We are 183

Linux & Command Line

and

Life after 183

L23: Monday – Week 14

Reminders

- Grading of core should be back to you soon
- Monday 4/18
  - Reach is Due
- Thursday 4/21
  - Showcase
  - Companies will be there to talk to you

Reminders

- Remember to fill out course evaluations, forward confirmation email to eecs-evals@umich.edu
  - See directions in email announcement

Discussion Sections

- Tuesday
  - Maxim: Creating a Personal Website
  - Leah: Creating the Perfect Resume
  - Grace: Head Start to 280
  - Anna R.: Resume and Technical Interview
- Wednesday
  - Kevin: Interview Tips and Tricks
- Thursday
  - Michael: Getting Started with Web Development
- Friday
  - Erin: Elevator Speeches and Preparing for Career Fairs

UNIX operating system

- **1969: Project Started at AT&T Bell Labs**
  - by Ken Thompson and Dennis Ritchie
- **1971: First Edition of Unix**
- **1972: Version 2**
  - Rewritten in C (made it portable)
- **1973: Version 5**
  - Licensed for the first time to Universities

UNIX System Allowed:

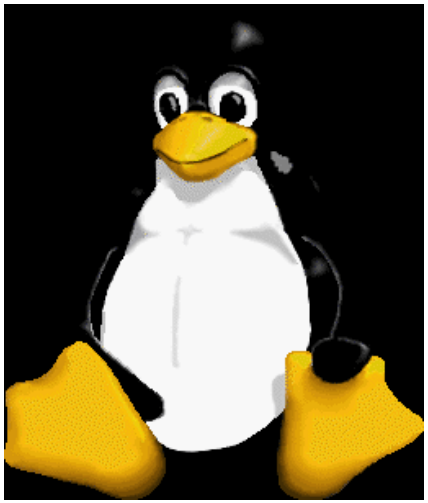
- **Computer aided design**
- **Manufacturing control systems**
- **Laboratory simulations**
- **The Internet**

Without UNIX…

- **The Internet would halt**
- **Most telephone calls could not be made**
- **Electronic commerce would halt**
- **"Jurassic Park" would have never been made**

Linux – A derivative of UNIX

- **Linux is an operating system**
  - 1991: Linus Torvalds created Linux as a hobby
  - Operating system for the autograder
- **Linux Kernel released in 1994**
  - Currently under the GNU General Public License
  - Freely available to everyone
- **Tux is the official mascot**



On Mac OS X

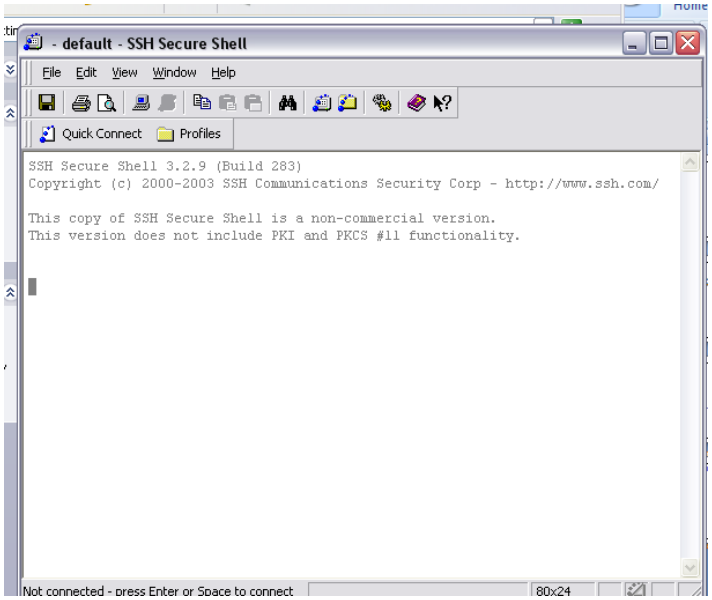
- **Mac OS X is based on UNIX**

- **"Terminal Window" is your friend**
  - Gives you "command line" power
  - Linux/Unix command rule

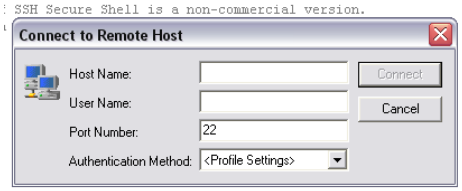
PC Users

- Putty is the recommended command line app
- Download
  - http://www.itd.umich.edu/bluedisc/
    - Pick the version for your operating system

PC: Putty Client



Quick Connect



Host Name: login.itd.umich.edu

User Name: uniqname

Connect

Password: your Kerboros passwd

Mac: terminal

```
username$ ssh uniqname@login.itd.umich.edu
```

Comes back and asks for password:

