What is Open Source Operating System?

Open source refers to software that is created by a development community rather than a single vendor. Typically programmed by volunteers from many organizations, the source code of open source software is free and available to anyone who would like to use it or modify it for their own purposes. This allows an organization to add a feature itself rather than hope that the vendor of a proprietary product will implement its suggestion in a subsequent release.

Not Necessarily Free

Although open source is technically free, a lot of open source software is paid. For example, if a vendor wants to add open source code to its application, it must comply with the open source license and make all of its software publicly available even though the open source code is only a small part. As a result, many vendors pay for open source code, which allows them to keep their applications private and not reveal proprietary techniques.

Peer Review

Open source developers claim that a broad group of programmers produces a more useful and more bug-free product, the primary reason being that more people are constantly reviewing the code. This "peer review," where another programmer examines the code Of the original programmer, is a natural byproduct of open source. Peer review is an important safeguard against poorly written code. In addition, the greater number of contributors provides enhancements and refinements that would take a lot longer with proprietary software or perhaps never be added.
Vendors of proprietary software counter by saying that "too many cooks spoil the broth!" They say that having complete control over software ultimately results in better products.

Useful Products May Last Longer

A distinct advantage of open source software is that as long as there is even one devoted contributor, the program will continue to

be enhanced. In the commercial world, useful software may be abandoned if it does not generate sufficient profit compared to Other products.

Is there any difference between UNIX and Linux?

UNIX is an operating system created in the early days of computers. More recently, Linux was created as an open-source, freeware operating system. It is "UNIX-LIKE", meaning that it uses many UNIX constructs but also departs from traditional UNIX in many ways. Like UNIX, Linux is faster than many of the other commercially available operating systems. It appears to also be far more robust than any of the Microsoft products. Linux is being used in many time critical applications because of its speed. It is also used in many applications that need to maintain uptime because Linux, like UNIX, can run for months at a time without rebooting. While the typical method of solving Microsoft problems is to "reboot", that particular requirement does not seem to be appropriate in a Linux/Unix environment. While UNIX has created a windows-like work environment, Linux has improved greatly on that concept. Linux has become a real player in the consumer operating system market... and it's free. While you may want to pay for a Linux distribution, the actual code is free and you are allowed to load it on as many machines as you want. You can get Linux for free if you wish to load it across the internet.

What are distributions?

A Linux distribution, often simply distribution is a member of the Linux family of Unix-like operating systems comprised of the Linux kernel. Linux distributions take a variety of forms, from fullyfeatured desktop and server operating systems to minimal environments.

There are currently over three hundred Linux distribution projects in active development, constantly revising and improving their respective distributions. One can distinguish between commercially backed
distributions, such as Fedora Core (Red Hat), SUSE Linux (Novell), Ubuntu (Canonical Ltd.) and Mandriva Linux and

community distributions such as Debian, Open SUSE and Gentoo. Usually, the procedures for assembling and testing a distribution prior to release are more elaborate the bigger the user base for the distribution is.

The UNIX Operating System

The UNIX system is mainly composed of three different parts: the **kernel**, the **file system**, and the **shell**.

The **kernel** is that part of the system which manages the resources of whatever computer system it lives on, to keep track of the disks, tapes, printers, terminals, communication lines and any other devices.

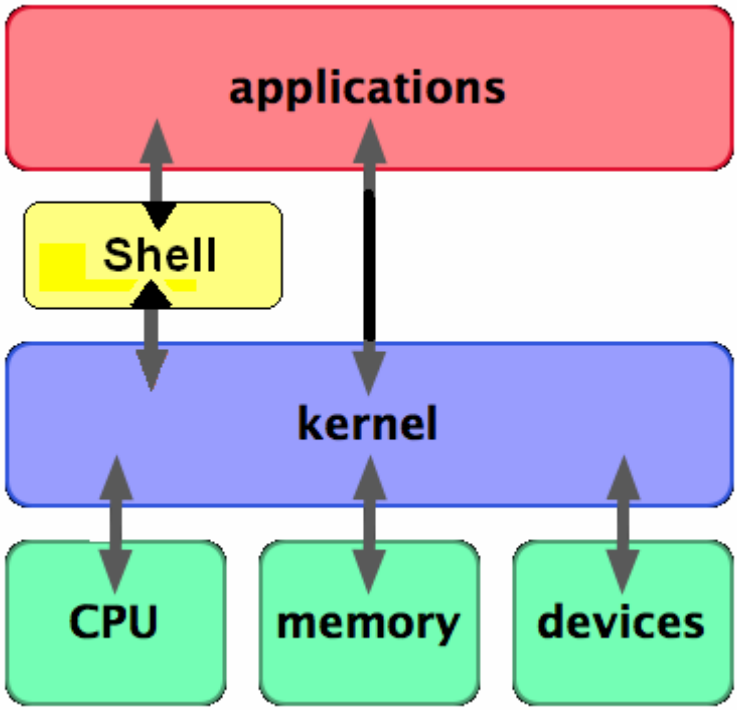
The **kernel** is the central module of an operating system. It is the part of the operating system that loads first, and it remains in main memory. Because it stays in memory, it is important for the kernel to be as small as possible while still providing all the essential services required by other parts of the operating system and applications. Typically, the kernel is responsible for memory management, process and task management, and disk management.

