Discussion Session
Week 1

Command Line

Command Line

- Different Languages
  - Windows uses one set of commands
  - Linux uses another
- “Shell Scripting” or Bash
- “dir” and “ls -l”
- “copy” and “cp”
- “del” and “rm”
- etc...

Standard

- Vast majority of software developers and companies will use Linux or some sort of linux system
- Get used to using bash
- There are tons of commands, and you will not always remember them all, but there are several key commands that you should remember (more on this later)

Before we get into bash, some background

- When we work with bash, we need to be familiar with the structure of Linux systems, and there are multiple keywords that are associated
- The faster you learn this jargon, the better off you are

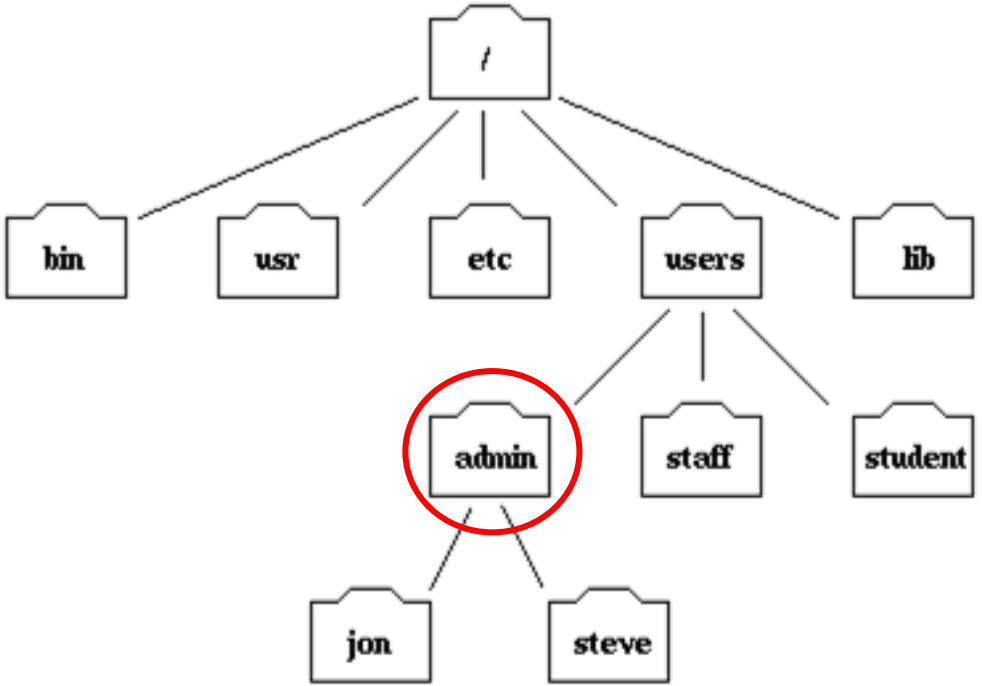
Linux File System

- The Linux file system can be viewed as a **tree** like structure
- The system is made of directories, subdirectories, and files
- For the purposes of this class, almost all work is done in the path **~/**
- **~** is the **home directory**