type your Kerboros passwd

Your first linux command: **man**

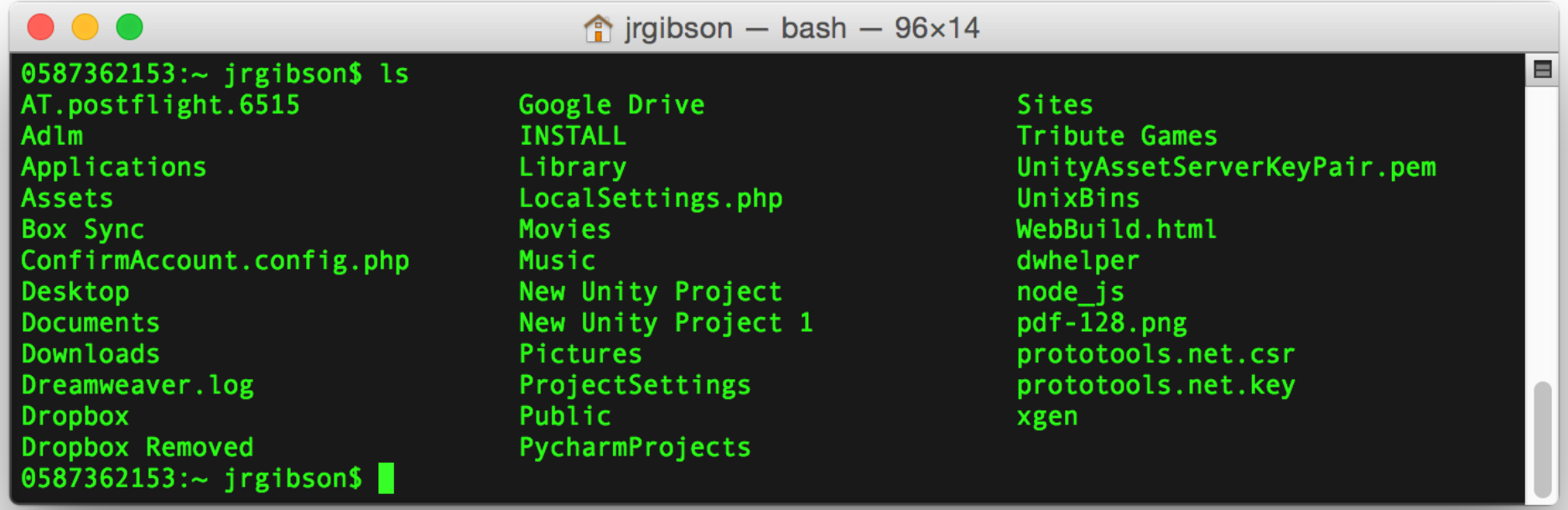
- **man ls** Manual for "ls"

```
LS(1)                    BSD General Commands Manual                    LS(1)

NAME
     ls -- list directory contents

SYNOPSIS
     ls [-ABCFGHLOPRSTUW@abcdefghiklmnopqrstuwx1] [file ...]

DESCRIPTION
     For each operand that names a file of a type other than directory, ls
     displays its name as well as any requested, associated information.  For
     each operand that names a file of type directory, ls displays the names
     of files contained within that directory, as well as any requested,
     associated information.

     If no operands are given, the contents of the current directory are dis-
     played.  If more than one operand is given, non-directory operands are
     displayed first; directory and non-directory operands are sorted sepa-
     rately and in lexicographical order.

     The following options are available:

:
```

List files in directory: **ls**

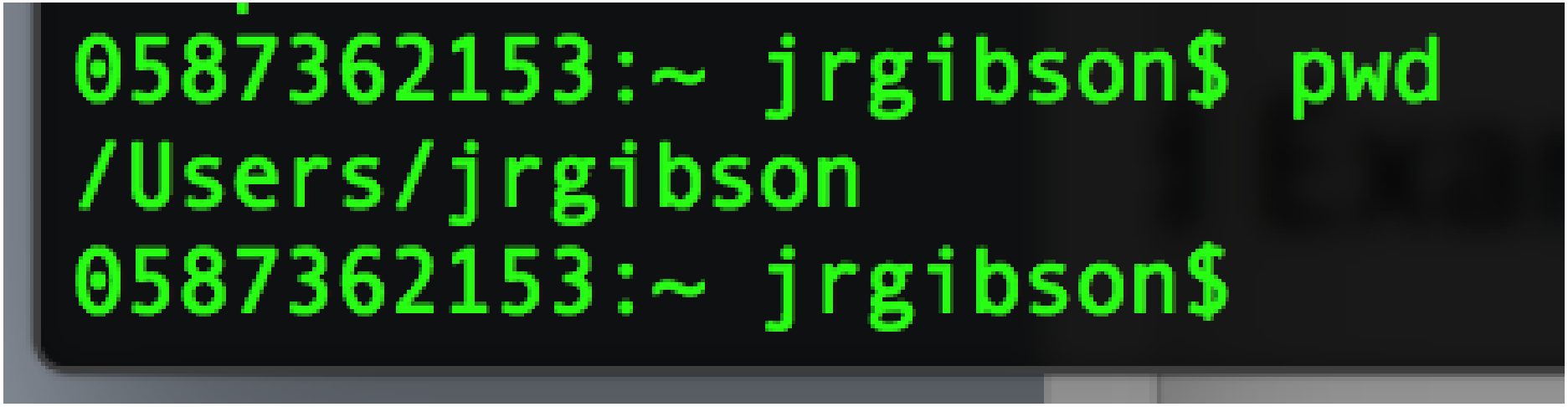
- **ls** list files in current directory

