Recall: Technical Interlude (Don't Panic)

Need to perform some Linux magic. The next steps will seem obscure.

We need to change your "working shell.

The "shell" is the set of commands that you type at a screen to get the computer to do what you want. See Rubin for more detail.

Download and save to the Desktop the file .tcshrc_3340

Found on the downloads page of the course home page.

`cd ~`	move to your home directory.
`cd Desktop`	move to Desktop directory
`cp .tcshrc_3340 ~/.tcshrc`	note the spelling !!
`echo /bin/tcsh >> ~/.bashrc`	Technical mumbo jumbo for now.
`cd ~`	move to your home directory.
`/bin/bash .bashrc`	do NOT forget the . (This will automatic from now on.)
Q: `set`	That's it. Tell me what you see.

Reminder: Important Home Directory Files (2)

Linux allows you to configure your "working environment."

The working environment is roughly the look/feel of your login session as well as values of important variables that affect command execution.

- Two Important home directory files: .login and .tcshrc

  .login usually contains commands and "aliases" (abbreviations).

  NB: The . is crucial!

Executed once per login session. (It turns out…. You don't have one!?)

~/.tcshrc is the other important file. Note the leading period . in the file name.

~/.tcshrc is executed every time you open a window/terminal.

`cat .tcshrc` command to list contents of non-directory file. Important!

Supposed to stand for concatenate. (Seems a bit obscure.)

Reminder: ~/.tchsrc file

Look at **your** .tcshrc file. (Try it!) What do you see? What is its structure?

```
set path = (. ~/bin /usr/local/bin /bin … )
```

Sets the important "shell" variable `path` to show what directories are searched for to execute a keyboard command **and** the search order.

```
set cdpath = (. .. ~ )
```

Set the important "shell" variable `cdpath` to show what directories, and their order, searched in to execute, eg., `cd some_stupid_dir`
If `some_stupid_dir` not in . or .. or ~ , an error msg is displayed.

`set noclobber` shell variable acting as switch. More later.

~/.tchsrc file (2)

Look some more at .tcshrc

```
setenv PRINTER hp2100
```

Sets the important "environmental" variable `PRINTER` to hp2100

hp2100 is the name of a printer (actually a printer "queue." ignore difference for now). Note the absence of the = sign. By convention, uppercase.

- As a practical matter, difference between shell and environmental variables is not completely crisp. Don't get too excited about difference.

Shell variables used for instances of a shell. Conventionally, lowercase.

Environmental variables used for other situations. (ie, set printer for acrobat reader to use when printing.) Conventionally, uppercase.

As with all conventions, not always followed. (Who knew?)

~/.tchsrc file (3)

Almost done with .tcshrc

```
alias h history
```

`alias` makes an abbreviation for some command, to save typing.
Here, h is the abbreviation and `history` is the real Linux command.

Q: What does `history` actually do? C'mon, just try it!

Hint: Use an alias to shorten a command you use often (or misspell !).

Important (and useful) technical detail:

Linux has 3 kinds of quotation marks: “, ‘ and `.
They are NOT the identical in functionality. (Life is hard.)

~/.tchsrc file (4)

Useful tip: Suppose you modify .tcshrc   How to effect the changes?
(You could … logout/login. Too inelegant for us)

```
source .tcshrc
```

(Useful) command to execute commands/aliases in file that follows it.

Quotation Marks

Distinguish between the 2 commands: `echo “date”` & `echo `date``

Note the different quotation marks (nothing special about `echo` command)

Q: Firstly, what does `echo` actually do? C’mon, just try it!

E.g., `echo ponyland`

Q: Secondly, what does Linux command `date` actually do?

Q: Now try
```
echo “date”
echo `date`
```

Use “ “ to quote text strings (has other uses also)

Use ` ` to enclose a command whose output you want

Setting variables

Sometimes useful to "set variables" in "t-shell." See ~/.tcshrc for examples.

Technique is simple: set your_variable = a character string or numbers

E.g., `set today = date`	
Examine the contents of variable `today` :	`echo $today`
Q: Not what you want? Need proper quoting:	``set today = `date` ``
Now do	`echo $today`
Q: How to see what variables are set?	`set`
Q: How to delete set variables?	`unset`

Use " " to quote text strings (has other uses also)

Use ` ` to enclose a command whose output you want

Redirection

Useful to direct command output to a file (rather than the screen).

Q: E.g., what does `cal` do?

Q: Try this: `cal > jan09.txt` (Note the ">" symbol.)

Examine jan08.txt w/ `cat jan09.txt`

Q: Try this: `cal 1 2009 > jan09.txt` What happens? Why?

A: see `set noclcobber` in ~/.tcshrc

Q: Now try this: `cal 2 2009 >! jan09.txt` NB >! What happens?

