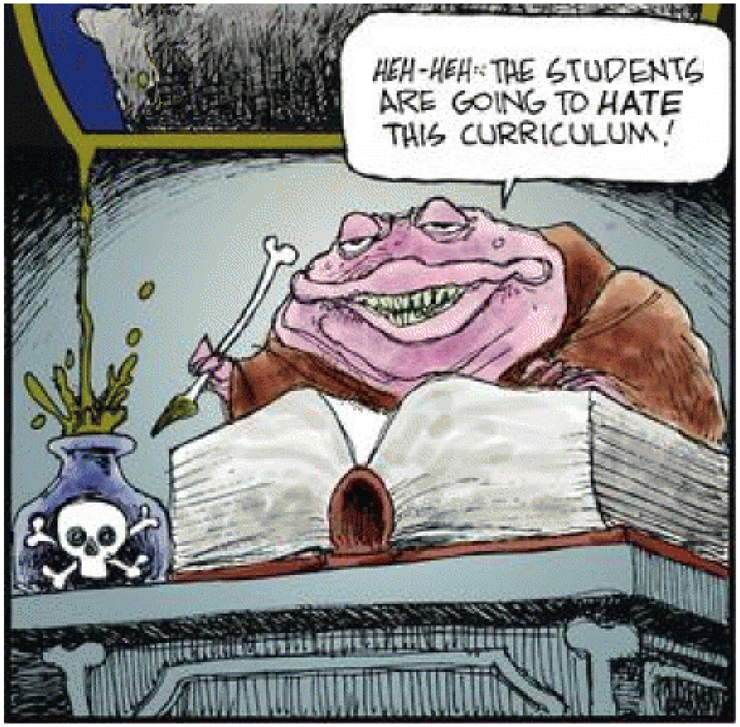
CMPE380
Applied Programming

Course Overview

Course Objective

- Learn "C"
  - Dynamic memory, timing and performance



© Non Sequitur by Wiley Miller

- Numerical Computing
  - Learn to love Linear Algebra
  - Root finding, Gaussian elimination, least squares, interpolation, cubic splines, numerical differentiation and integration, simulation

Spring Class Info

- Monday/Wednesday 5:00PM - 6:15PM
  - LBR-A201
  - No Class March 13, 15

- A First Course in Numerical Methods
  - U. Ascher, C. Greif, SIAM Publishing,
  - ISBN 978-0-898719-97-0
  - New $91
  - Not required