  “dir” in DOS on PC



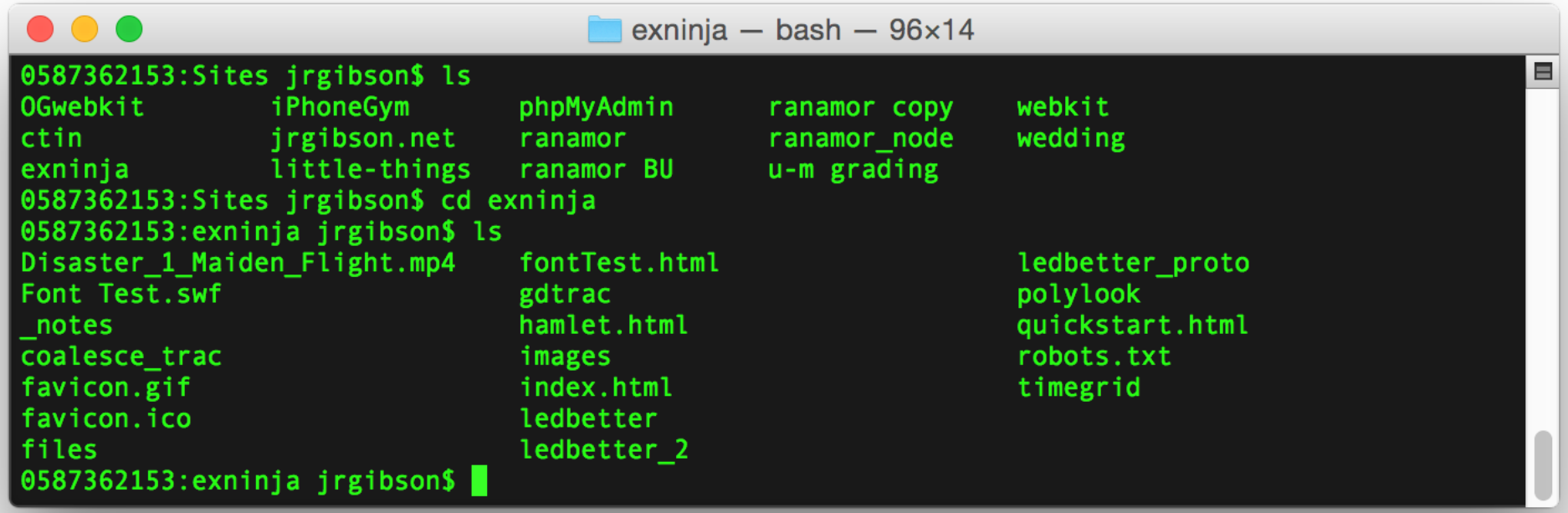
Print Working Directory: **pwd**

- **pwd** present working directory



Change Directory: **cd**

- **cd** xxxx change directory to "xxxx"
  - Example: **cd exninja**

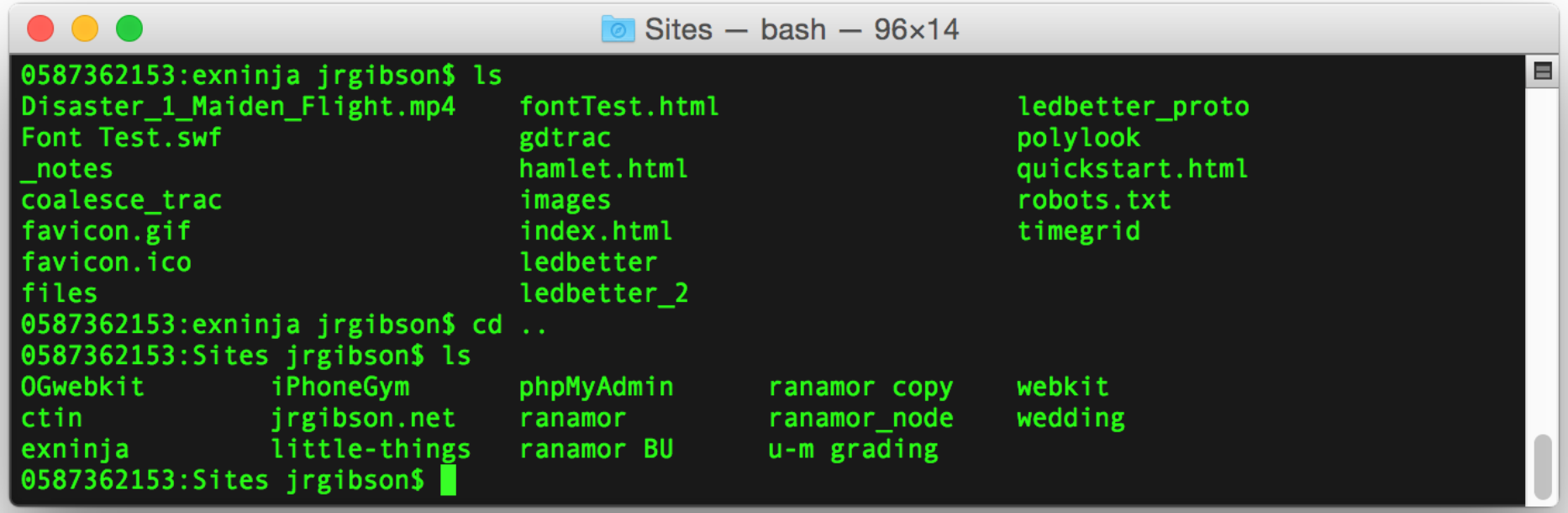


Change Directory (Up): `cd ..`

