Introduction to UNIX™

- why learn unix?
- about this course
- about unix
- about cygwin
- some basic commands
- the more and man commands
- the look command
- the shell
- exercise 1
- some additional commands
- exercise 2
- the file system
- path names of files
- commands affecting directories
- accessing RCS files
- exercise 3
- environment variables
- bash functions
- exercise 4
- for more information
- take-home exercise
- future short courses

conducted by

Mike Kupferschmid
Scientific Programming Consultant
VCC 322, x6558, kupfem@rpi.edu
Why Learn unix?

to do research computing
  writing your own programs
  using programs others wrote for unix
  using the ARC batch and Linux clusters

as a prerequisite to other short courses
  vi, advanced vi, and emacs
Shell Programming
Parallel Programming with MPI

for your own PC
  cygwin
  Linux
  FreeBSD

for your resume
  unix is widely used in industry

“On the road of computing,
  there are passengers and there are drivers.”
About This Course

This is NOT a course in operating systems!

To learn about the design or internals of unix, take CSCI-4210 Operating Systems.

This is a course in how to use unix to do some very simple things.

To learn more about using unix, read about Shell Programming short course.
About unix

you converse with unix in a unix window on a Windows PC, use cygwin from a Windows PC, SecureCRT to a server running Linux, open an xterm

you type a command followed by RETURN commands are mixed-case

unix does what you told it to do no confirmation is requested from you no acknowledgement is given some commands generate printed output but many do not on failure most print error messages

unix has no use for the mouse you might be able to cut and paste some applications use the mouse but unix itself has nothing to click

using unix often involves manipulating files computer data are stored in files unix files are created by reference
About cygwin

a good but not perfect emulation of unix
included in standard RPI laptop image
to download (long) go to www.cygwin.com

if you want real unix get Linux (site licensed)
it is possible to dual-boot Linux with Windows
ask at the Help Desk
watch for an ACM installfest

starting cygwin
  start → All Programs → Cygwin → XTerm
opens unix window with command prompt
  $
RCShome is a link to your RCS home directory

to print a file from cygwin
  open the file with notepad filename
  use the notepad print function

to cut and paste you need a 3-button mouse
cygwin names executables a.exe
Some Basic Commands

display file contents by screens \texttt{more}
display a manual page \texttt{man}
list file names \texttt{ls}
rename a file \texttt{mv}
copy a file \texttt{cp}
concatenate files \texttt{cat}
compare two files \texttt{diff}
remove a file \texttt{rm}
show the date and time \texttt{date}
clear the screen \texttt{clear}

Many commands have options; for example,
\texttt{ls -l} lists files with dates and other data.
\texttt{ls -F} appends a suffix showing file type.

To interrupt any unix command,
hold down the \texttt{CTRL} key and press \texttt{c}. 
The more and man Commands

more filename

displays the contents of the file named filename, one page at a time.
to see the next page, press the space bar
to see the next line, press the enter key
to go back up, press b
to go down, press d
to search for string, type /string

man commandname

uses more to display the manual pages
describing the unix command named commandname, and its options;
to exit type ZZ.

On most unix systems, man pages are the primary on-line source of information about the system.
The look Command

```
prompt[1] look gn
gnash
gnat
gnats
gnaw
gnawed
gnawing
gnaws
gnome
gnomon
gnu
```

The `look` command shows dictionary words that begin with the given character string. This example shows the words beginning `gn`.

`look` writes to `standard-out`, which is normally the screen.
The Shell

The **shell** is the unix command interpreter.

It can redirect standard-out to a file

- `look word > junk` write to empty file
- `look word >> junk` append to existing file

Or pipe standard-out into another command

- `look word | more` *more* reads from standard-in

Wild cards can be used in filenames

- `ls *` list filenames in this directory and subdirectories
- `ls *.tex` list filenames ending `.tex`
- `ls *.*` list only filenames containing a dot
- `ls 200?` list filenames starting 200 and ending with any single character
- `ls ?` list 1-character filenames

To use a special character literally, escape with \.
Example: `rm a\*b` removes the file named a*b.
Exercise 1

Figure out what command you should enter to list, on the screen, the names of all your files and their last change times, sorted in order from newest to oldest.

How can you save this list in a new file named mystuff?

– answer on next slide –
Exercise 1

Figure out what command you should enter to list, on the screen, the names of all your files and their last change times, sorted in order from newest to oldest.

How can you save this list in a new file named mystuff?

```bash
ls -l -t > mystuff
```
Some Additional Commands

**cal**  display the calendar for a month  
     cal 11 1946

**wc**  count lines, words, and characters  
     wc mystuff

**grep**  scan for a given character string  
     grep RCS mystuff

**sort**  sort lines of text  
     sort mystuff

**vi**  simple text editor

**cc**  C language compiler

**f77**  FORTRAN language compiler

**make**  automates program maintenance

**tar**  tape archive program
Exercise 2

How many spelling words begin with “air” and contain the letter “e”?  