Online Materials

- MyCourses
  - Lectures notes, References
  - Home work, Drop box
  - Grades

Homework Policies

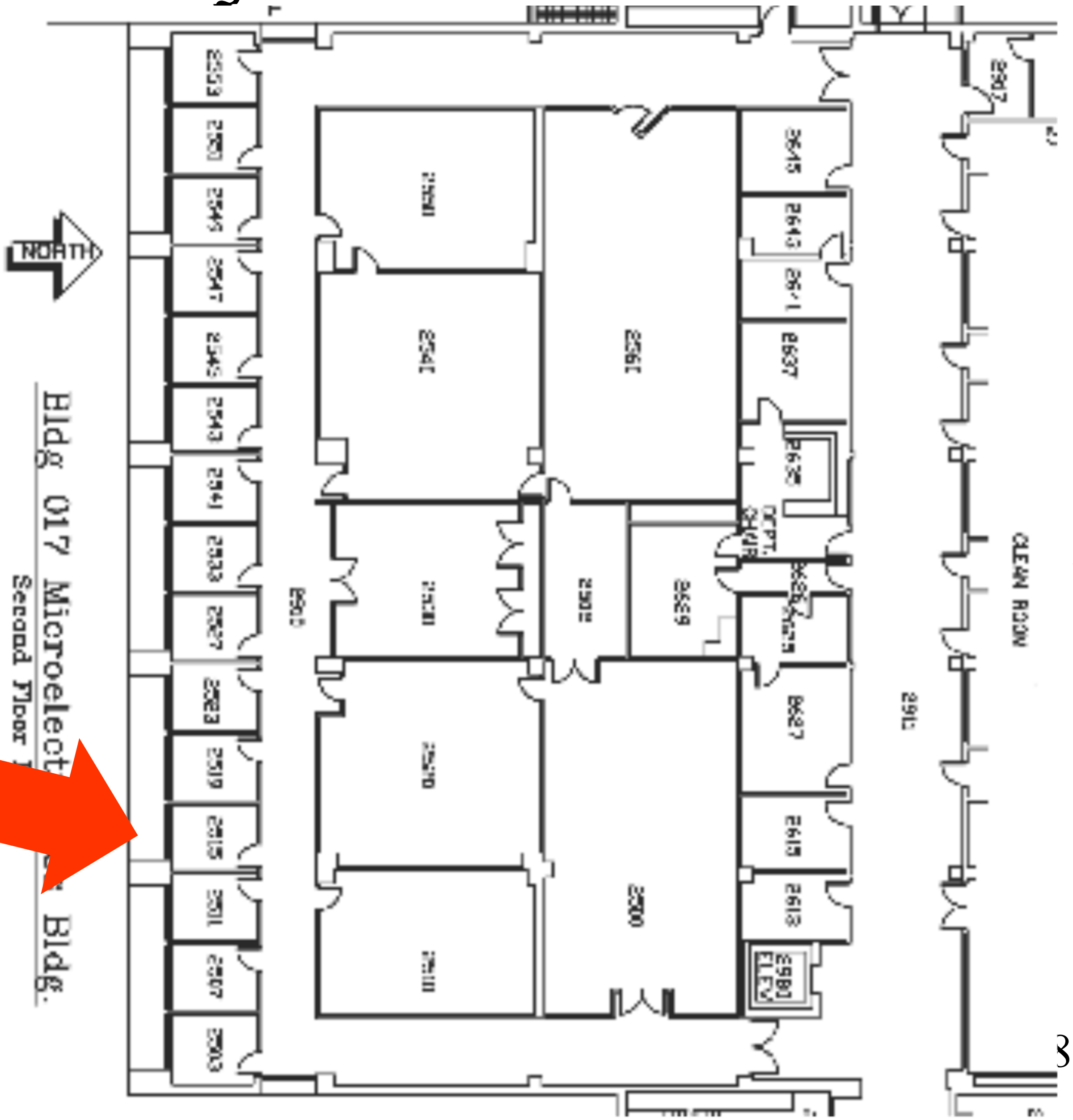
- All work **must** be YOURS

- Deadline - ALWAYS 11:50 PM
  - Submit your homework to MyCourses – drop box
  - Must be a **TAR** file
    - Must contain ALL required files
    - Only the **LAST submission** is retained
    - Always verify your tar files
  - **10% daily penalty for late submissions (1-5 days)**
  - **100% after 5 days**.

Help

- Email questions - **rhreec@rit.edu**
  - **Email** your work to me
- ADA
  - Let me know about any special accommodations
- Office Hours
  - Rm 17-2515
  - Sunday at 1:00 PM
    - Or by appointment (Email me a day ahead of time)
    - Or after Class
- Talk to me if there are issues!

My Micro-E Office



My Office:
17-2515

Micro E
Clean Room

Grading

- 6 Quizzes 15%
  - **No Final or midterm**
  - Drop lowest
- 9 Homework's 85%
  - Drop lowest
- Final Grade: (Curve applied if necessary)

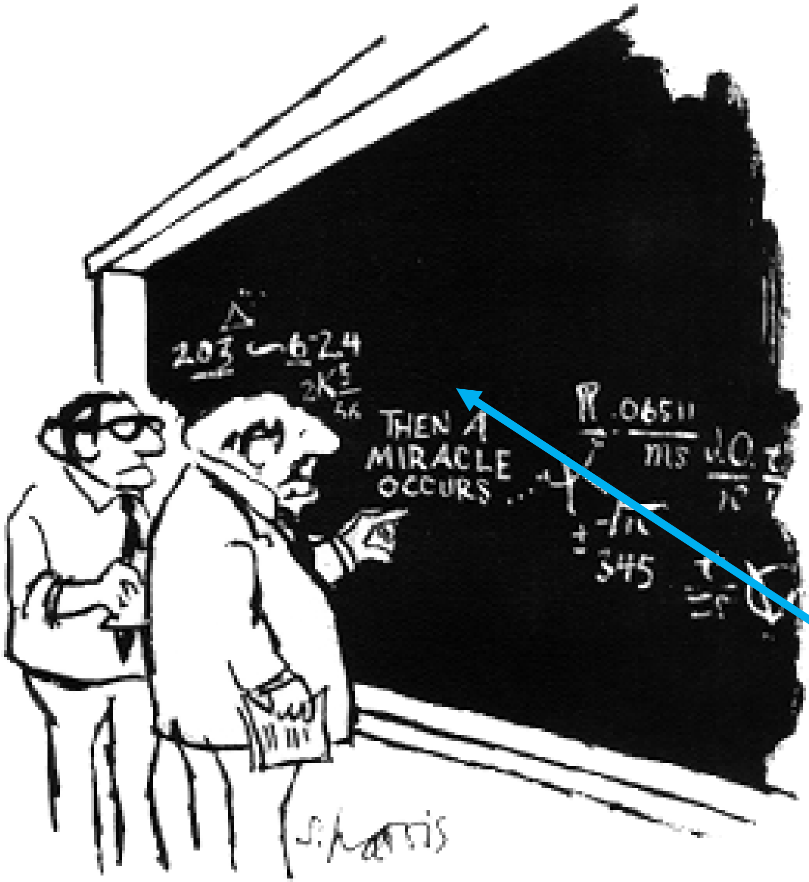
100 - 90	89 - 86	85 - 83	82 - 80	79 - 76	75 - 73	72 -70	69 - 66	65 - 60	59 - 0
A	A-	B+	B	B-	C+	C	C-	D	F