- **`cd ..`** Goes up one directory
  - Example: **/Sites/exninja/**

    `cd ..`

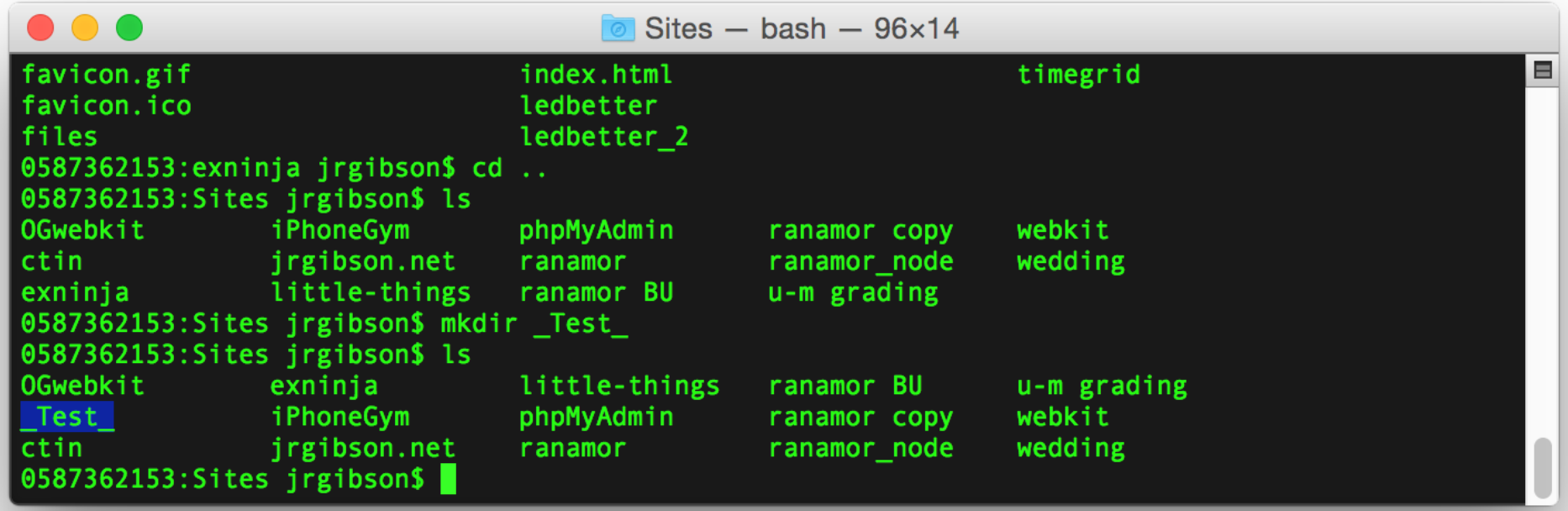
    /Sites/



Make directory: **mkdir**

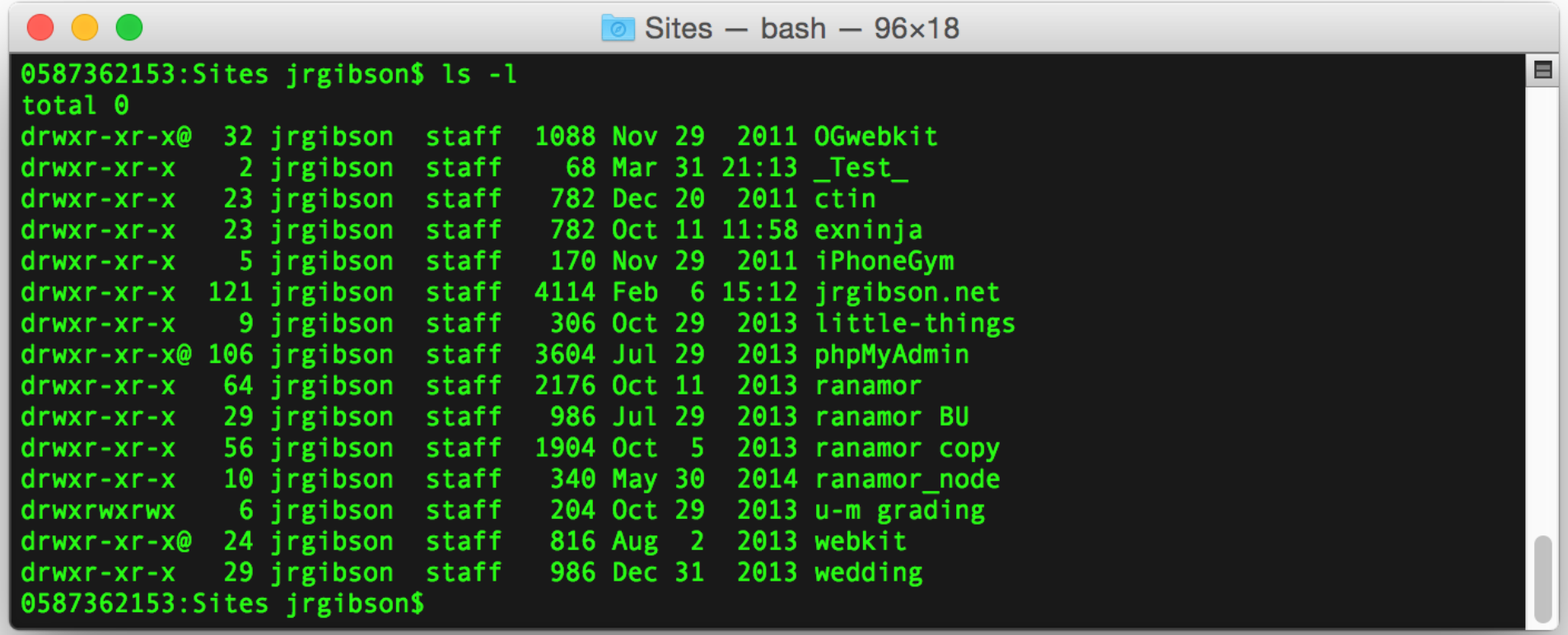
- **mkdir xxxx** make directory "xxxx"
  - Example: **`mkdir _Test_`**