Q: And this this: `cal 2 2009 >> jan09.txt` NB >> What happens?

command > silly_file	command output written to new file.
command >! silly_file	same as above, overwrites contents, if any.
command >> silly_file	appends output to file.
command >& silly_file	output and error msgs to file, no overwrite
command >>& silly_file	appends output and error msg to file.

(Really) Simple Editing

We need to know how to write/edit text files.

<u>Numerous</u> ways to do this.

Start simply. Get fancy later.

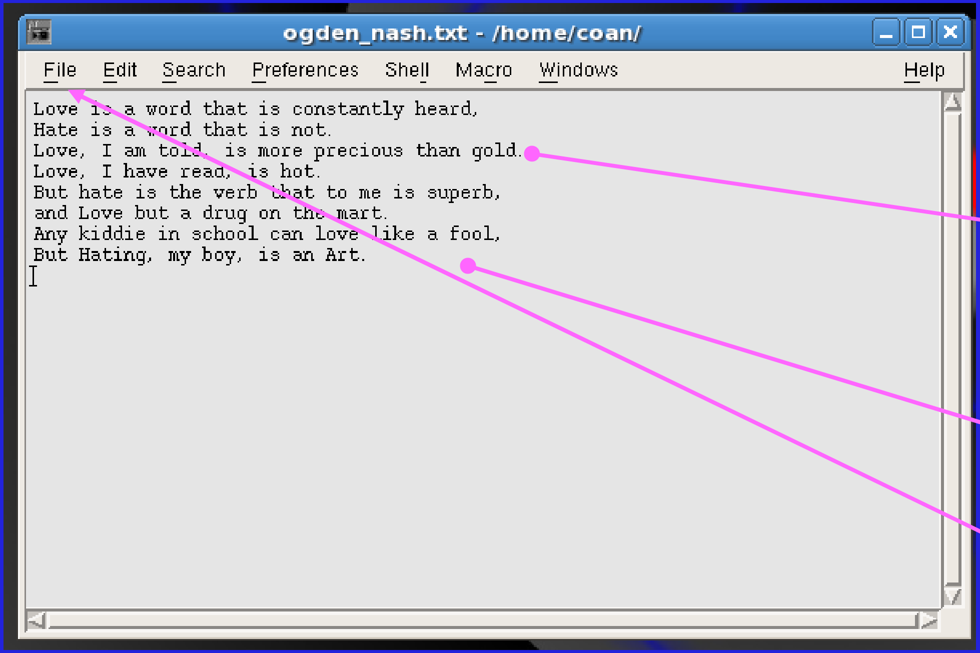
Issue command: `gedit junk1.txt &`

File name of your choosing

Linux cmd

Tells linux to run command "in the background."

Frees window for use. A good idea. See tutorial #5.



```
Love is a word that is constantly heard,
Hate is a word that is not.
Love, I am told, is more precious than gold.
Love, I have read, is hot.
But hate is the verb that to me is superb,
and Love but a drug on the mart.
Any kiddie in school can love like a fool,
But Hating, my boy, is an Art.
```

Can enter text by typing

Can also cut and paste

Select "Unix" option when saving

Printing (i.e., who needs trees?)

Printing is a bit complicated in Linux (I don't know why.)

Files intended for printing have various formats: plain text, pdf, postscript, …

Because we are sophisticates, we will be fancy (i.e., we like fancy output.)

Postscript looks the best when printed. (Warning: MPO.)

Linux command

Cmd option "B" for "no headers" (Why "B"? dunno.)

Cmd option "p" for output file.

```
enscript -B -p ogden_nash.ps ogden_nash.txt
```

To-be-created output file.

Existing input file to be converted. Not deleted.

`man enscript` Gives all options. Many.

Printing (2)

Hey buster, printing means paper output. Show me the paper.

Cmd option “P” for “printer queue”

```
lpr -Php2100 ogden_nash.ps
```

Name of file to be printed.

Linux command

Specific name for printer queue.

- Can also print directly via `enscript`
- Use –P option. See man page.

FYI: can create pdf files from ps files:

```
ps2pdf ogden_nash.ps
```

Creates `ogden_nash.pdf` automagically.

Reminder: Linux tutorial

One stop shopping for a decent Linux online tutorial:

http://www.ee.surrey.ac.uk/Teaching/Unix/index.html

Says “Unix” but ok for Linux.

Repeats/expands on what is said here.

Adds additional info.

Easily digestible (key features, non-exhaustive).

Read “.cshrc” as “.tchsrc” .

- ➢ Read tutorial at home (# 7 is a bit much for now)

- ➢ Useful to skim through glossary (app B) in Rubin et al.

Summary

.login and .tcshrc are important home directory files.

Relevant only for TC shell (which is what we are using).

Know how to set variables.

Distinguish quotation marks: “ v. ‘ v. `

Output redirection: > or >> or >!

nedit is a simple-to-use text editor. (We get fancy later.)

`enscript` and `lpr` used for printing. Know your printer queue name.

➢ Read (and perform!) Linux tutorial

Don’t suffer in silence. Scream for help!!!