Princeton University
Department of Operations Research
and Financial Engineering

ORF 201
Computer Methods in Problem Solving

Lab 1: Introduction to Unix and Editing

There is nothing to turn in for this lab

---

1. INSTRUCTIONS

In this lab, you will learn how to use the Silicon Graphics workstations to create, edit, and manipulate files. It is important that the basic concepts are well understood now so that you can concentrate on your programming assignments later.

There is *nothing to turn in* for this lab.

In handouts, we will use the `typewriter font` to indicate things that you will type into the computer as well as responses you may receive back from the computer. If something is in brackets (such as `[Enter]`), this indicates that you should press that key on the keyboard.

2. THE SILICON GRAPHICS WORKSTATIONS

The lab classroom, E-423 E-Quad, contains approximately twenty Silicon Graphics workstations. You will be doing your labs on these machines. Later, you will learn how to access these machines remotely but, for now, it is best simply to come to this classroom to work.

3. LOGGING IN

At a workstation terminal, you will see a screen, a keyboard and a small device called a *mouse* sitting on a vinyl pad, called the mouse pad. The mouse is used to move a pointer around on the screen. The use of the mouse will be described in detail later. If your screen is blank, hit the [Shift] key and you should see a prompt that looks like

Login:
Password:

You should note that you can work on *any* of the workstations in the classroom. Any of your work can be accessed from any of the workstations. In fact, your work can also be accessed from your room using DormNet (more on that later). After you see the `Login:` prompt, you should type your UserID and then press [Enter]. Your UserID is listed in the campus telephone directory.

After you have typed in your UserID, you need to give your password. Your password is your phone access code (PAC) or the last eight digits of your social security number.

If you obtained a message that says `Login Incorrect`, then you have either typed your UserID or password incorrectly. Try to log in again, and if you still have problems, notify your lab instructor.

If your login attempt was successful, the computer screen should go blank and then two *windows* will appear, one in the upper left corner (titled Desk 1) and the other in the lower left corner. For the moment, we shall be concerned mainly with the window in the lower left corner. This window is like a virtual piece of paper that you use to give commands to the computer. Before you can interact with this window, you must "position" yourself in it. To do this, grab the mouse and move it on the mouse pad until the red pointer on the screen is inside the window.

Now that you are in the window, you will notice that you are being asked if you want to read various system information messages. To ignore them (a good idea), simply type `q` and press [Enter].

4. Basic Unix Commands

The computer prompts you for Unix commands using a prompt that looks something like:

myname>

where `myname` is your UserId. The > symbol indicates that the computer is ready to receive your next command.

Modern computers store vast amounts of information. To organize this information effectively, it is necessary to have a systematic storage scheme (analogous to file cabinets in an office). On Unix systems, information is stored in a *directory tree*. The *root node* of the tree is called `/`. All information, including all the files of every student at the university, hangs from this root directory. In particular, your files hang from a particular branch of the tree, `/u/myname/`, which is called your home directory. The slash `/` at the end is used to indicate that the name is a directory name, rather than a file name (the `/` is actually not part of the directory name).

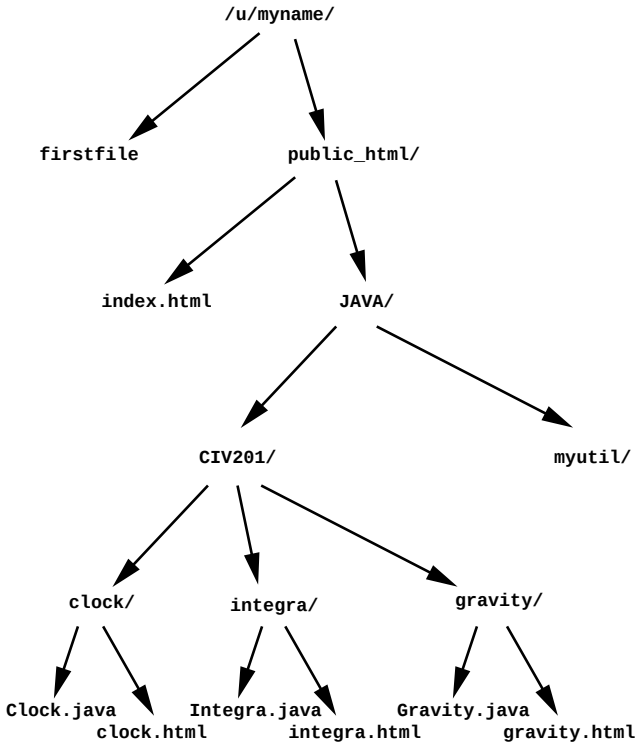

FIGURE 1. A directory tree.