- **Note:** linux does NOT like spaces in names



List files in long form: `ls -l`

- **ls –l** list files in **l**ong form



List files in long form: `ls -l`

% ls -l

total 476

```
-rwxr-xr-x 1 mdorf staff 455203 Nov 19  2011 a.out
drwxr-xr-x 3 mdorf staff   2048 Jan 21  2004 f02_proj1
drwxr-xr-x 2 mdorf staff   2048 Oct 29  2009 proj4
drwxrwxrwx 3 mdorf staff   2048 Nov  6  2010 test
-rw-r--r-- 1 mdorf staff 455203 Nov 19  2012 conn4.exe
```

List files in long form: `ls -l`
1st Character – File Type

% ls -l

total 476

```
-rwxr-xr-x 1 mdorf staff 455203 Nov 19  2011 a.out
drwxr-xr-x 3 mdorf staff   2048 Jan 21  2004 f02_proj1
drwxr-xr-x 2 mdorf staff   2048 Oct 29  2009 proj4
drwxrwxrwx 3 mdorf staff   2048 Nov  6  2010 test
-rw-r--r-- 1 mdorf staff 455203 Nov 19  2012 conn4.exe
```

```
-   normal file
d   directory
s   socket file
l   link file
```

List files in long form: `ls -l`
Permissions

% ls -l

total 476

```
-rwxr-xr-x 1 mdorf staff 455203 Nov 19  2011 a.out
drwxr-xr-x 3 mdorf staff   2048 Jan 21  2004 f02_proj1
drwxr-xr-x 2 mdorf staff   2048 Oct 29  2009 proj4
drwxrwxrwx 3 mdorf staff   2048 Nov  6  2010 test
-rw-r--r-- 1 mdorf staff 455203 Nov 19  2012 conn4.exe
```

Next 9 characters

read/write/execution permissions for

user, group, world

(more later)

List files in long form: `ls -l`
Number of links

% ls -l

total 476

```
-rwxr-xr-x  1 mdorf staff 455203 Nov 19  2011 a.out
drwxr-xr-x  3 mdorf staff   2048 Jan 21  2004 f02_proj1
drwxr-xr-x  2 mdorf staff   2048 Oct 29  2009 proj4
drwxrwxrwx  3 mdorf staff   2048 Nov  6  2010 test
-rw-r--r--  1 mdorf staff 455203 Nov 19  2012 conn4.exe
```

number of links for that file

List files in long form: `ls -l`

Owner

```
% ls -l
total 476
-rwxr-xr-x 1 mdorf staff 455203 Nov 19  2011 a.out
drwxr-xr-x 3 mdorf staff   2048 Jan 21  2004 f02_proj1
drwxr-xr-x 2 mdorf staff   2048 Oct 29  2009 proj4
drwxrwxrwx 3 mdorf staff   2048 Nov  6  2010 test
-rw-r--r-- 1 mdorf staff 455203 Nov 19  2012 conn4.exe
```

lists the "Owner" of the files

List files in long form: `ls -l`

Group

```
% ls -l
total 476
-rwxr-xr-x 1 mdorf staff 455203 Nov 19  2011 a.out
drwxr-xr-x 3 mdorf staff   2048 Jan 21  2004 f02_proj1
drwxr-xr-x 2 mdorf staff   2048 Oct 29  2009 proj4
drwxrwxrwx 3 mdorf staff   2048 Nov  6  2010 test
-rw-r--r-- 1 mdorf staff 455203 Nov 19  2012 conn4.exe
```

lists which "Group" the files are part of

List files in long form: `ls -l`

Size

```
% ls -l
total 476
-rwxr-xr-x 1 mdorf staff 455203 Nov 19  2011 a.out
drwxr-xr-x 3 mdorf staff   2048 Jan 21  2004 f02_proj1
drwxr-xr-x 2 mdorf staff   2048 Oct 29  2009 proj4
drwxrwxrwx 3 mdorf staff   2048 Nov  6  2010 test
-rw-r--r-- 1 mdorf staff 455203 Nov 19  2012 conn4.exe
```

lists the size of the file in bytes

List files in long form: `ls -l`
Last modified date & time

% ls -l

total 476

```
-rwxr-xr-x 1 mdorf staff 455203 Nov 19  2011 a.out
drwxr-xr-x 3 mdorf staff   2048 Jan 21  2004 f02_proj1
drwxr-xr-x 2 mdorf staff   2048 Oct 29  2009 proj4
drwxrwxrwx 3 mdorf staff   2048 Nov  6  2010 test
-rw-r--r-- 1 mdorf staff 455203 Nov 19  2012 conn4.exe
```

lists date file was last "modified" along with the time if it was modified recently

List files in long form: `ls -l`
File name

```
% ls -l
total 476
-rwxr-xr-x 1 mdorf staff 455203 Nov 19  2011 a.out
drwxr-xr-x 3 mdorf staff   2048 Jan 21  2004 f02_proj1
drwxr-xr-x 2 mdorf staff   2048 Oct 29  2009 proj4
drwxrwxrwx 3 mdorf staff   2048 Nov  6  2010 test
-rw-r--r-- 1 mdorf staff 455203 Nov 19  2012 conn4.exe
```

name of the file

Permissions

```
d: directory
r: read
w: write
x: execute
```

```
-rw-r--r-- 1 mdorf staff 455203 Nov 19  2012 conn4.exe
```