Quizzes

- At the end of class every other Wednesday
- Based on the previous home work(s) and class problems

- Non-programmable calculator allowed on most quizzes
- No computers or cell phones
- If you miss the quiz you get a zero
  - Contact me AHEAD of time if you have conflicts

Expectations: Know Why



Copyright 1999 by Sidney Harris

Getting the correct answer is not the most important part of the solution

"I think you should be more explicit here in step two."

How to Succeed in this Course

- Start the homework early.
  - Don't wait until the due date
    - 10% penalty per day for late work. 100% after 5 days
  - Homework is HARD
    - First is easy ☺
- Review the material discussed in class before the next lecture

Prerequisites

- Software Development
  - You are expected write properly documented code (`including comments`) for all homework
  - Poorly commented programs will be penalized, we will be more strict as the semester progresses

- Unix / Linux
  - Working knowledge of Linux development environment
    http://www.cs.purdue.edu/homes/amadkour/files/presentations/LinuxCrashCourse.pdf

(slides 40+)

- File Transfer using `sftp`, `scp` or `filezilla`

- Compiling / Linking modules and programs using the command line and

Prerequisite Topics

- Mathematics and Calculus
  - Mathematical functions, continuity, limits, approximation, convergence, derivatives and integrals
  - Infinite series and Taylor series expansion
  - Differential equations
  - Basic Matrix Algebra

Prerequisite Topics

- “Computer Science”
  - Recursion
  - Searching
  - Data Structures (lists, queues, trees)
  - Functional Decomposition
  - Principles of file I/O

About Linear Algebra

- Linear Algebra I
  - We will be solving numerically *systems of linear equations.*
  - Mostly need to have a working knowledge of *Matrix Arithmetic* and *Gaussian Elimination*

Course Objectives

1. Improve your **proficiency in** the **C language**
   - Homework will be assigned to write C programs and modules implementing the various algorithms discussed in class.

2. Learn how to select **suitable algorithms** to solve (numerically) engineering problems (Scientific Computing)
   - Numerical algorithms will be presented and discussed in class. You will be tested on these in the quizzes.

Online Resources

- MyCourses will be used:
  - Assignments
  - Reference material
  - Grades

Computing Resources

- Use any of the *VLSI-lab machines*
  - `eng-2500-XX.main.ad.rit.edu`
    - `XX=06-30`

      `E.g. eng-2500-06.main.ad.rit.edu`

- Physically in the lab or remote login
  - **Remote recommended**

- login with your RIT credentials

Remote Access

- Only SSH logins accepted
- Use your RIT login and password

- Off-campus MAY require a VPN connection
  - Windows/Mac: http://www.rit.edu/its/services/vpn/protected
  - Linux `openconnect (see appendix`)
  - No VPN required on campus
  - Try access without the VPN first

System Problems

- With accounts

  Rick Tolleson

  Email: `rateec@rit.edu`

  Office: GLE-3411

- With systems: **`*.ce.rit.edu`**

  Rich Flegal (sysadmin)

  Email: rkfeec@rit.edu

  Office: GLE-3415

High Level Tools

- Use secure **terminal** emulators to access Linux systems from you PC/Mac
  - **ssh, PuTTy**

- Use secure **file transfer** tools to copy files between PC/Mac and Linux
  - **scp, winscp, filezilla**

- Run an **X-terminal** emulator to see Linux graphics
  - **Xming (pc)**
  - **Xquartz (Mac)**

Remote Access

- Only SSH logins accepted
- Use your RIT login and password

Homework Workflow

1. Download homework files from MyCourses
2. Copy them (**scp, winscp, filezilla**) to Linux
3. Unarchive your files into directories called
    1. **`hw1`** for hw # 1, **`hw2`** for hw # 2, etc …
4. Write your program, makefiles, etc (**`vi, nano`**)
5. Compile and debug your program (**`gcc, make`**)
6. Archive your work files (**`tar`**)
7. Copy back to your PC/Mac (**scp, winscp, filezilla**)
8. Upload and submit to drop box

Analysis.txt

- Required for each homework
  - Spelling & Grammar are important
  - Formatted for 78 column text terminal
  - All the time, every time

- Normally a full page of text or more. (~3KB)

Analysis.txt

- Tell me the **story** of your homework.