File System Overview

Current Working Directory

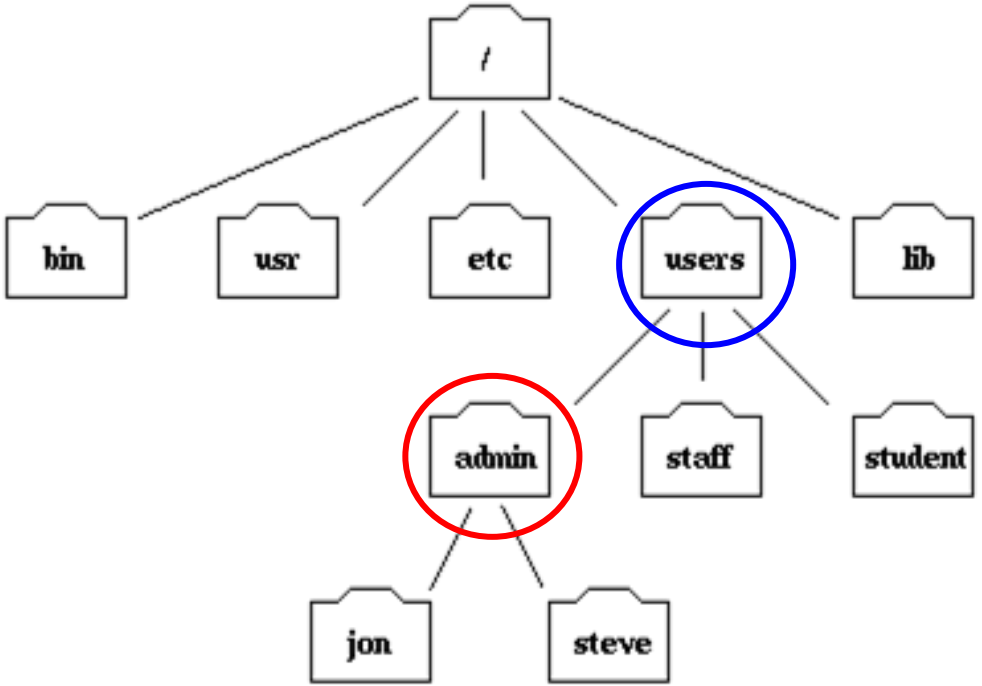
Denoted as “.”



File System Overview

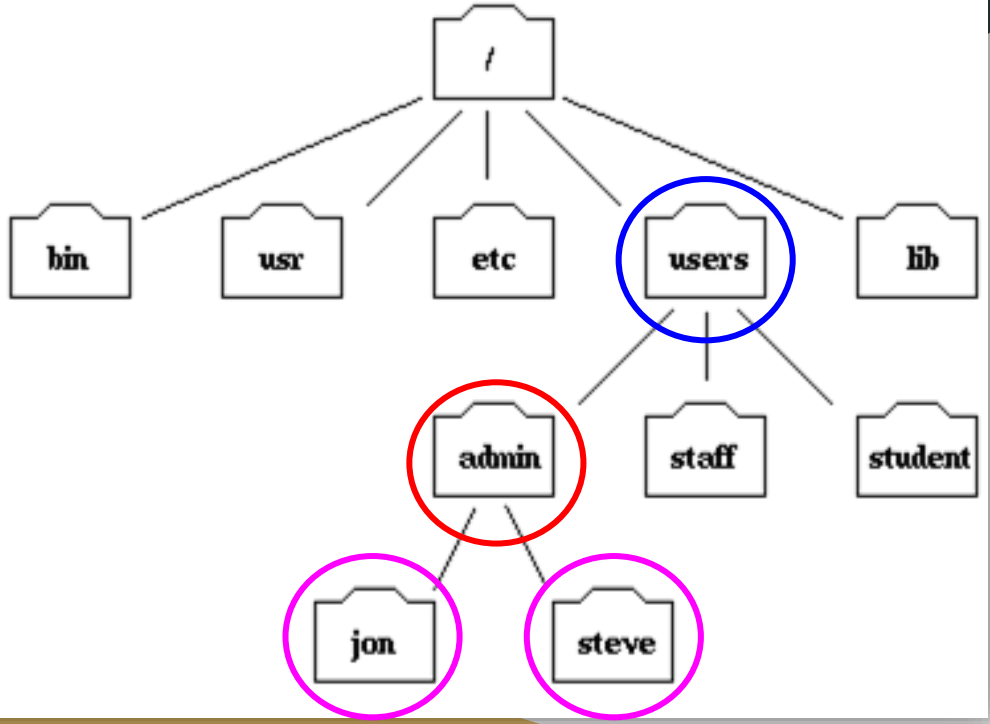
"Parent Directory"

Denoted as ".."



