Basic UNIX commands*

Please remember that UNIX is **case-sensitive**.

Files

- ls --- lists your files

ls -l --- lists your files in 'long format', which contains lots of useful information, e.g. the exact size of the file, who owns the file and who has the right to look at it, and when it was last modified.
ls -a --- lists all files, including the ones whose filenames begin in a dot, which you do not always want to see.
There are many more options, for example to list files by size, by date, recursively etc.

- mv filename1 filename2 --- moves a file (i.e. gives it a different name, or moves it into a different directory (see below)
- cp filename1 filename2 --- copies a file
- rm filename --- removes a file. It is wise to use the option rm -i, which will ask you for confirmation before actually deleting anything. You can make this your default by making an alias in your .cshrc file.
- diff filename1 filename2 --- compares files, and shows where they differ
- wc filename --- tells you how many lines, words, and characters there are in a file

Directories

Directories, like folders on a Macintosh, are used to group files together in a hierarchical structure.

- mkdir dirname --- make a new directory
- cd dirname --- change directory. You basically 'go' to another directory, and you will see the files in that directory when you do 'ls'. You always start out in your 'home directory', and you can get back there by typing 'cd' without arguments. 'cd ..' will get you one level up from your current position. You don't have to walk along step by step - you can make big leaps or avoid walking around by specifying pathnames.
- pwd --- tells you where you currently are.

Compiling/Running/Debugging Programs

- g++ <list of .cc file names> -- to **compile and link** the source codes listed, and generate an executable file called a.out. You can specify the executable file name to something else using –o option. For example, to generate an executable named myProg, use the following command:

```
g++ -o myProg <list of .cc file names>
```

It's highly likely that there are syntax (grammar) errors in the code. g++ will reports syntax errors in the code, you should correct them starting from the first error message, and then the second, and so on. Sometimes, multiple error messages can be caused by the same mistake (such as forgetting to declare a variable, mistype a variable name, etc). Therefore, after you correct some of the mistakes, you can try to compile again and work on correcting remaining errors.

To compile your code so that it can be debugged using gdb command, you should use –g option:

```
g++ -o myProg –g <list of .cc file names>
```

- **Run your program**. If g++ successfully compile your code, it creates an executable file (named a.out by default, see above bulletin for how to specify a different name) in your current directory. To run the program, just type:

  ```
  ./a.out  or
  ./myProg
  ```

Note here we needs to have "./" so that the terminal (shell) knows that the program is located in the current directory.

- **gdb <executable_file_name>:** debug the program. gdb allows you to trace the program (i.e., run the program line by line), examine the core dump (a file recording the status of the program when it encounters a fatal run time errors), etc.

***Part of the document is based on Dr. Li's Unix command references.**