- Should include
  - What you did
  - How you did it
  - Problems encountered and the resolution
  - Important intermediate results or tests
    - Data tables and comparisons
  - Any conclusions

Analysis.txt - bad

- “When performing iterations for timing purposes, the Secant method took the longest amount of time. The Newton method took the second longest amount of time.”
  - How much longer?
  - How many times did you run the test, 1 or 100?
  - Absolute seconds AND percentage difference?
  - A data table would be good here.

PuTTY

- SSL terminal emulation program
  - For Windows
  - Download

    http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html

- Configuration
  - Next 3 slides

  http://dag.wiee.rs/blog/content/improving-putty-settings-on-windows

PuTTY Summary 1

- Session
  - Connection: SSH
  - Close window on exit: Never
- Terminal Bell
  - Action: Visual bell
  - Taskbar/caption indication: Flashing
- Window
  - Lines of scrollback: 20000
  - Reset scrollback on keypress: Checked
  - Reset scrollback on display activity: Unchecked

PuTTY Summary 2

- Window Appearance
  - Font: Lucida Console, 9-point
  - Font quality: ClearType
  - Gap between text and window edge: 3
- Window Translation
  - Character set: UTF-8
  - Handling of line drawing characters: Unicode
- Window Selection
  - Action of mouse buttons: Compromise
  - Paste to clipboard in RTF as well as plain text: enabled

PuTTY Summary 3

- Window > Colours
  - ANSI Blue: Red:74 Green:74 Blue:255
    ANSI Blue Bold: Red:140: Green:140 Blue:255
- Connection
  - Seconds between keepalives (0 to turn off): 25
- Category Connection
  SSH (expand) X11
  - Enable X11 forwarding: enabled

Mac Terminal Commands

- Open the Mac terminal
  - Launch pad -> other -> terminal
- ssh xxxxxx@**eng-2500-07.main.ad.rit.edu**
  - **Where xxxxxx is your RIT computer account name**
  - **Use –X to start an X server**
  - **Use –Y to disable connection time out**

e.g. ssh rhreec@eng-2500-07.main.ad.rit.edu –X -Y

- Enter your RIT password
  - Execute Linux commands

Mac File Commands

- Open a Mac terminal
- scp <file> xxx@eng-2500-07.main.ad.rit.edu:~/<file>
  - Where xxx is your RIT computer account name
  - <file> the item you want to copy
- Enter your password

Examples:

```
scp hw01.tar rhreec@eng-2500-07.main.ad.rit.edu:~/hw01.tar
scp rhreec@eng-2500-07.main.ad.rit.edu:~/hw01.tar hw01.tar
```

- Use the opposite order to copy data from Linux

X-Windows

- X-Windows is the Linux graphical display system
  - Used to draw graphs and view images
  - NOT required for simple text display
- Download Xming for Microsoft X-windows support
  - For Microsoft Windows
  - Follow installation instructions

  http://sourceforge.net/projects/xming/

Mac X-Windows

- OS X no longer ships with an X server
  - download XQuartz:

https://www.xquartz.org/

- Download and install it
  - log out and back into your Mac to refresh the DISPLAY environment variable

Mac X-Windows

- Example:

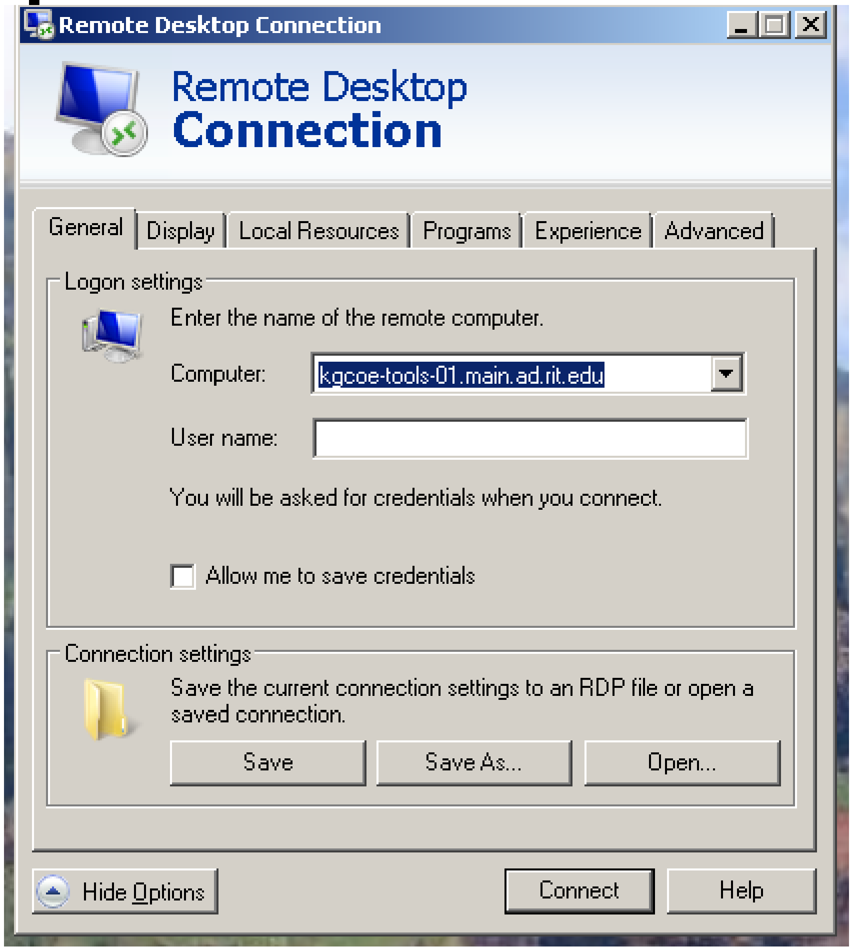
  Start Xquartz (if not already running)

```
ssh xxxxxx@eng-2500-07.main.ad.rit.edu   -X –Y
```