File System Overview

"Child Directory/Subdirectory"



Other Jargon

- Commands
  - What you write on the command line to perform actions
- Options
  - Add-ons to commands that change behavior of commands
- Operators
  - Symbols such as +, -, >>, <<, |, that perform specific actions

Commands/Operators

- man
- echo
- ls
- pwd
- cd
- cat (and other readers)
- mkdir
- rm, cp, mv
- touch
- grep
- | (Pronounced “Pipe”)
- && (Pronounced “And”)
- || (Pronounced “Or”)
- >> (Pronounced “Redirect”)

man

Displays the manual page for a given command

Usage: man {command}

```
derek@DESKTOP-3L8T6AU:/mnt/c/Users/Derek Jacobs$ man man
```

```
MAN(1)                                    Manual pager utils                                    MAN

NAME
       man - an interface to the on-line reference manuals

SYNOPSIS
       man [-C file] [-d] [-D] [--warnings[=warnings]] [-R encoding] [-L locale] [-m system[,...]] [-M path] [-S list] [-e extension] [-i|-I] [--regex|--wildcard] [--names-only] [
       [-u] [--no-subpages] [-P pager] [-r prompt] [-7] [-E encoding] [--no-hyphenation] [--no-justification] [-p string] [-t] [-T[device]] [-H[browser]] [-X[dpi]] [-Z] [[secti
       page[.section] ...] ...
       man -k [apropos options] regexp ...
       man -K [-w|-W] [-S list] [-i|-I] [--regex] [section] term ...
       man -f [whatis options] page ...
       man -l [-C file] [-d] [-D] [--warnings[=warnings]] [-R encoding] [-L locale] [-P pager] [-r prompt] [-7] [-E encoding] [-p string] [-t] [-T[device]] [-H[browser]] [-X[dp
       [-Z] file ...
       man -w|-W [-C file] [-d] [-D] page ...
       man -c [-C file] [-d] [-D] page ...
       man [-?V]

DESCRIPTION
       man is the system's manual pager.  Each page argument given to man is normally the name of a program, utility or function.  The manual page associated with each of these ar
       ments  is  then found and displayed.  A section, if provided, will direct man to look only in that section of the manual.  The default action is to search in all of the ava
       able sections following a pre-defined order ("1 n l 8 3 2 3posix 3pm 3perl 3am 5 4 9 6 7" by default, unless overridden by the SECTION directive in  /etc/manpath.config),
       to show only the first page found, even if page exists in several sections.
```

When in doubt, look it up

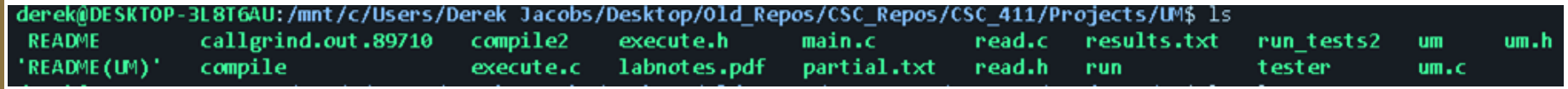
echo

Prints out a variable/string

```
derek@DESKTOP-3L8T6AU:/mnt/c/Users/Derek Jacobs$ temp="Hello World"
derek@DESKTOP-3L8T6AU:/mnt/c/Users/Derek Jacobs$ echo $temp
Hello World
derek@DESKTOP-3L8T6AU:/mnt/c/Users/Derek Jacobs$ echo Hello World
Hello World
derek@DESKTOP-3L8T6AU:/mnt/c/Users/Derek Jacobs$ echo "My name is Derek"
My name is Derek
derek@DESKTOP-3L8T6AU:/mnt/c/Users/Derek Jacobs$ echo -e "$temp\nMy name is Derek"
Hello World
My name is Derek
derek@DESKTOP-3L8T6AU:/mnt/c/Users/Derek Jacobs$
```

