, and others.

General syntax:

```
chmod [OPTIONS] MODE FILENAME
```

OPTIONS include:

- `-R` recursively change mode on a directory and all of the directories and files inside it

`MODE` can be set in a couple different ways as described on the next slides.

File Permissions

Changing Permission

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Permission
Groups

Permission
Types

Changing
Permission

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Syntax:

```
chmod [OPTIONS] MODE FILENAME
```

The MODE can be changed by describing

1. To whom permission changes should be applied
2. Whether permissions should be added, subtracted, or set to a value
3. What permissions should be changed

File Permissions

Changing Permission

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Permission
Groups
Permission
Types
Changing
Permission

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Syntax:

```
chmod [OPTIONS] MODE FILENAME
```

Build MODE by moving left to right:

Who	Add or Subtract	What
u for user	+ to add permissions	r for read
g for group	- to remove permissions	w for write
o for others	= to set the permissions	x for execute
a for all		

Some examples:

```
$ chmod -R go-rwx privatefiles    # me only
$ chmod go+rx $HOME                # careful here!
$ chmod a+w $HOME                  # DONT!!!
$ chmod +x myscript                # executable script (often used)
```

File Permissions

Changing Permission

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Permission
Groups
Permission
Types

Changing
Permission

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Exercises: Before we begin, create a new file and apply a set of starting permissions. We'll give the new file read and write permissions to the user and group, and allow others to only read the file.

```
$ cd ~/unix101/basic_commands
$ touch example.txt # create a new file example.txt
$ ls -l example.txt
-rw-r--r-- 1 ddietz itap 0 Jan 27 10:13 example.txt
$ chmod u=rw,g=rw,o=r example.txt # change permissions
$ ls -l example.txt
-rw-rw-r-- 1 ddietz itap 0 Jan 27 10:13 example.txt
```

File Permissions

Changing Permission

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Permission
Groups
Permission
Types
Changing
Permission

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Exercises: Select the correct answer and use "ls -l" to check your work

```
$ ls -l example.txt
-rw-rw-r-- 1 ddietz itap 0 Jan 27 10:13 example.txt
```

- | | |
|--|------------------------------------|
| 1. Remove my execute permission to example.txt | a. chmod u-x example.txt |
| 2. Don't allow the group to write to example.txt | b. chmod o=rw,ug-x example.txt |
| 3. Don't allow others to do anything to example.txt | c. chmod a+x example.txt |
| 4. Give execute permission to everyone | d. chmod u+x example.txt |
| 5. Remove execute permission from myself and other members in group itap, while give read and write permission to others | e. chmod o-rwx example.txt |
| 6. Restore the original permission | f. chmod o-w example.txt |
| | g. chmod u=rw,g=rw,o=r example.txt |
| | h. chmod g-w example.txt |

File Permissions

Changing Permission

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Permission
Groups
Permission
Types
Changing
Permission

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Answers in red:

1. Remove my execute permission to example.txt **a**
 2. Don't allow the group to write to example.txt **h**
 3. Don't allow others to do anything to example.txt **e**
 4. Give execute permission to everyone **c**
 5. Remove execute permission from myself and other members in group itap, while give read and write permission to others **b**
 6. Restore the original permission **g**
- a. `chmod u-x example.txt`
 - b. `chmod o=rw,ug-x example.txt`
 - c. `chmod a+x example.txt`
 - d. `chmod u+x example.txt`
 - e. `chmod o-rwx example.txt`
 - f. `chmod o-w example.txt`
 - g. `chmod u=rw,g=rw,o=r example.txt`
 - h. `chmod g-w example.txt`

Compressing and Archiving

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

gzip
tar
zip
Exercises

Redirects,
Pipelines and
Scripts

Compressing and Archiving

- gzip
- tar
- zip
- Exercises

Compressing and Archiving

gzip

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

gzip
tar
zip
Exercises

Redirects,
Pipelines and
Scripts

The `gzip` command compresses or decompresses a single file using gzip compression