  - **Where xxxxxx is your RIT computer account name**
  - **Note the capital “-X –Y”**

Remote desktop - XRDP

- Use Windows or Mac remove desktop tools
  - – Provides X support



Logon settings

Enter the name of the remote computer.

Computer: kgcoe-tools-01.main.ad.rit.edu

User name:

You will be asked for credentials when you connect.

□ Allow me to save credentials

Connection settings

Save the current connection settings to an RDP file or open a saved connection.

Save | Save As... | Open...

- Select “options”
- Enter:

  Computer: kgcoe-tools-01.main.ad.rit.edu

  User name: <your user id>

  - You may see security warnings

Remote desktop - XRDP

- Click on: System tools -> terminal
  - No extra PC software needed

- Click on <your name> -> quit
  - To exit

  - Note: This does not execute bash so you will have to set the required environment variables yourself.

Matlab Settings

- Enable Matlab in your current session:

  module load matlab

- Run Matlab: matlab
  - Note: Matlab will start in GUI mode with a splash screen
  - Requires Xwindows

Module info: wiki.rit.edu/display/kgcoeuserdocs/Modules

Optional Matlab Settings

- Matlab can be run in character terminal mode.
  - No graphical displays or plots (no X needed)
  - matlab -nosplash -nodisplay –noawt
  - Faster graphical startup
  - matlab -nosplash

- Add an alias to your .bashrc:

```
alias tmatlab=‘matlab -nosplash -nodisplay –noawt’
alias xmatlab=‘matlab –nosplash’
```

  - Rerun your .bashrc file (source ./.bashrc)

Multiple Terminals

- Linux supports more than one window into the system at a time
  - One to compile
  - One to edit

- Use multiple PuTTy, or other tools instances

- Use Linix “tmux” feature

Multiplexing Connections

- `tmux` (terminal multiplexer)
  - It creates multiple “screens” and divides screen in “panes”
  - Reference in MyCourses

- Intro video

http://www.youtube.com/watch?v=BHhA_ZKjyxo

- Cheat sheet in myCourses

Minimal tmux

- The absolute minimum you need to know:
  - Preceded commands with the action: **<ctrl-b>**

- Starting: type “tmux”
  - New window: <ctrl-b> c
  - Rename window: <ctrl-b>, (comma)
  - Previous/next window: <ctrl-b>p/n
  - Kill current window: <ctrl-b>&

- Help: <ctrl-b> ?