ls

Used to list files and subdirectories in the current working directory



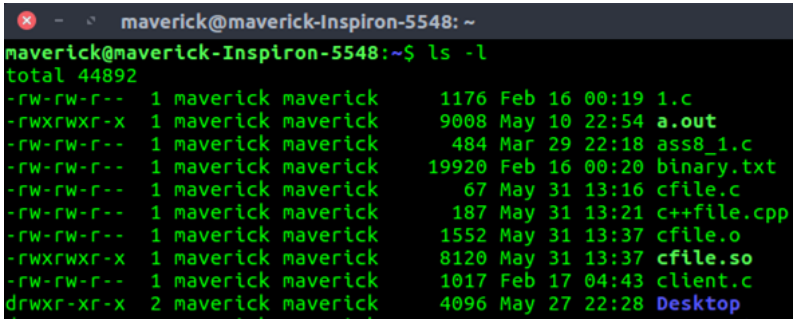
Useful options

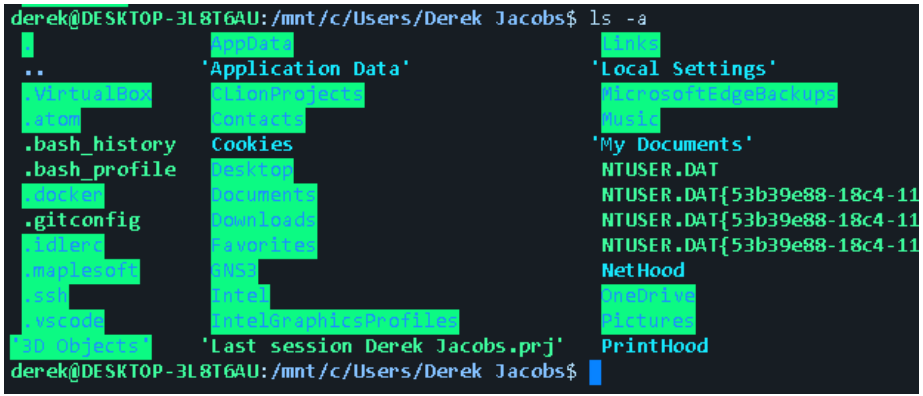
-l

Lists in “long format”

-a

Lists all files, even hidden ones





pwd/cd

pwd:

Prints the current working directory

cd:

Used to change the current working directory

Can use either a relative path or an absolute path

Relative: In relation to the current working directory

Absolute: Containing full path from home directory to target directory

cd Examples

```
derek@DESKTOP-3L8T6AU:/mnt/c/Users/Derek Jacobs/Desktop/CSC$ ls
461  544  550  TA
derek@DESKTOP-3L8T6AU:/mnt/c/Users/Derek Jacobs/Desktop/CSC$ cd 550/Programming_Assignments/
derek@DESKTOP-3L8T6AU:/mnt/c/Users/Derek Jacobs/Desktop/CSC/550/Programming_Assignments$ cd ../../461/Projects/
derek@DESKTOP-3L8T6AU:/mnt/c/Users/Derek Jacobs/Desktop/CSC/461/Projects$ cd /mnt/c/Users/Derek\ Jacobs/Desktop/CSC/544/Notes/
derek@DESKTOP-3L8T6AU:/mnt/c/Users/Derek Jacobs/Desktop/CSC/544/Notes$
```

Exercise 1 (5 Min)

Provide a sequence of commands to

a) Print your current working directory
b) Print all files (including hidden ones) of your current working directory in long, human readable format
    i) Hint: Use ‘man’
c) Change directory to a directory of your choice
d) Change back to your original path using a relative path

File Readers

cat, more, less

Used to print out the contents of files

Difference is how it’s printed out

File Readers example

```
derek@DESKTOP-3L8T6AU:/mnt/c/Users/Derek Jacobs/Desktop/Old_Repos/CSC_Repos/CSC_411/Projects/Arith$ cat bitpack.c
#include <bitpack.h>
#include <math.h>
#include <stdio.h>
#include <stdlib.h>

#include "assert.h"
#include "except.h"
Except_T Bitpack_Overflow = { "Overflow packing bits" };


static inline uint64_t shift_leftu(uint64_t value, uint64_t shift) {
  if(shift == 64) {
    value = 0-1;
  }
  else {
    value <<= shift;
  }
  return value;
}
```

rm,cp,mv

mv:

Used to move files or rename them

mv ./file1 ../file1

cp:

Used to copy files or directories

rm:

Used to delete existing files or directories

Useful Options

-r Recursively delete contents of subdirectories

-f Force deletion

mkdir

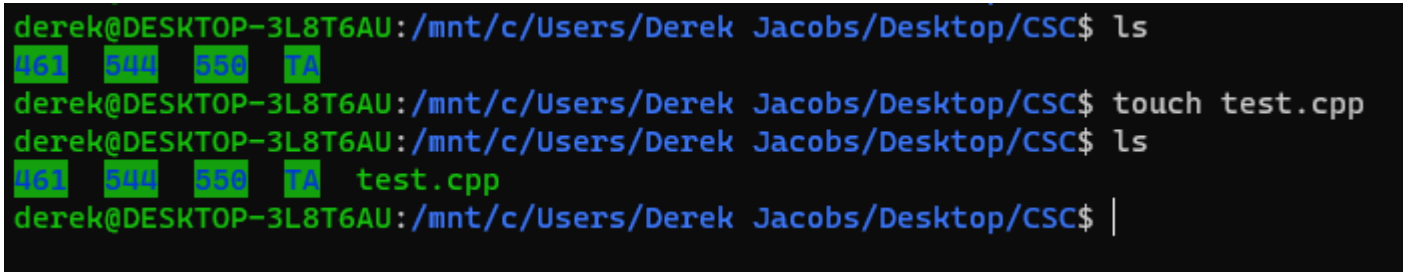
Creates a new directory

```
derek@DESKTOP-3L8T6AU:/mnt/c/Users/Derek Jacobs/Desktop/CSC/544$ ls
Notes
derek@DESKTOP-3L8T6AU:/mnt/c/Users/Derek Jacobs/Desktop/CSC/544$ mkdir temp
derek@DESKTOP-3L8T6AU:/mnt/c/Users/Derek Jacobs/Desktop/CSC/544$ ls
Notes  temp
derek@DESKTOP-3L8T6AU:/mnt/c/Users/Derek Jacobs/Desktop/CSC/544$ rm -rf temp
derek@DESKTOP-3L8T6AU:/mnt/c/Users/Derek Jacobs/Desktop/CSC/544$ ls
Notes
derek@DESKTOP-3L8T6AU:/mnt/c/Users/Derek Jacobs/Desktop/CSC/544$
```

```
derek@DESKTOP-3L8T6AU:/mnt/c/Users/Derek Jacobs/Desktop/CSC$ ls
461  544  550  TA  test.txt
derek@DESKTOP-3L8T6AU:/mnt/c/Users/Derek Jacobs/Desktop/CSC$ echo "This is a test file" >> test.txt
derek@DESKTOP-3L8T6AU:/mnt/c/Users/Derek Jacobs/Desktop/CSC$ cat test.txt
This is a test file
derek@DESKTOP-3L8T6AU:/mnt/c/Users/Derek Jacobs/Desktop/CSC$ cp test.txt ./TA/testCopy.txt
derek@DESKTOP-3L8T6AU:/mnt/c/Users/Derek Jacobs/Desktop/CSC$ cat ./TA/testCopy.txt
This is a test file
derek@DESKTOP-3L8T6AU:/mnt/c/Users/Derek Jacobs/Desktop/CSC$
```

touch

Used to create files



grep

Used to search for a phrase or word

Usage: grep {searchTerm} searchFile/Directory

```
derek@DESKTOP-3L8T6AU:/mnt/c/Users/Derek Jacobs/Desktop/CSC/550/Notes$ grep Divi *
Big_Number_Arithmetic.txt:        Division of a two place int by a one place int, provided the quotient is a one plac
nt,
Big_Number_Arithmetic.txt:Division
Big_Number_Arithmetic.txt:        Dividend u: m+n digits
Big_Number_Arithmetic.txt:        Divisor v: n digits
Maple_Intro.txt:    Greatest Common Divisor
derek@DESKTOP-3L8T6AU:/mnt/c/Users/Derek Jacobs/Desktop/CSC/550/Notes$
```

Exercise 2 (10 Min)

Provide a sequence of commands to

a) Create a directory called “Exercise_2” and cd into that directory
b) Create a file called “bashIntro.txt”
    i) Add the following string to the file
        1) “I am learning bash!”
c) Output the contents of bashIntro.txt
d) Make 3 copies of bashIntro.txt, named “copy1.txt”, “copy2.txt”, and “copy3.txt”
e) Output a list of files containing the string “I am learning bash!”
    i) You’ll need to use man again