Permissions

d: directory
r: read
w: write
x: execute

in groups of threes
owner group others

-rw-r--r-- 1 mdorf staff 455203 Nov 19 2012 conn4.exe

Permissions

d: directory
r: read
w: write
x: execute

in groups of threes
owner group others

-rw-r--r-- 1 mdorf staff 455203 Nov 19 2012 conn4.exe

'owner'
rw-

'group'
r--

'others'
r--

Permissions

d: directory
r: read
w: write
x: execute

in groups of threes
owner group others

-rw-r--r-- 1 mdorf staff 455203 Nov 19 2012 conn4.exe

'owner'
rw-

'group'
r--

'others'
r--

NO ONE can "execute"

Change permissions: **chmod**

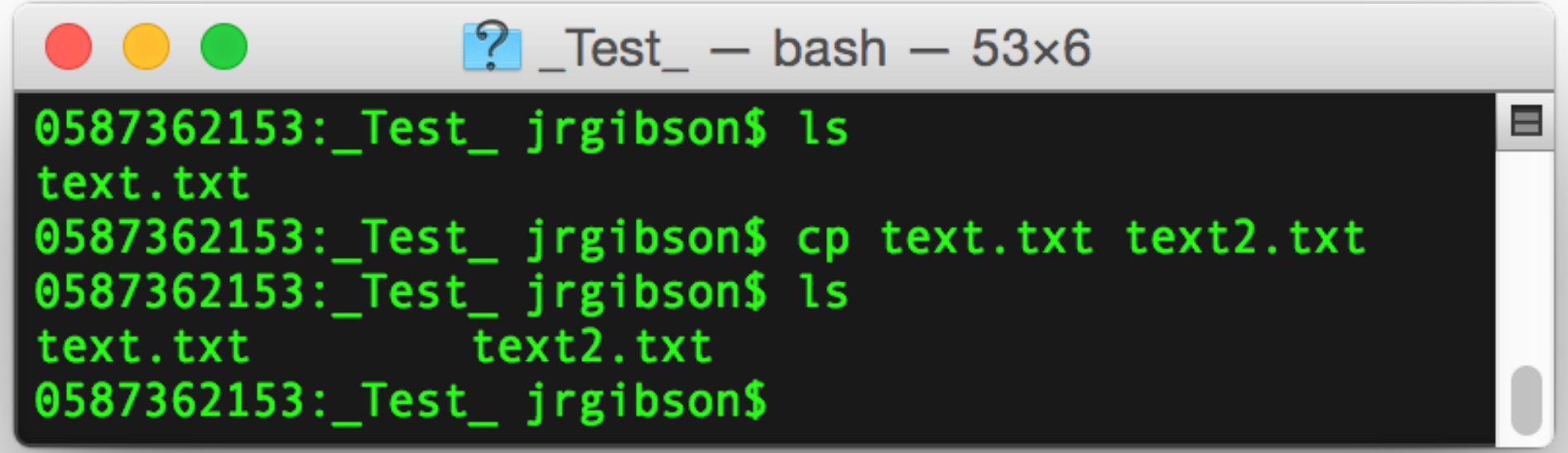
```
-rw-r--r-- 1 mdorf staff 455203 Nov 19  2012 conn4.exe

owner   group   other
 "who" are permissions being changed for
u      user who owns the file
g      group the file belongs to
o      other users
a      all of the above
```

Change permissions: **chmod**

```
-rw-r--r-- 1 mdorf staff 455203 Nov 19  2012 conn4.exe

owner   group   other
 "who" are permissions being changed for
u      user who owns the file
g      group the file belongs to
o      other users
a      all of the above

add 'execute'
  1) chmod u+x conn4.exe
  2) chmod g+x conn4.exe
  3) chmod o+x conn4.exe
  4) chmod a+x conn4.exe
```

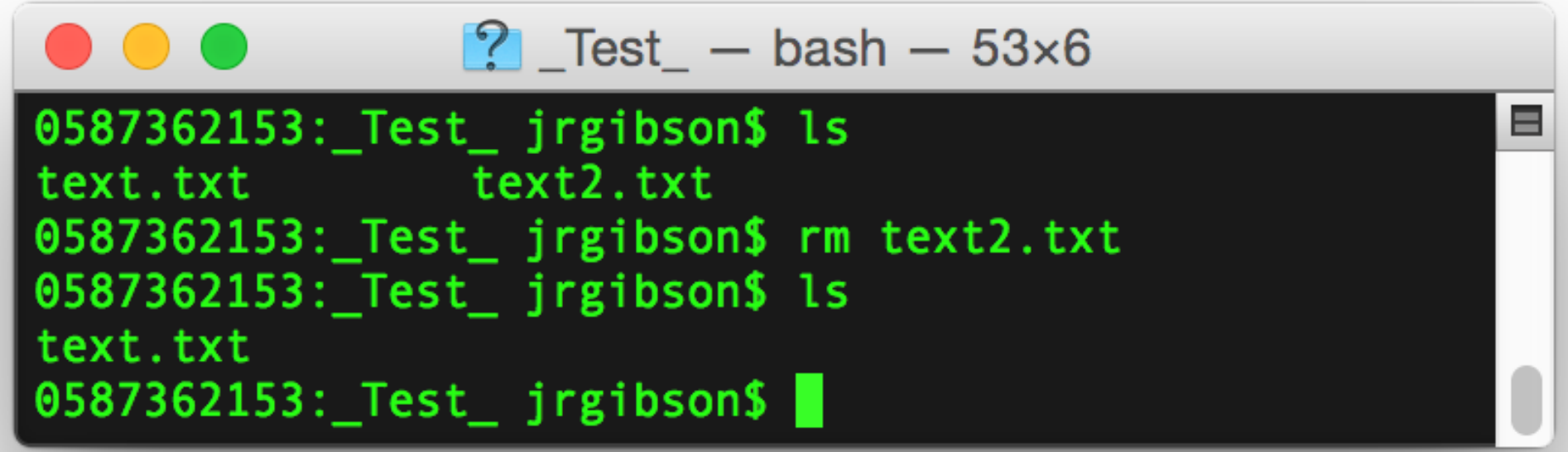
Copy file: **cp**

- **cp** xxxx yyyy copy xxxx file to yyyy file
  - Example: **`cp test.txt test2.txt`**