General syntax:

`gzip [OPTIONS] FILENAME`

OPTIONS include:

- `-d` decompresses file instead of default of compressing

Example, noting file size change:

```
$ cd ~/unix101/basic_commands
$ ls -lh SP_R1.list
-rwxrwxr-x 1 gandino rcacsupp 197M Feb 15 2016 SP_R1.list
$ gzip SP_R1.list
$ ls -lh SP_R1.list.gz
-rwxrwxr-x 1 gandino rcacsupp 29M Feb 15 2016 SP_R1.list.gz
$ gzip -d SP_R1.list.gz
```

Compressing and Archiving

tar

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

gzip
tar
zip
Exercises

Redirects,
Pipelines and
Scripts

The tar (tape archive) command creates an archive out of a set of files. Historically called "tape archive" because its original intent was to make archives suitable for tape medium. Nowadays, it still does, but is used for general purpose packaging of files.

General syntax:

`tar OPTIONS FILENAME`

OPTIONS include:

- `-c` create tar archive file (requires `-f`)
- `-f FILENAME` use with `-c` to specify name of new archive
- `-t` list contents of a tar archive file (requires `-f`)
- `-x` extract contents of a tar archive file (requires `-f`)
- `-v` verbose (list each file as it is archived or extracted)

Compressing and Archiving

tar

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

gzip
tar
zip
Exercises

Redirects,
Pipelines and
Scripts

Examples:

- Create archive of a directory:

```
tar -cvf NEWARCHIVE.tar DIRECTORY
```

- List files in an archive:

```
tar -tvf NEWARCHIVE.tar
```

- Extract files from an archive:

```
tar -xvf NEWARCHIVE.tar
```

Compressing and Archiving

tar

Acknowledgments

Logging In

Files and Directories

First Command

Basic Commands

File Permissions

Compressing and Archiving

gzip tar zip Exercises

Redirects, Pipelines and Scripts

tar does not compress files by default, it only packages files together in sequential order. A tar file could be compressed in one step or two.

Two steps:

```
tar -cvf NEWARCHIVE.tar DIRECTORY
gzip NEWARCHIVE.tar
```

One step by using additional options to tar:

-z use gzip compression
-j use bzip2 compression

```
tar -czvf NEWARCHIVE.tar.gz DIRECTORY
tar -xzvf NEWARCHIVE.tar.gz
```

Compressing and Archiving

zip

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

gzip
tar
zip
Exercises

Redirects,
Pipelines and
Scripts

The `zip` command archives and compresses a file or directory using zip compression. Typically not used in UNIX world (tar and gzip are most common) but may be found on Windows.

General syntax:

```
zip [OPTIONS] FILENAME
```

OPTIONS include:

- `-r` recursively archive and compress directory

Note: if the target archive already exists, `zip` by default updates it (appends to it) rather than overwrites (not a typical Unix behavior, can yield some weird surprises).

Compressing and Archiving

zip

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

gzip
tar
zip
Exercises

Redirects,
Pipelines and
Scripts

The `unzip` command unpacks and decompresses a zip archive. Typically not used in UNIX world (tar and gzip are most common) but you may occasionally encounter a zip file you need to unpack.

General syntax:

```
unzip [OPTIONS] FILENAME
```

OPTIONS include:

- `-l` list content

Compressing and Archiving

Exercises

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

gzip
tar
zip
Exercises

Redirects,
Pipelines and
Scripts

Change directory:

```
$ cd ~/unix101/basic_commands
```

1. gzip the file `list_of_reads.txt` and examine the size and decompress it back.
2. Archive all `fastq*` files in `basic_commands` directory and name it `fastq.tar`.
3. List the content of file `fastq.tar`.