Each directory contains two types of things (i) *files*, and (ii) *sub-directories*. A sub-directory is simply another directory that is a "child" of the "parent" directory. For example, in Figure 1 `public_html` is a sub-directory of the directory `myname` (your home directory), and `index.html` is a file in the directory `public_html`. Thus, `public_html` is a child of `myname`, and `myname` is the parent directory of `public_html`. Similarly, `clock`, `integra`, etc., are sub-directories of the `CIV201` directory. The directory `clock` contains the file `Clock.java` and `clock.html`.

Directories establish the organizational hierarchy of the file system, whereas data is stored in the files. In terms of the tree structure, files are always leaves of the tree, whereas directories are usually not leaves.

Now that you are logged in, you are "connected" to your home directory, which should be named something like `/n/homeserver/p/myname`, where `myname` is your UserID. To see the name of your home directory, you can type

pwd [Enter]

The letters `pwd` stand for "print working directory." To see the contents of that directory, you can type

ls [Enter]

The letters `ls` are a mnemonic for "list files." Unless you've done other computer work here at Princeton, you currently have no files in your home directory. We will now create one by copying over a file from the ORF 201 common directory. Do this by typing:

cp /u/orf201/lab1/firstfile . [Enter]

Note the period! Also, the spaces are important. This is a general copying command. The letters `cp` stand for "copy". If you now type `ls`, you will see that `firstfile` is in your directory. The general form of the copying command is

cp source_file destination [Enter]

In the example above, you copied a file named `firstfile` from the directory `lab1`, which is a subdirectory of `orf201`, which is a subdirectory of `u`. You copied it to your current working directory. A shorthand for that directory is a lone period. If you now type

cp firstfile firstfilecopy [Enter]

you will make a copy of the file. After you do this, you will have two copies of the same file. Note that the destination can be either a directory or a file. If it is a directory, then the source file gets copied into the destination directory, whereas, if it is a file name, then the source file gets copied to that destination file.

It is clear that to type a command into the computer, you must always hit the [Enter]key in order to mark the end of the command. Henceforth, we will not use the [Enter]symbol to indicate the end of a command.

Note that you can use the `[BackSpace]` key to correct input on the current line being typed. The program that is picking up and interpretting the commands you type is called the *Unix shell*. It is case sensitive. The vast majority of the commands (and probably all of the commands you'll encounter) are all lower case. If for some reason everything you type appears in upper case, check to see whether the *Caps Lock* light on. If it is, press the `Caps Lock` so that it "unlocks."

Each line you type is a command for the Unix shell to do something. If the shell understands your command, it will execute the command and when finished it will ask you for another command (with the `myname>` symbol). If it does not understand the command, it will print a message indicating that you did something wrong. Sometimes these messages are hard to understand, but a little thought about the message and what you typed can lead you on the way to finding your error. Don't be afraid to use your brain!!! You're still smarter than the computer!

Note that the shell understands a large number of commands so that sometimes if you make a typing error you will still have typed a command that the shell understands but it's action may be very different from what you intended. Hence, type very carefully. Those of you who are familiar with PCs, I guess that's all of you, are used to being asked for confirmation before actually doing anything. This confirmation step provides a safety net against errors but it also slows you down

greatly. Unix treats you like an adult. If you say you want to do something it assumes that you mean it and it does it without question. So, again, be careful.

Short MS-Windows Tirade. By the way, for some reason Microsoft thinks that the answer to a yes/no question should always be either `Ok` or `Cancel`. This leads to very strange dialogs. Recently, I was working on a PC and wanted to cancel something that I was in the middle of. I clicked the appropriate button and the confirmation dialog box asked: "Are you sure you want to cancel? Ok or Cancel." It seems that clicking either Ok or Cancel should do the same thing but, of course, it doesn't. It is these things that make the Windows operating system profoundly annoying. Unix is harder to learn but everyone who does eventually realizes that it is worth the effort. It is the same as in literature. It is easy to read and even enjoy a cheap novel. Shakespeare, on the other hand, requires a real effort but it is worth it. *End of Tirade.*

For this course, it is required that you customize your operating environment slightly by typing the following command

```
source /u/orf201/bin/configure
```

In order to look at the contents of a file, such as `firstfile`, you can use the "more" command:

```
more firstfile
```

This command tells the computer to type out the contents of the file onto your screen, one "page" at a time. After each page, you will see the word `More` at the bottom of the screen with a number indicating the percentage of the file viewed. To see the next page, you must utter to yourself "Please sir, I want some more" and then hit the space bar. After hitting the space bar, the next page will be displayed. Note that the contents of `firstfile` are a short description of some Unix commands. If you get tired before viewing the whole file, you can quit out of "more" by pressing the q-key.

