Using the Linux Filesystem

- The Hierarchical Filesystem
- Unix Files and Directories
- Filenames
- Case Sensitivity
- Filename Extensions
- Current Directory
- Your Home Directory
- Navigating the Hierarchical File Systems

Class Exercise Scripts

- What I am going to suggest here is, essentially, a shortcut, just to save some typing.
- First, run the **`history`** command:
  - It will list the last several hundred commands you entered
  - You should follow this with **`| less`** because it can be very long!
  - Look for the number representing where you first started
  - Determine number of lines from there to the bottom of your history.
- Then, you can type this command:

```
history | tail -45 | awk '{$1=""; print $0}' >> ex8.sh
```

- Replace **45** with the actual number of lines and **8** with the actual exercise number!

Class Exercise Scripts

- This will write to your file those lines from your history, but without the line numbers at the beginning.
- **NOTES:**
  - You will ***still*** need to open that new file up in **`nano`** and edit it for both correctness and neatness.
  - Again, this just saves you typing; it doesn't do *all* your work for you.
  - At this stage, you probably will not know how the awk command is working, which is okay. It will be addressed later.
  - Finally, the instructions assume that you are currently located in the directory where the file is supposed to go. If not, then you may need to adjust accordingly.

The Hierarchical Filesystem

- A **filesystem** is:
  - an arrangement of data on a storage device and...
  - the software used to store and retrieve the data
- All files are contained in directories
- On Windows and the Mac these directories are called folders
- A directory can contain other directories
- A **hierarchical filesystem** starts with a special directory, located at the top
- This directory is called the **root directory**

The Hierarchical Filesystem

- In Unix, we represent this directory with a single slash character, /

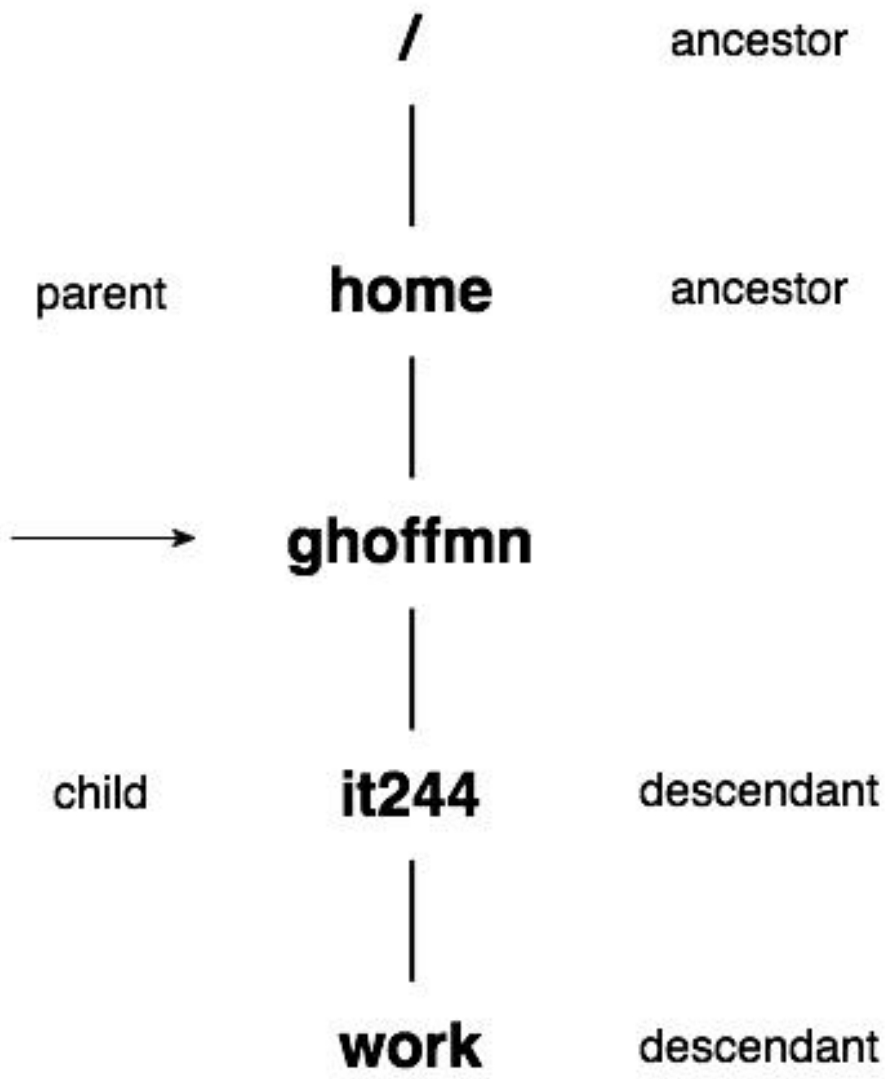
```
$ ls /
bin       initrd.img  proc tmp
boot      lib         root        tools
courses   lib32       run         users
data      lib64       sbin        usr
dev       lost+found  selinux     var
etc       media       sources     vmlinuz
groups    mnt         spool
home      nobackup    srv
home.ORIG  opt        sys
```

The Hierarchical Filesystem

- **All** other directories are contained within the root directory or one of its many subdirectories
- This structure is called a ***tree*** because it looks like a tree turned upside down
- Unlike in nature, this tree grows down from the root
- A hierarchical filesystem is endlessly extensible
- We can have as many files and directories as we want -- as long as there is enough disk space to hold them

