CS 170-001: Introduction to Computer Science I
Emory University, Math/CS Dept. September 6, 2011

Useful Unix Utilities

When using a command prompt, thousands of programs become available. However, the average person uses only a handful of them on a regular basis. The most commonly used programs are those which are used for navigating the filesystem and manipulating files.

`pwd`	print your current working directory
`cd` *directory*	change the directory that you are working in
`ls`	list contents of the current directory
`mv` *source destination*	moves a file from the location to the destination
`cp` *source destination*	copies a file from the location to the destination
`rm` *filename*	removes a file
`mkdir` *filename*	creates a directory
`rmdir` *filename*	removes an empty directory
`cat` *filename*	output the contents of a file
`wc` *filename*	count the number of lines, words, and characters in a file
`chmod` *permissions filename*	change file access permissions
`man` *program*	prints a manual describing the functionality of a program

Unix Shortcuts

One common way to save time while navigating a Unix filesystem is to use *relative* pathnames as opposed to *absolute* pathnames. Absolute pathnames are the complete location of a file or directory and always start with a `/` (slash). An example of an absolute pathname is `/home/cs170001/lab1`. Relative pathnames are pathnames expressed in relation to the current directory. From your home directory, some relative pathnames are `cs170`, `share`, and `priv`. Each directory also has two special relative pathnames `./` and `../` which refer to the current directory and the parent directory, respectively.

The tilde character can also be used to save time while navigating the Unix filesystem. If you prefix a filename with `~/`, the tilde is replaced with the pathname for your home directory. Thus, `cd ~/cs170` will take you to `/home/`<*EmoryID*>`/cs170` directory. If you prefix a filename with `~`, it will be replaced with `/home/`. This makes other home directories easily accessible. Thus, if you enter `cd ~cs170001/share` into the command prompt, it will take you to the class share folder.

The final Unix shortcut is the use of Unix wildcards. The most common wildcard is the asterisk character (`*`), which can be used to match multiple values. When entering Unix commands, this character can be inserted anywhere into a filename to match one or more files. The `*` wildcard comes in handy for programs which can process multiple files at once, such as `cp`, `mv`, `rm`, and `wc`. Here are some examples of commands using wildcards:

`cp *.txt ~/share/`	copies all text files in a directory to your share folder
`rm garbage*`	removes all files beginning with garbage
`wc *.java`	counts the lines, words, and characters of all java files in a directory

File permissions

In Unix filesystem defines three permissions for each file: read, write, execute. File access permissions are defined for owner of the file, group and other users. You can check file permissions using command `ls -l` and change them using command `chmod`.

Operations on permissions which use `chmod` have defined syntax. First you need to specify group of permissions you will change – owner user, group, others (use `u`, `g`, `o`, respectively). Then you need to specify if you would like to grant or clear some permissions (use `+` or `-`, respectively). Finally, you need to define which permissions will be changed: read, write, execute (use `r`, `w`, `x`, respectively). For example, if you want to add a permission for read and write for other users for file `filename.txt`, you may call command `chmod o+rw filename.txt`.

A permission to execute is used for programs (including scripts) and directories. This permission on a directory allows you to change current directory to one specified as parameter. (Which command will you use to change current directory?)