Minimal tmux

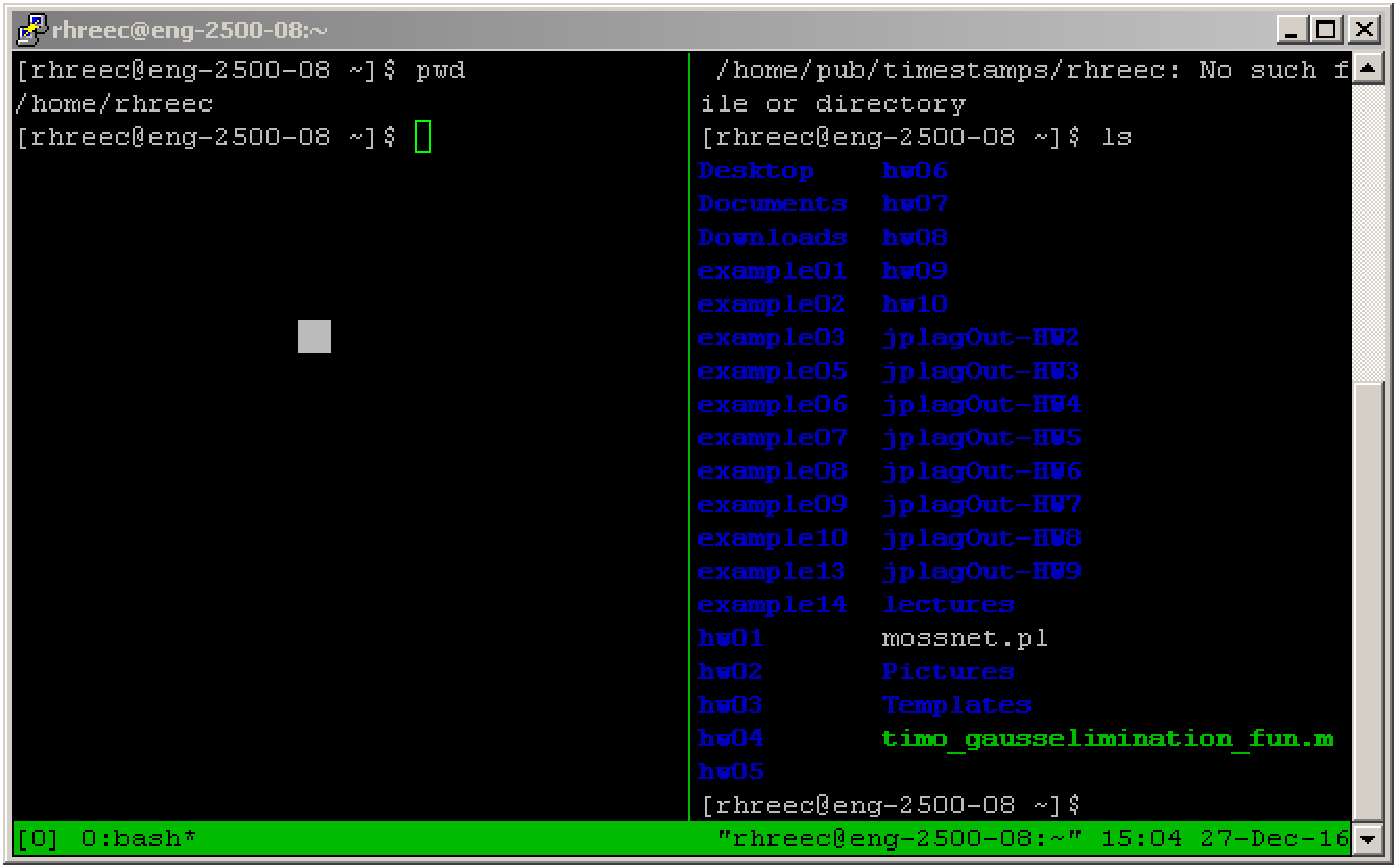
- split horizontal, pane below: <ctrl-b> "
  - Select upper/lower: <ctrl-b> (up/down arrow)

- **Split vertical, pane right: <ctrl-b> %**
  - **Select left/right: <ctrl-b> (left/right arrow)**

- Close split: <ctrl-b> !