– answer on next slide –
Exercise 2

How many spelling words begin with “air” and contain the letter “e”?  

look air | grep \ e | wc
The File System

unix files are organized in directories.

File and directory names are case-sensitive.
Path Names of Files

A **path** tells how to get to a file or directory.

- **absolute** tells how from a given directory
- **relative** tells how from the current directory

Special names for directories

- `.` current directory
- `..` parent directory
- `~` home directory of current user
- `~smith` home directory of user **smith**

In **cygwin** you are the only user.

Some possible paths to **schedule**

- `/home/jones/schedule` (absolute)
- `~jones/schedule` (absolute)
- `../schedule` (relative to **letters**)

Commands Affecting Directories

```
cd   change directory
    cd  path  to change to there
    cd   to change to home directory

pwd  print working directory in cygwin
    /bin/pwd print working directory in RCShome

mkdir make a new directory
    mkdir tryit

rmdir remove a directory (must be empty)
    rmdir tryit
```
Accessing RCS Files

The `cygwin /home` directory is local to the PC, so `~smith` is `/home/smith` in cygwin.

RCS files are under `/afs/rpi.edu/home/nn`.
`RCShome` is a link to your RCS home directory in `RCShome /bin/pwd` will tell you `nn`.
On RCS you can use these commands (see `fs help`):

- `fs lq` display disk allocation and usage
- `fs listacl` display permissions for a directory
- `fs setacl` set permissions for a directory
  `fs setacl . smith rl`
Exercise 3

Fill in the missing file and directory names in the diagram below.

– answer on next slide –
Exercise 3

Fill in the missing file and directory names in the diagram below.

```
/afs/rpi.edu/home
  `..`
  05
  ..
  home directory
  kupfem2
    public  hello.f  Mail  bin  yesterday
      `sun4x_56  sgi_65  rs_aix43`
        `pgm1`
```
Environment Variables
used to configure your unix session

many variables are predefined for you
to see them all use  
env | more

PATH tells unix where to look for executables
to see it use  
echo $PATH

you can reset the values of existing variables
PS1="What is thy bidding? "

and you can make up your own variables
Y="look y"

 echo $Y  prints  look y
$Y  executes the command  look y

export makes them visible to other processes
export AFILE="datafile"
now AFILE can be read by a program

Using an editor such as vi or emacs, you can
define environment variables in your .bashrc file
so they are present whenever you start a unix window.
bash Functions
used to abbreviate long commands

function lsd { /usr/bin/ls -1F | grep "/" ;}
Now entering lsd lists only the directories.

function ftn { f77 -C -g $* ;}
Now entering ftn file compiles file.
This saves having to type -C -g.

ftn /afs/rpi.edu/home/05/kupfem2/hello.f
a.exe

Using an editor such as vi or emacs, you can
put function definitions in your .bashrc file
so they are defined whenever you start a unix window.
Exercise 4

Write a bash function named myls that invokes ls using whatever flags are set in the environment variable LSFLAGS. Set LSFLAGS to various values and verify that your function does what it should.

(LSFLAGS is a variable we invented, not one of the variables that is pre-defined for you.)

– answer on next slide –
Exercise 4

Write a bash function named *myls* that invokes *ls* using whatever flags are set in the environment variable *LSFLAGS*. Set *LSFLAGS* to various values and verify that your function does what it should.

```bash
function myls {
  /usr/bin/ls $LSFLAGS ;
}
```

Here is an alternative that allows command-line options to be used as well.

```bash
function myls {
  /usr/bin/ls $LSFLAGS $* ;
}
```

This is how the function might be used.

```
$ export LSFLAGS="-F"
$ myls
```

The output produced is that from *ls* `-F`
For More Information

review the Shell Programming course handouts
read RPI Memos 112, 113, and 114 on unix
read books on unix and Linux from O’Reilly
    look at www.ora.com
get consulting help from Mike
    VCC 322, kupfem@rpi.edu, x6558
Take-Home Exercise

Use a single unix command to list the names of all files in the RCS directory

/afs/rpi.edu/campus/math/lapack/2.0/distrib/SRC

whose names end in .f and include the letter combination dt. Are there 20 file names in your list? What does the file dtzrqf.f contain?

Write a bash function named cksrscs that executes the command but gets the directory name from an environment variable named SRCDIR and allows the letter combination to be specified as a parameter. Test your creation by using

cksrscs dt

to generate the same list you obtained before.
– answer on next slide –
function cksrcs { ls -1 $SRCDIR *.f | grep $1 ;}

export SRCDIR=/campus/math/lapack/2.0/distrib/SRC

cksrcs dt