The Hierarchical Filesystem

- A hierarchical filesystem resembles a family tree
- We often use "family" terms when talking about directories
  - The directory one level up is called the **parent directory**
  - Any directory above another directory can be called an **ancestor**
  - Directories inside the directory are its **child directories**
  - Any file or directory below the current directory is a **descendant**
  - Directories and files within the same directory are called **siblings**


/
ancestor
parent
home
ancestor
ghoffmn
child
it244
descendant
work
descendant

Unix Files and Directories

- **Files** are linear arrangements of data on some storage device:
  - Some files are programs
  - Program files contain binary instructions that the machine understands
  - Some files contain data in various formats such as text
  - Some Unix text files contain configuration information:
    - These configuration files modify your Unix environment
    - They perform the same function as the Windows Registry

Unix Files and Directories

- Since these files are text you can read and edit them easily
- You only need a text editor to change them
- Unix treats directories as files too although they are a special kind of file:
  - It also treats devices as files
  - But this is invisible to the ordinary user
  - You can't run *cat* or *less* on a directory or a printer
- Treating directories and devices as files allows Unix to keep things simple
- Directories and devices are just special kinds of files

Filenames

- When you ask Unix for a file you must give it two pieces of information
  - The name of the file
  - The location of the file in the hierarchical file system
- Every file has a name - the filename
- Different Unix flavors allow filenames to be of different length
- Most Unix systems allow filenames to be no more than 255 characters

Filenames

- It's best to keep filenames short
- This makes them easier to type
- You're less likely run into trouble if you move the file to another system where the maximum filename length may be shorter
- To learn the maximum length of a filename on any machine use *stat*

```
$ stat -f /home | grep Namelen
ID: 0          Namelen: 255      Type: autofs
```

Filenames

- Notice that I used a pipe to get just the information I wanted
- You can use a space in filenames if you enclose them in quotes
- This is a **VERY BAD IDEA**
- When you run */s* on a directory that has a filename with a space in it, it looks like you have more than one file

```
$ touch 'foo bar'
$ ls
foo bar
```

- It looks like you have two files, **foo** and **bar** – whereas, in reality, you only have one file, **foo bar**

Filenames

- If you encounter such a file, then you have to escape the space character to work with the file

```
$ ls -l foo\ bar
-rw-r--r-- 1 it244gh it244-2G 0 2011-09-24 14:34 foo bar
```

- Instead of a space in a filename, use an underscore, _

```
$ touch foo_bar
$ ls
foo_bar
```

Filenames

- To avoid problems, only use the following characters in file names
  - Uppercase letters (A-Z)
  - Lowercase letters (a-z)
  - Digits (0-9)
  - Underscore _
  - Dash -
  - Period .
- You cannot have two files with the same name in the same directory

Case Sensitivity

- Unix is **case sensitive**
- This means that **Foo**, **foo**, and **FOO** are three different files as far as Unix is concerned

```
$ ls

$ touch Foo foo FOO

$ ls
foo  Foo  FOO
```

Case Sensitivity

- Unix commands are always lower case
  - This is just a convention
  - But it is a helpful convention
  - Otherwise you would have to remember capitalization as well as the name of the command
- Some operating systems are **not** case sensitive like DOS and Windows
- Stick to the convention of always using lowercase filenames this will make it easier to move between different operating systems

Filename Extensions

- Extensions are strings of characters that appear at the end of the filename after a period
- Unix ignores extensions
- As far as Unix is concerned, extensions are just part of the filename
- Some Unix programs, **do** require certain extensions
  - The C compiler, ***gcc***, expects the filenames of source code files to end in `.c`
  - The Java compiler, ***javac***, expects Java source files to have `.java` extension
  - The compilers themselves, enforce these requirements
- Unix couldn't care less

Current Directory

- The way a Unix command works depends on your environment
- One important part of your environment is your current directory
- Think of your **current directory** as your current location in the filesystem
  - The ***pwd*** (print working directory) command will always tell you your current directory
  - If you use a command that expects a directory as an argument, you can usually leave out the directory name and the command will assume your current directory
  - For example, ***ls*** used with no argument will list the contents of your current directory

```
$ ls
assignments_submitted  it244   nsmail
bin                    it341   public_html
course_files           it441   test
examples               mail    test_taken
html                   Mail    vp
it114                  News
```

Your Home Directory

- Whenever you log in to a Unix host you should always find yourself inside your **home directory**
  - This directory belongs to your Unix account only
  - You have full privileges in your home directory
  - You can create files or directories in your home directory
  - You can let other users have access to the files and directories in your home directory
- To get back to your home directory use ***cd*** (**c**hange **d**irectory) with no arguments

```
$ cd

$ pwd
/home/ghoffmn
```

Your Home Directory

- Your home directory contains a number of "hidden" files which customize your environment
- These files begin with a period and do not ordinarily appear in an ***ls*** listing -- unless you run ***ls*** with the **-a** option
- On most Unix systems, your home directory will be in the **/home** directory
- On a Mac it will appear in the **/Users** directory

Navigating the Hierarchical File Systems

- Any file or directory in the hierarchical filesystem will be one of four positions relative to your current directory
  - *Inside* your current directory
  - *Below* your current directory
  - *Above* your current directory
  - *Off to the side* of your current directory
- In this last case, you must go up before you can go down to reach the file or directory