1.1 role of computing in cs&e

- user interface
  (GUI or command-line interface)
  for setting up input files to a simulation

- simulation

- analysis and visualization of output files
  (data collection and experiment control)
1.2 bash (a Unix shell)

"UNIX ... wasn’t designed as a commercial operating system meant to run application programs, but as a hacker’s toolset, by and for programmers."
file system

```
/ ← root
 / | /
 bin usr Users
  |   |
 moi ← home
 /     /
 todo.txt Desktop ← current, e.g.
```

path (specifies a file or directory)
absolute /Users/moi/todo.txt or ~/todo.txt
relative ../todo.txt  . . denotes parent
commands

$ # syntax: <cmd> <arg1> ... <argn>
$ ls ~
$ # there are hidden files & directories,
$ # e.g., ~/.profile
$ ls ~ -a # all
$ # by convention, hyphen denote option (flag)

$ grep Solution 1.txt 2.txt 3.txt
$ grep Solution [123].txt # globbing
$ # specifies a set of files/directories
more about bash than you want to know because what you do not know can hurt you

((DEMO))
Command line editing

^b ^f ^n ^p  ↔  arrow keys
^r <str> recall previous entry that matches <str>
ESC f ahead one word
ESC b back one word
^a to start
^e to end
^d delete
del / ^h delete previous
^k kill text that follows
^u kill text that precedes
^w kill preceding word
^y restore killed text
^t transpose
ESC t transpose words
1.2 review

navigating: pwd, ls, cd
options/flags:  −<keyword> <value>
   e.g., ls −F
print: echo
glob:
   pattern for specifying a set of "file" names
      *  ? [<list of chars>]  −
modifying: touch, rm, mv, cp, mkdir, rmdir
<name>=<string>  # might need (single) quotes
export <name>=<string>

$ touch a b c
$ for x in *
> do
>   echo $x
> done
1.2 review (contd)

$ grep SSN *.txt # what does this do?

$ zip <name> <list> # creates <name>.zip
$ unzip <name>.zip

child shell:
$ bash # hides non-exported variable values
$ <command>
$ ...
$ <command>
$ exit
1.2 review (contd)

job control: \^Z, &, bg, fg, jobs, kill
i/o redirection:
  < input, > output, 2> error, | pipe
e.g.,
$ man gcc > tmp
$ grep verbose tmp
equivalent to
$ man gcc | grep verbose

$ echo $PATH
/usr/local/bin:/Users/skeel/anaconda/bin:.:/bin
$ # colon-separated list of directory paths
$ # where system seeks command names
1.2.1–1.2.4: special topics

1.2.1 text editors
   emacs, vi, notepad, TextEdit
   \[\rightarrow \text{use plain text} \leftarrow\]

1.2.2 integrated development environments

1.2.3 shell scripts
   permissions
   arguments
   .profile and .bashrc \(rc='\text{run commands}'\)

1.2.4 unix tools
   e.g., awk edits files, especially data text files

((DEMO))
1.2.3 shell scripts review

```
#!/bin/bash  # thrice.sh
echo $1 $1 $1

#!/bin/bash  # to_txt.sh
for x
do
    cp $x $x.txt
done
```

```
$ chmod 700 thrice.sh  # change mode
# 7 = 4 + 2 + 1
#     read write execute
#     permissions
$ ./thrice.sh No!
```
1.2.3 shell scripts review

::: ~/.profile :::

```
export PATH=.:$PATH
...
source .bashrc
echo executed .profile
```

::: ~/.bashrc :::

```
alias mv='mv -i'
alias cp='cp -i'
alias e=emacs
...
echo executed .bashrc
```
1.3 installing the software

1. binaries
2. from source
   - unpack
   - configure
   - build
   - install (may need to use sudo)
   - test
3. using a package manager
   automates installation from source

    OS X: instructor; Windows, Linux: TA
    ((NOTES))
Please do not contact RCAC for routine information.

*Note.* A Unix text file is one long string of characters. The newline character `\n` marks end of each line. It is good practice to have `\n` at very end of file. (There is no end-of-file character.)
When doing cut and paste, be conscious of `\n`.

((DEMO: ssh -Y you@scholar.rcac.purdue.edu))

use `-Y` only for graphics or GUIs
1.4 review

ssh you@scholar.rcac.purdue.edu
module load python
scp