Tmux - example

- Commands: tmux   <ctrl-b> %



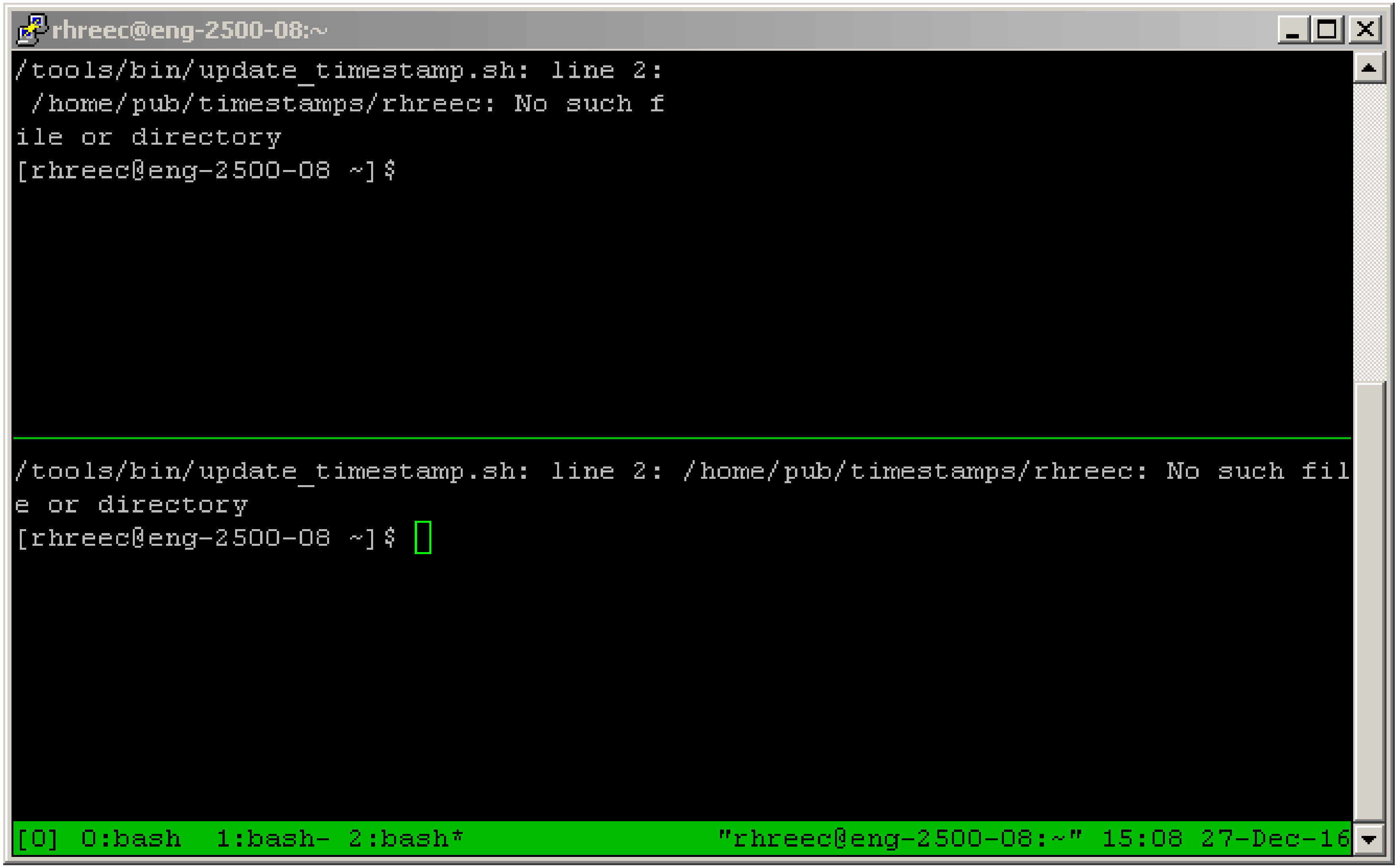
```
[rhreec@eng-2500-08 ~]$ pwd
/home/rhreec
[rhreec@eng-2500-08 ~]$ 
```

```
 /home/pub/timestamps/rhreec: No such f
ile or directory
[rhreec@eng-2500-08 ~]$ ls
Desktop      hw06
Documents    hw07
Downloads    hw08
example01    hw09
example02    hw10
example03    jplagOut-HW2
example05    jplagOut-HW3
example06    jplagOut-HW4
example07    jplagOut-HW5
example08    jplagOut-HW6
example09    jplagOut-HW7
example10    jplagOut-HW8
example13    jplagOut-HW9
example14    lectures
hw01         mossnet.pl
hw02         Pictures
hw03         Templates
hw04         timo_gausselimination_fun.m
hw05
[rhreec@eng-2500-08 ~]$
```

```
[0] 0:bash*                               "rhreec@eng-2500-08:~" 15:04 27-Dec-16
```

Tmux - example

- Commands: tmux <ctrl-b> "



```
/tools/bin/update_timestamp.sh: line 2:
 /home/pub/timestamps/rhreec: No such f
ile or directory
[rhreec@eng-2500-08 ~]$
```

```
/tools/bin/update_timestamp.sh: line 2: /home/pub/timestamps/rhreec: No such fil
e or directory
[rhreec@eng-2500-08 ~]$
```

```
[0] 0:bash  1:bash- 2:bash*                   "rhreec@eng-2500-08:~" 15:08 27-Dec-16
```

File Transfer

- To upload and download files use secure copy (scp)
- Linux command line **scp**

  http://www.examplenow.com/scp/

- FileZilla (Windows, cross-platform)

  http://filezilla-project.org

- WinSCP (Windows)

  http://winscp.net/eng/index.php

Linux Tools Used

- Compiling: `gcc`,
- Debugging: `gdb`, `valgrind`
- Editing: `nano` or `vim`
- Other: `tar, display, make`

Homework Summary

- Will use mostly C99
- Due at 11:50 pm (not midnight) on the dates shown in the Drop Box (MyCourses)
- All the tools needed to complete your homework are available in the CE machines.
- Program must run and compile on the CE machines (it does not matter where you develop it)
- Homework must be submitted before the deadline
  - **10%/day late penalty for days 1-5**
  - **100% late penalty after 5 days**

Homework Submissions:

- Due ON TIME, no excuses, PERIOD
- Some invalid excuses:
  - The clock in the server was wrong and did not take my submission.
  - My hard disk (flash drive, etc.) crashed.
  - The Internet was down when I was going to submit my work so I missed the deadline.
  - **I accidentally erased my files when preparing the tar file.**
  - The grader said that my tar file was empty or corrupted but it was fine when I sent it.