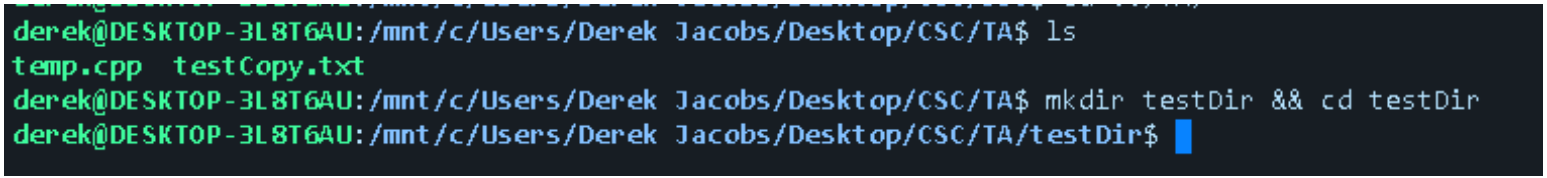
| (Pipe)

Used to redirect output of one command to the input of another

```
derek@DESKTOP-3L8T6AU:/mnt/c/Users/Derek Jacobs/Desktop/CSC/461/Notes$ cat ML_Background.txt | grep -i supervised
    Machine Learning (Supervised)
        "Supervised"...when its working, it uses info from the input and output
        1) Supervised Learning
        2) Unsupervised Learning
```

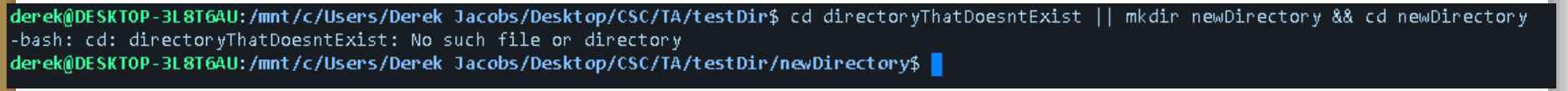
&& (And)

Used to execute commands sequentially (iff the left hand side succeeds)



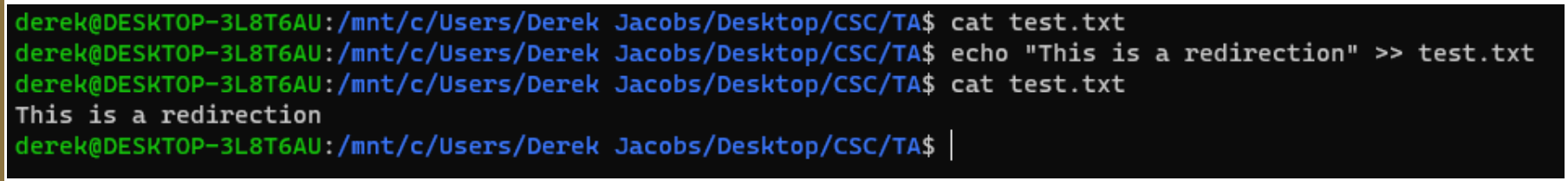
|| (Or)

Used to complete commands sequentially regardless of success status



>>, << (Redirect)

Used for other manipulation of command outputs



Scripting

Scripts

Sequences of commands that are executed from start to finish

Commands may fail, but the script will not stop

Running a script:

bash {scriptName}

Sample script

```
#!/bin/sh

#Compile the files
./compile2

#Remove any callgrind.out files
rm callgrind.out.*

echo "RUNNING WITH -O2"
#Run the um on each input, and time it
echo "Running Callgrind..."
valgrind --tool=callgrind -q ./um /csc/411/um/midmark.um > /dev/null
temp=`cat callgrind.out.* | grep totals:`
echo "Total Instructions = " ${temp##*totals:} >> results.txt

echo "Timing midmark..."
#Time midmark
time -o ./results.txt -a -f "Midmark time: %E" ./um /csc/411/um/midmark.um > /dev/null
echo "Timing sandmark..."
#Time sandmark
time -o ./results.txt -a -f "Sandmark time: %E" ./um /csc/411/um/sandmark.umz > /dev/null
echo "Timing advent..."
#Time advent partial solution
cat ./partial.txt | time -o ./results.txt -a -f "Advent time: %E" ./um /csc/411/um/advent.umz > /dev/null
```