Compressing and Archiving

Exercises

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

gzip
tar
zip
Exercises

Redirects,
Pipelines and
Scripts

Change directory:

```
$ cd ~/unix101/
```

1. Use one step to archive and compress the BACKUP directory and name it backup.tar.gz.
2. Make a directory named test_tar.
3. Copy backup.tar.gz to the directory test_tar and use two-step method to extract its content.

Compressing and Archiving

Exercises

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

gzip
tar
zip
Exercises

Redirects,
Pipelines and
Scripts

Change directory:

```
$ cd ~/unix101/basic_commands
```

1. gzip the file `list_of_reads.txt` and examine the size and decompress it back.

```
$ gzip list_of_reads.txt
$ ls -lh list_of_reads.txt
-rw-r--r-- 1 goughes itap 29M Feb  8 08:52 list_of_reads.txt.gz
$ gzip -d list_of_reads.txt.gz
$ ls -lh list_of_reads.txt
-rw-r--r-- 1 goughes itap 197M Feb  8 08:52 list_of_reads.txt
```

Compressing and Archiving

Exercises

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

gzip
tar
zip

Exercises

Redirects,
Pipelines and
Scripts

2. Archive all `.fastq` files in `basic_commands` directory and name it `fastq.tar`.

```
$ tar -cvf fastq.tar SP_R1.fastq SP_R2.fastq
```

3. List the content of file `fastq.tar`.

```
$ tar -tf fastq.tar
SP_R1.fastq
SP_R2.fastq
```

```
$ tar -tvf fastq.tar
-rw-r--r-- goughes/itap 1612210 2017-02-08 08:52 SP_R1.fastq
-rw-r--r-- goughes/itap 1612210 2017-02-08 08:52 SP_R2.fastq
```


Compressing and Archiving

Exercises

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

gzip
tar
zip
Exercises

Redirects,
Pipelines and
Scripts

Change directory:

```
$ cd ~/unix101/
```

1. Use one step to archive and compress the BACKUP directory and name it backup.tar.gz.

```
$ tar -czf backup.tar.gz BACKUP
```

2. Make a directory named test_tar.

```
$ mkdir test_tar
```

3. Copy backup.tar.gz to the directory test_tar and use two-step method to extract its content.

```
$ cp backup.tar.gz test_tar  
$ cd test_tar  
$ gzip -d backup.tar.gz  
$ tar -xf backup.tar
```

Redirects, Pipelines and Scripts

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

**Redirects,
Pipelines and
Scripts**

Redirects

Pipelines

Scripts

Redirects, Pipelines and Scripts

- Redirects
- Pipelines
- Scripts

Redirects, Pipelines and Scripts

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Redirects
Pipelines
Scripts

"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."

– Doug McIlroy, Bell Labs

Small versatile modular software tools that can be assembled into complex workflow pipelines.



<http://www.johnlewis.com/paul-lamond-games-pipeline-game/p62019>

Redirects, Pipelines and Scripts

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Redirects
Pipelines
Scripts

To demonstrate pipelines we need to quickly introduce a couple of new commands. Most of the commands we have talked about so far are not frequently used in pipelines - we're just now getting to the fun stuff! We'll go into these in much more detail in the next workshop.

- `sort` - sorts lines of a text file
- `wc` - count characters, words, and lines of a text file
- `tr` - translate characters into something else

Redirects, Pipelines and Scripts

Redirects

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Redirects
Pipelines
Scripts

With every UNIX program three standard *streams* are created

- Standard output (stdout):
Normal output, printed to your screen
- Standard error (stderr):
Error messages, printed to your screen
- Standard input (stdin):
File for command to read in as input

Change directories:

```
$ cd ~/unix101/redirects/
```

Redirects, Pipelines and Scripts

Redirects

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Redirects
Pipelines
Scripts

Using redirects and pipelines, we can redirect these streams elsewhere such as to a file or another command.

Why?

- Your code or program spams your screen with a ton of text and output. Rather than scrolling your screen for hours, we can send output to a file. With the output in a file, we can use one of the tools (or many others) we have talked about so far to search for interesting lines.
- Send output of one command to another one for further processing or refinement.