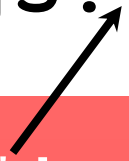
Remove: **rm**

- **rm** xxxx remove file xxxx
  - Example: **rm test2.txt**



remove

- `rm fileName`
- `rm proj5.*  //dangerous`



- This * is a "wild card"
- Anything will match it!
- ALL of these files would be deleted:
- proj5.txt
- proj5.mov
- proj5.exe
- proj5.key

remove

- `rm fileName`
- `rm proj5.* //dangerous`
- `rm *.* //DANGEROUS`

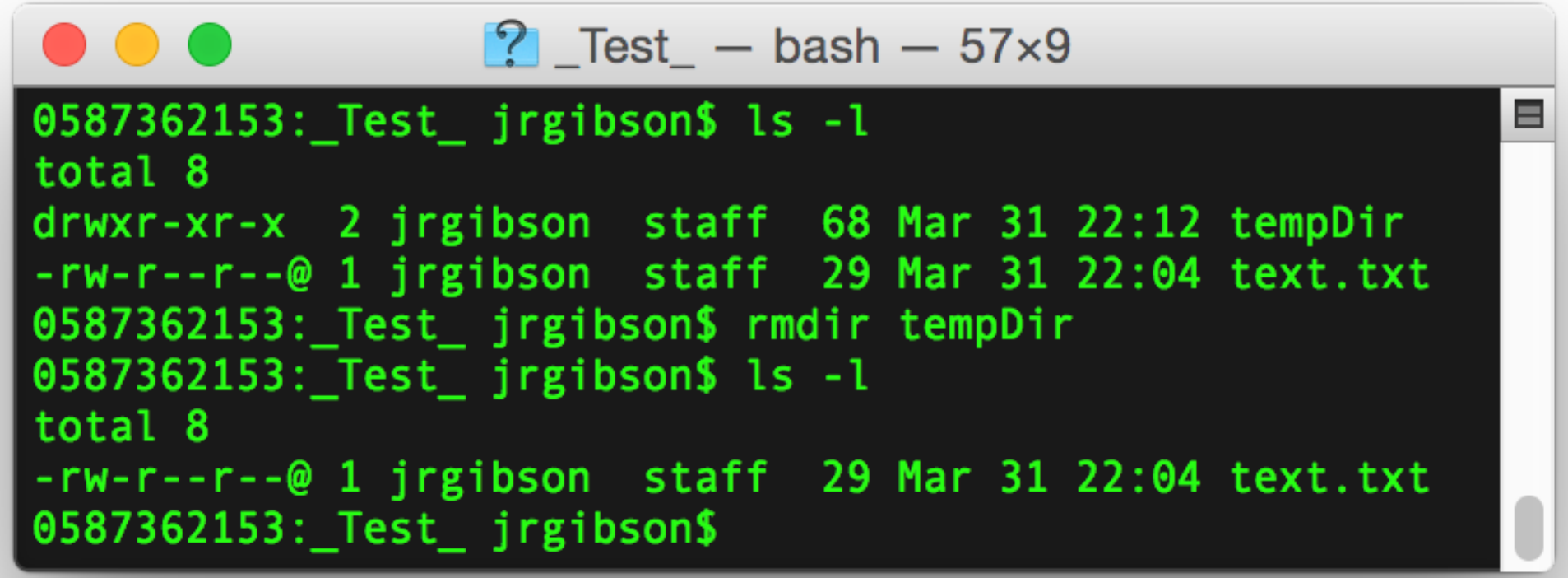
remove

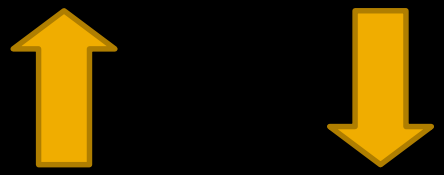
- `rm fileName`
- `rm proj5.* //dangerous`
- `rm *.* //DANGEROUS`
- `rm * //REALLY DANGEROUS`

*** will also match directories in the directory!!!!!**

remove directory: **rmdir**

- **`rmdir dirName`** Removes directory dirName



arrow keys 

- use up and down arrow keys to cycle through previously used commands
- easy way to type it once and run it often

compile

```
g++ debug.cpp
```

compile
but don’t build executable

```
g++ -c Point.cpp
g++ -c Line.cpp
```

compile

Compile every *.cpp in current directory and create executable

```
g++ *.cpp
```

compile

```
Compile every *.cpp in current
  directory and create executable

g++ *.cpp


./a.out     run the executable
```

compile

```
g++ fileName.cpp -o exeName


./exeName        run the executable
```