Now let's make some directories. Our aim is to begin setting up a tree structure similar to the one shown in Figure 1. Let's create a subdirectory called `public_html`. To do this, type

```
mkdir public_html
```

The letters `mkdir` stand for "make directory." Now do an `ls`. You should see what appears to be a file named `public_html`. What happens if you try to look at the contents of the file `public_html` using `more`? Try it and see.

Now suppose you want to move your file named `firstfile` into the directory named `public_html`. To do this, type

```
mv firstfile public_html
```

The letters `mv` are a mnemonic for “move.” If you now type `ls`, you will not see the file named `firstfile`. In order to make it easy to work with files in your new directory, you should move yourself into it. This is done by typing

cd public_html

If you now type `pwd`, you will see that your current working directory is something like

/u/myname/public_html

If you type `ls`, you will see that the file `firstfile` has found its new home.

While we’re working with directories, it is convenient to know a shorthand for the parent of your current working directory. For example, if you were currently working in the directory `/u/myname/public_html`, the parent directory would be `/u/myname`. If you want to connect to that parent directory, you can type

cd ..

The two periods are essential. They form the shorthand for “parent of my current working directory.” After typing this command, you should be connected to your home directory.

By the way, sometimes you might forget where a file is stored. For example, if you’ve forgotten where `firstfile` is you can type

find /u/myname -name firstfile -print

Go ahead and try it. The `find` command can be used even when you don’t exactly remember the file name. For example, to find all files with the “first” as part of the name, type

find /u/myname -name ”*first*” -print

Oh and another thing, `$HOME` is a convenient shorthand for your home directory, `/u/myname`, so you could also type

find $HOME -name ”*first*” -print

Now it is time to learn about how to erase a file. If you type

rm firstfilecopy

you will *remove* the file from your directory. It is gone for good, so use the `rm` command carefully. If you now do an `ls`, you will see that `firstfilecopy` has disappeared.

Finally, we would like you to place all of your ORF 201 work in one particular directory. To do this, type the command

```
mkdir -p public_html/JAVA/ORF201
```

The "-p" is a flag telling `mkdir` to create not only `ORF201` but all other directories along the "path" from where you are to there, which in this case means to create the directory `JAVA` too.

4.1. **You Say Csh, I Say Tcsh.** There are several Unix shells. When CIT created an account for you they probably assigned you `csh`. However, `tcsh` is more modern and offers some nice typing shortcuts. Therefore, you should type

```
chsh tcsh
```

to change your shell to `tcsh`. The change will not take effect until you log out and back in.

5. Editing Files

What does it mean to edit a file? Imagine having a piece of paper that you have typed up. In order to make changes, you might delete some of the text, add new material and cut out one paragraph and paste it in another place. Editors on a computer allow you to do these manipulations without paper and pencil. You do them with your keyboard and mouse.

On Unix Systems, there are several popular editors that one can use. In this course we will only talk about one of them, `xemacs`. You must learn xemacs. This editor takes a little time to learn (it is admittedly cryptic), but it is widely available and has several useful features. Thus, if you are working on this course on a PC in your dorm room or elsewhere, you can use `xemacs` to edit files on the PC. Then you can use your PC as a "dumb terminal" to first send the file to your ORF201 account using `DormNet` and then run and modify your code using the PC.

Xemacs has a nice tutorial built in. To do the tutorial, first fire up xemacs by typing

```
xemacs
```

and then type

```
[CTRL]-H t
```

([CTRL]-H refers to simultaneous pressing of the control key and the H key).

6. Practice Your Editing!

In order to get familiar with editing files, you should create a file of your own that is a summary of the Unix commands described in this document. Create a file with a short description of each command for your own reference. Make the file short so that the information can appear on one page (50 lines maximum).

7. Printing Files

When you have finished for the day, you may want to print out on the printer a file you have created. This is done by typing

```
civprint filename
```

which will print out the file called `filename` on the laser printer in E-417 or E-423. Please don't print your files too much. The printer is meant for "normal" use, which means you shouldn't print out your files too often. You will need to print your files occasionally so that you can study at home the programs you create in the classroom. Also, note that the `civprint` command is a special command that we have created for your use in ORF 201 and will not work on other Unix machines elsewhere.

8. Logging Out

When you are finished, you must tell the computer that you are leaving. This is called *Logging Out*. To do this, place the mouse over any part of the background graphics screen and click the right-mouse button. Click the right mouse button after moving the pointer over the choice `LogOut`. The computer will ask you to confirm this choice. Do so by clicking the right-mouse button over the choice for `Yes`.

IT IS VERY IMPORTANT THAT YOU LOG OUT BEFORE YOU LEAVE THE TERMINAL. OTHERWISE, SOME SHIFTY INDIVIDUAL CAN DESTROY ALL OF YOUR WORK. MAKE SURE YOU LOG OUT!!!!