Redirects, Pipelines and Scripts

Redirects

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Redirects
Pipelines
Scripts

Change output:

COMMAND > FILE

Take output of a command and put it into FILE, rather than print it on your screen. **This overwrites FILE if it is already present, so be careful!**

Example:

```
$ ls -l > out.log
$ cat out.log
total 0
-rw-r--r-- 1 ddietz rcacsupp 16 Jan 24 13:07 file1.txt
```

Redirects, Pipelines and Scripts

Redirects

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Redirects
Pipelines
Scripts

Change input:

`COMMAND < FILE`

Take contents of `FILE` and feed it into a command. Some commands, such as `tr`, cannot take a file name (like the commands we have seen so far) as an argument so you must feed it in by changing its standard input.

Example:

```
$ cat file1.txt
This is a file.
$ tr i u < file1.txt
Thus us a fule.
```


Redirects, Pipelines and Scripts

Redirects

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Redirects

Pipelines
Scripts

Change input and output:

```
COMMAND < FILE > OTHERFILE
```

Example:

```
$ tr i u < file1.txt > out.log  
$ cat out.log  
Thus us a fule.
```

Redirects, Pipelines and Scripts

Pipelines

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Redirects
Pipelines
Scripts

Most commands on a UNIX system operate in very similar fashions.

- Common behavior pattern:
 - Take input (from file(s) or another program)
 - Selectively do something to certain lines or sections of the data
 - Spit out processed output
- Most commonly operate on text-based files.
- `grep`, `awk`, `sed`, `tr`, `cut`, `paste`, `join`, `sort`, `uniq`
- Always more than one way to solve a problem

Redirects, Pipelines and Scripts

Pipelines

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

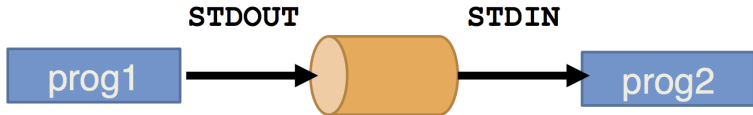
File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Redirects
Pipelines
Scripts

We can pass the output of one command into another command by using *pipes* (`|`) - the so called *pipelines*. These allow a command to read the output of a previous command and do additional processing or manipulation. This output could then be passed onto more commands, and so on, where each step performs a small operation on the way to your end goal.



Redirects, Pipelines and Scripts

Pipelines

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Redirects
Pipelines
Scripts

This is a very basic example. Here we list files in a directory and use `wc` to count them:

```
$ cd ~/unix101/  
$ ls basic_commands | wc  
      6      6     80 # 6 lines/files and 80 characters!
```

Here we are instructing the system to take the output of `ls` and feed it into `wc` to count the lines. This tells us there are 6 files!

Redirects, Pipelines and Scripts

Pipelines

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Redirects
Pipelines
Scripts

Quick review! First get to the Shakespeare directory:

```
$ cd ~/unix101/Shakespeare
```

Find all lines that contain Rosencrantz in Hamlet:

```
$ grep "Rosencrantz" Hamlet.txt
```

Now save those lines to a file:

```
$ grep "Rosencrantz" Hamlet.txt > r-lines
```

Redirects, Pipelines and Scripts

Pipelines

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Redirects
Pipelines
Scripts

Exercise:

- Find lines in `Hamlet.txt` containing `Rosencrantz`
- Sort the lines

One solution:

```
$ grep "Rosencrantz" Hamlet.txt > tempfile.txt  
$ sort tempfile.txt  
$ rm tempfile.txt
```

Redirects, Pipelines and Scripts

Pipelines

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Redirects
Pipelines
Scripts

Exercise:

- Find lines in `Hamlet.txt` containing `Rosencrantz`
- Sort the lines

One solution:

```
$ grep "Rosencrantz" Hamlet.txt > tempfile.txt  
$ sort tempfile.txt  
$ rm tempfile.txt
```