redirect input

```
./a.out < data1.txt
```

acts as though input is coming from keyboard

redirect output

```
./a.out > runOut.txt
```

puts all output to the file

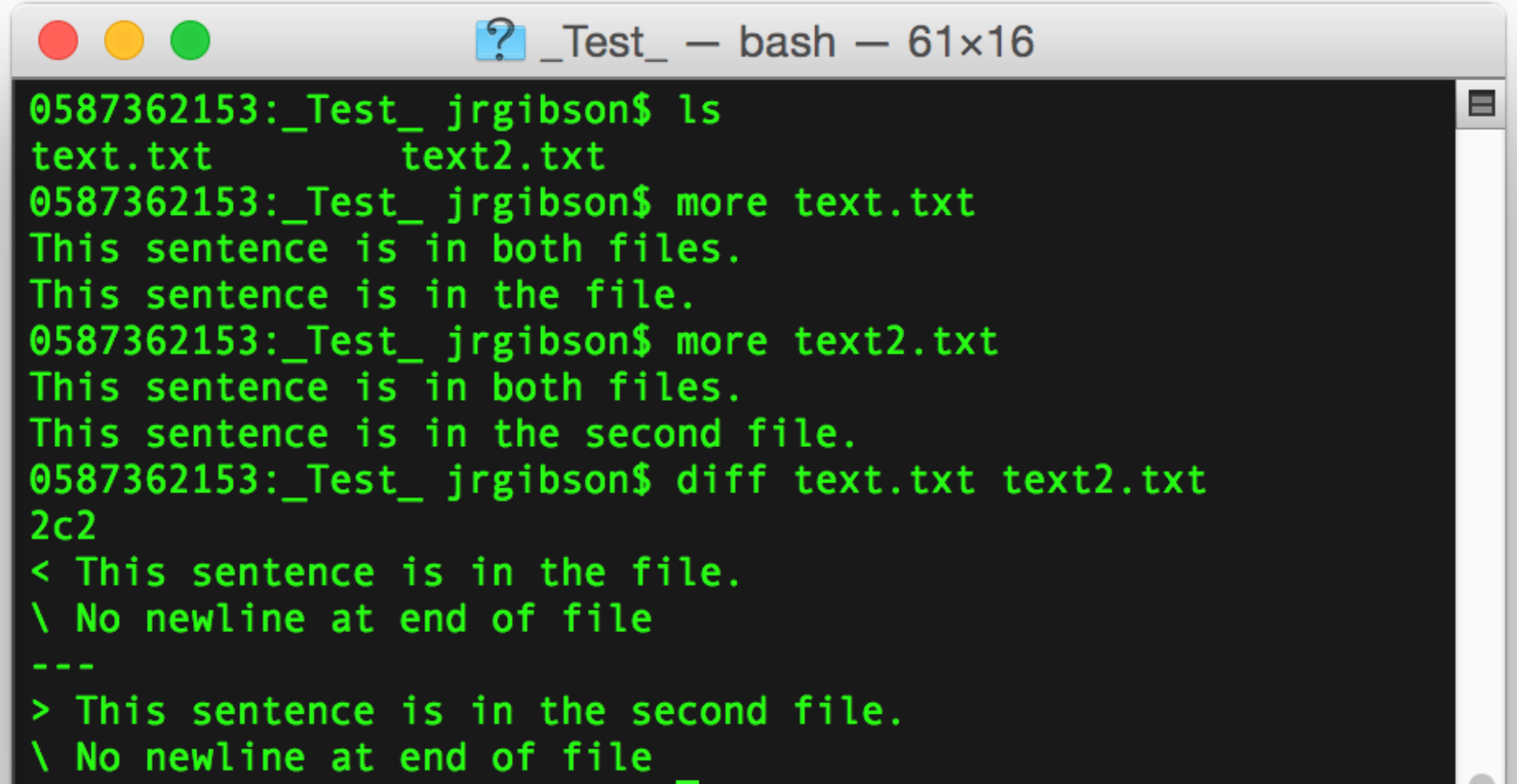
redirect both

```
./a.out < input.txt > runOut.txt
```

reads from "keyboard" and outputs to the file

Find Differences: **diff**

diff file1.txt file2.txt



Text editors

- `vim`
- `emacs`
- `Gedit`
- `nano`

Editor: vim

- **cheat sheets:**
  - http://www.tuxfiles.org/linuxhelp/vimcheat.html
  - http://bullium.com/support/vim.html

Minimum you need to know:
:q to exit

FYI:

- develop in Xcode/Visual Studio and upload to a linux system to make sure it works
  - upload is easy
  - PC: Putty (blue disk)
  - Mac: Fugu (blue disk)

- use redirection and different “input files”
- “diff” output with “expected output”

Request:

- If any faculty and/or GSI/IA in EECS 280 tells you that you MUST develop in linux
- please please relay this to 183 staff
- this is NOT true, nor needed
- use what is easy for you
  - upload into linux and rerun your test suite

Life after 183

EECS 280

- Similar to 183 – including workload
- programming in C++
- assumes:
  - conditionals
  - loops
  - functions
  - arrays/vectors

- continues from there
- going deeper into classes
- projects about as complex

EECS 203

- Paper/pencil course
- Learn theory used in EECS 281
  - proof techniques
  - analysis of algorithm
  - probability
  - concepts of trees, graphs

EECS 281

- pre-reqs
  - EECS 203
  - EECS 280
- Workload – higher
  - you design and implement
  - the addition of the "design" is what make it more difficult and time-consuming

Major

- Pre-Reqs
  - EECS 203
  - EECS 280
  - Calc I
  - Calc II
- EECS 281
- EECS 370
- EECS 376
- Stats 250

need at least a C in all
at least a 2.5 GPA over all

- 16 hrs of ULCS
  - 4 courses
- Capstone

Minor

- Pre-Reqs
  - EECS 183 need at least a C in all
  - Calc I
  - Taking or having taken one of EECS 203, 280, 281
- EECS 203
- EECS 280
- EECS 281
- One ULCS from a list

We are 183

Wednesday – Security