# UCF Physics PHZ 3150: Introduction to Numerical Computing

## Learning the Unix Command Line ("Shell")

The GUI interfaces available on Unix/Linux, Windows, and Macintosh platforms can effectively support research, up to a point. That point is the need to do complex file manipulations that do not exist in a GUI, such as sorting all the files that contain a certain text pattern according to file creation time. Doing things nobody thought of before is the basis of research, but if nobody thought of it, it's not in a GUI.

Command-line interfaces excel here, since you can program them easily. Unix has the best of these, and is free, open-source, and very well supported. GUI software programs come and go over the years, but solutions built on command-line interfaces are stable for decades.

Here are some tips for learning the Unix command line, or "shell".

1. Know and use your documentation!

   (a) Use Google, read a tutorial, or watch a video online:
   `http://www.doc.ic.ac.uk/~wjk/UnixIntro/index.html`
   `http://linuxcommand.org/`

   (b) There are "manual pages" accessible via the `man` command. The job of these is not to teach, but rather to give a summary of a command's behavior and all the command line options, as a reference. For example, if you know that `ls` lists files but you want to list them in the order they were created rather than alphabetically, `man ls` is your friend.

   (c) There is often additional documentation from the authors of a program in the directory `/usr/share/doc`, or on web sites devoted to the program.

   (d) Some programs have an "info" entry. Info pages have both tutorial and reference sections. You can read info from inside `emacs`, or by typing `info`. The `emacs` interface is better since you can use the mouse. Type `q` to quit.

   (e) If you are interested in a new application or task, see if there is a Linux HOWTO on it. These are docs written by users and for users. `http://tldp.org/HOWTO/`. In particular, check out `Emacs-Beginner-HOWTO.html`.

   (f) If you are on a Linux machine, explore the web site for your distribution, such as http://help.ubuntu.com.

2. Play and experiment. Don't memorize; solve. Start with logging in, clicking around the Dash (it's the main thing that differs between flavors of Linux) and menus, and starting browser and terminal windows. Read or watch a tutorial online. Look at the files and directories on the disk. Make some, move them around, rename them, copy them, delete them. Most things have been done before, so ask around, borrow, modify, adapt (with permission, in line with syllabus rules, etc.). Stay curious and ask why.

3. Learn to edit ASCII text in `emacs`. While there are other text editors, `emacs` is by far the most powerful: if you run your code under `emacs`, `emacs` will help you debug. Take a day to learn it. There is a tutorial within `emacs`, a HOWTO, and tons online. Keyboard

shortcuts make it even faster; make a list of the most useful ones and get used to using them. You'll never go back!

4. Cutting and pasting in Unix's X Window System is even easier than on Mac and Windows: just highlight the text with the left mouse button and it's copied to the mouse buffer. Put the cursor where you want it and click the middle button, and it's pasted. If the selection is long, you can highlight the top of it, scroll to the end, click the right button, and this will extend the selection to where you click the right button. No more ˆC ˆV nonsense. It's very fast.

5. What scares most people about Unix is its reputation for having a steep learning curve. These days, you can do all the normal stuff with a GUI, reserving command lines for things that you couldn't do at all otherwise (e.g., in Windows). Simple commands are no trouble, but passing data between several commands can look ugly. It's much easier to learn the command language and invent what you need on the fly than to memorize a huge number of complicated command lines. When you get to the point of needing them, take the few hours needed to learn how to do it, and you will be much happier in the long run.

6. There is a decent office suite called LibreOffice (variants are StarOffice and OpenOffice) on the panel (or type `loffice`). It has a spreadsheet, a drawing program, a presentation program, and a word processor. The word processor reads and writes MSWord files, which is its main use by Linuxheads. If you really must use MSWord, LibreOffice can make PDF files for handing in. Just read in your MSWord file and print to file. Check to be sure that LibreOffice got your file right before handing it in.

7. Make a crib sheet that lists the commands you use and what they do. Here's a start:

| | |
|---|---|
| `ls` | `lists files` |
| `cd` | `change directory` |
| `cat` | `print file on screen all at once` |
| `cp` | `copy a file, use -a option to preserve date` |
| `mv` | `move (rename) a file` |
| `less` | `print file on screen a page at a time` |
| `rm` | `remove file, FOREVER! -i option confirms.` |
| `mkdir` | `make directory` |
| `rmdir` | `remove directory (remove files in it first)` |
| `file` | `analyze file contents and give type` |
| `emacs &` | `edit text (or click in panel)` |
| `gnome-terminal &` | `terminal window (or click in panel)` |
| `python3` | `Python interpreter` |
| `ipython3` | `interactive Python interpreter (nicer)` |
| `locate` | `locate disk files by name (not on all Unices)` |
| `xdg-open` | `open any file (not on all Unices)` |
| `passwd` | `change login password` |

8. Learn to use functions in your `~/.bashrc` file to abbreviate common actions, such as:

```
o () { xdg-open "$@"; }
```

See also:
`http://files.fosswire.com/2007/08/fwunixref.pdf`
`http://store.xkcd.com/products/linux-cheat-shirt`