Now do this in one step:

```
$ grep "Rosencrantz" Hamlet.txt | sort
```

Redirects, Pipelines and Scripts

Pipelines

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Redirects
Pipelines
Scripts

Multiple commands in a pipeline:

```
$ grep "Rosencrantz" Hamlet.txt | grep "Polonius" | wc
```

Compare to multiple awkward steps:

```
$ grep "Rosencrantz" Hamlet.txt > step1.txt  
$ grep "Polonius" step1.txt > step2.txt  
$ wc step2.txt  
$ rm step1.txt  
$ rm step2.txt
```


Redirects, Pipelines and Scripts

Pipelines

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Redirects
Pipelines
Scripts

Exercise:

- Find lines in `Hamlet.txt` containing both the word Hamlet and the word King
- Count them
- Sort them
- Save the sorted lines into a file

Redirects, Pipelines and Scripts

Pipelines

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Redirects
Pipelines
Scripts

Exercise:

- Find lines in Hamlet.txt containing both the word Hamlet and the word
- Count them
- Sort them
- Save the sorted lines into a file

Answer:

```
$ grep "Hamlet" Hamlet.txt | grep "King" | wc
      4      31     204  # first number is number of lines
$ grep "Hamlet" Hamlet.txt | grep "King" | sort > kinghamlet.txt
```

Redirects, Pipelines and Scripts

Scripts

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Redirects
Pipelines
Scripts

A *shell script* is a collection of command lines in a file that can all be executed in sequence by giving the name of the file to the bash interpreter.

The importance of creating a script is that the script defines a pattern of actions. The same pattern can be applied to different inputs.

A shell script is just like the script to the play *Hamlet*. Just like the director uses the playscript to instruct each actor their lines to say, our shell script instructs the system (director) to run commands (actors) with different inputs and arguments (lines of dialog).

Redirects, Pipelines and Scripts

Scripts

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Redirects
Pipelines
Scripts

Simple example script:

```
#!/bin/bash

echo "Hithere"
echo "files in this directory are:"

ls -l
```

Redirects, Pipelines and Scripts

Scripts

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Redirects
Pipelines
Scripts

Variables are placeholders. They give a constant, known name to values that might change.

Variables are CREATED with an attached =

```
$ GREETING="Hithere"    # NO SPACE AROUND THE =
```

Variables are USED with a prefixed \$

```
$ echo $GREETING  
Hithere
```

Redirects, Pipelines and Scripts

Scripts

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Redirects
Pipelines
Scripts

Simple variable substitution:

```
#!/bin/bash
# the # means this is a comment and won't
# be executed.
# the USER variable is automatically defined
# (and so are some others)

GREETING="Hithere $USER"
TEXT="files in this directory are:"

echo $GREETING
echo $TEXT
ls -l
```

Redirects, Pipelines and Scripts

Scripts

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Redirects
Pipelines
Scripts

Your shell defines a number of variables for you that you can use as shortcuts. These variables can be used on the command prompt or in a shell script. Good practice dictates using these variables where possible to make your script as flexible and portable as possible.

For example, `$HOME` is defined for you containing the path to your home directory:

```
$ cd $HOME  
$ pwd  
/home/myusername
```

Your scripts are more powerful if they use variables such as `$HOME` instead of hard-coding your home directory as you could pass your script to a labmate and he or she would not have to go through and substitute their own username.

Redirects, Pipelines and Scripts

Scripts

Acknowledgments

Logging In

Files and
Directories

First
Command

Basic
Commands

File
Permissions

Compressing
and Archiving

Redirects,
Pipelines and
Scripts

Redirects
Pipelines
Scripts

At their very simplest, shell scripts are a list of commands you want executed because we're lazy and don't want to type the commands over and over. Shell scripts can get quite complex with logic, flow control, loops, and most things a programming language can do. We'll talk more in-depth about these things in the next workshop.