Applied Programming

- **Short History of the C Language**

Know your AB

- **C** *is* a procedural (or *imperative*) programming language designed by Dennis Ritchie in 1972 at "AT&T Bell Labs" as an improvement over **B** (Ken Thompson of Bell Labs) for the development of an operating system for a PDP-11
- Dennis Ritchie and Ken Thompson received the *National Medal of Technology* in *1998*

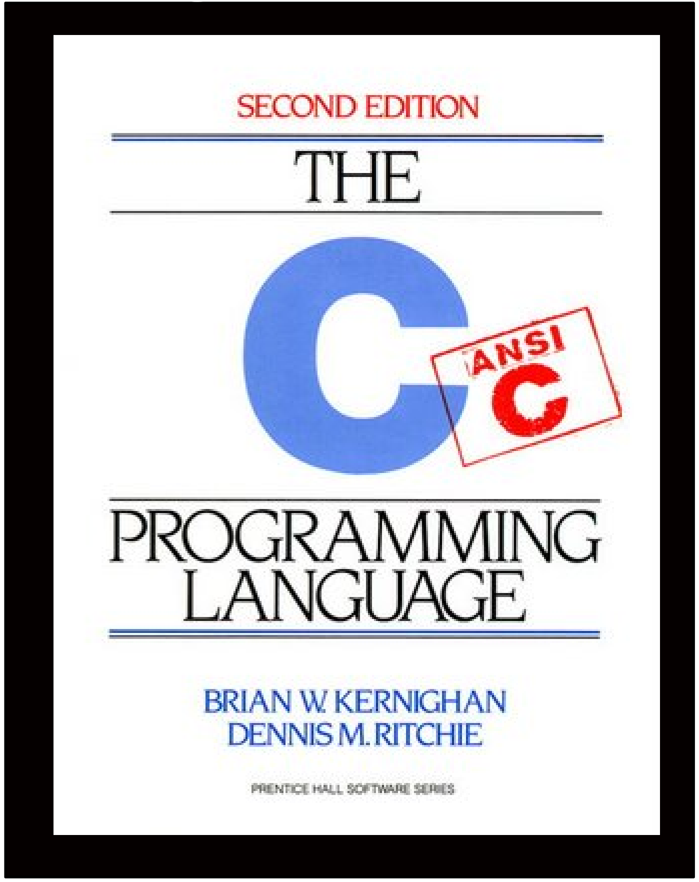
http://www.youtube.com/watch?v=LXZ1OL2U3lY

Know your AB

- **C** was standardized by the American National Standards Institute in the 80s to become **ANSI C** (ANSI X3.159-1989)

- The original C is called *KR* C

- The International Standards Organization followed with the development of their standard for the C language (ISO 9899:1990)



Where is the Fun ?

image from www.ioccc.org

- **The International Obfuscated C Code Contest**

  http://www.ioccc.org/

- **Goals of the Contest**
  - Write the most Obscure/Obfuscated C program
  - Show the importance of programming style
    - In an ironic way.
  - To stress C compilers with unusual code.
  - To illustrate subtleties of the C language.
  - To provide a safe forum for poor C code. :-)

Obfuscated C Code: Example

- This is an example of a winner

```
#include <stdio.h>
#include <math.h>
#include <unistd.h>
#include <sys/ioctl.h>

             main() {
         short a[4];ioctl
      (0,TIOCGWINSZ,&a);int
    b,c,d=*a,e=a[1];float f,g,
  h,i=d/2+d%2+1,j=d/5-1,k=0,l=e/
 2,m=d/4,n=.01*e,o=0,p=.1;while (
printf("\x1b[H\x1B[?25l"),!usleep(
79383)){for (b=c=0;h=2*(m-c)/i,f=-
.3*(g=(l-b)/i)+.954*h,c<d;c+=(b=++
b%e)==0)printf("\x1B[%dm ",g*g>1-h
*h?c>d-j?b<d-c||d-c>e-b?40:100:b<j
||b>e-j?40:g*(g+.6)+.09+h*h<1?100:
 47:((int)(9-k+(.954*g+.3*h)/sqrt
  (1-f*f))+(int)(2+f*2))%2==0?107
    :101);k+=p,m+=o,o=m>d-2*j?
      -.04*d:o+.002*d;n=(l+=
         n)<i||l>e-i?p=-p
             ,-n:n;}}
```

- What does it do ?

Exercise 1

- What is a “**X**” terminal and when do you use one?

- “X” is the Linux/Unix graphical display terminal language.
- “X” is required whenever you want to display an image or plot.
  - Linux/Unix normally displays TEXT in a character terminal window.

HW Problem Review