The **file system** is the organizing structure for data. The file system is perhaps the most important part of the Linux operating system. The file system goes beyond being a simple repository for data, and provides the means of organizing the layout of the data storage in complex ways.

The shell is the command interpreter. Although the shell is just a utility program, and is not properly a part of the system, it is the part that the user sees. The shell listens to your terminal and translates your requests into actions on the part of the kernel and the many utility programs.

One can imagine the Linux system as a series of three concentric circles, with the inner circle representing the kernel, the second circle representing the programming shell, and the last one

representing application programs. Figure 1.1 illustrates the relation between the kernel and other parts of computer.



The **shell** communicates to the **kernel**, and vice versa. The application programs can communicate directly with both the **shell** and the **kernel**.

Properties of Linux

Multi-tasking, Time Sharing

Linux is a multi-tasking operating system, which means that a number of programs can run at the same time. Those programs (called processes) can communicate with each other.
For example, a C program could be compiling as mail is being read or a file is being edited.
Processes that "wake-up" occasionally, and/or regularly, are called daemons. Daemons are used to send and receive mail, print documents, and so on.

Multi-user

UNIX is also multi-user: two, three, or more users are able to use the same processor to execute their programs.

Network Capabilities
Today's UNIX/Linux workstations come with TCP/IP and Ethernet connections. The Ethernet network connects dozens PCs together. From any of those nodes, it is simple to logon to a remote machine, send mail to a user on those machines, or transfer files to or from those nodes.

Portability
Linux can run on any hardware platform.

Flexibility
Linux is also a very flexible operating system, both for system administrators and users. Program names can be changed. Arguments to programs can also be changed. New programs can be built, thus allowing further customization of the system.

Software Available
Thousands of application packages are available for the UNIX/Linux System.

In addition to the commercial packages, many programs are written and made available in the public domain, and many other utilities/applications written by individuals and organizations, for the benefit of the "Open Source Community".
Linux is a perfect example of the incredible amount of software available, at no cost to individuals. In fact, most open source, freeware, or public domain packages used in the scientific world are developed and maintained on a Linux platform. The same can be said of any open source, freeware, or public domain packages developed on any UNIX variant: Linux is the platform of choice to develop/write software.

Virtual Memory
The UNIX operating system has virtual memory, or swap space, which means it, can run programs bigger than the amount of RAM the computer actually has! The amount of virtual memory is decided upon by the system administrator.

Case Sensitivity
The most common mistake for beginners involves the use of mixed case in UNIX/Linux commands: UNIX/Linux is case sensitive, i.e.,

"a" is different from "A".

Exercises

1. What is the operating system that covers the widest range of architectures? Why?
2. What are some of the advantages of using UNIX/Linux?
3. Name some of the reasons people would install and use Linux on their desktops or servers.

File System

The Linux File System manages and controls access to files and directories. It keeps track of opened and closed files, and manages files on the hard disk.

File and Directory Names

Conventions

Naming files is very simple: any ASCII character can be used. It is, however, recommended that no metacharacters (*, {,}, [,],? $, \, ~, >, <, |, &) be used. It is also recommended to simply use letters, digits, underscore (_), hyphen (-) and the dot (.).
By convention, files ending with .c are C language files.
File names beginning with a. (dot) are hidden files: they do not appear when files are listed. Those files are typically configuration files. Usually, their name consists of the package they represent.

Length

The maximum length of directory and file names varies from system to system. It depends on the file system type. For example Ext2 File System provides long file names. It uses variable length directory entries. The maximal file name size is 255 characters. This limit could be extended to 1012 if needed.

Basic Linux Commands

These are just a few of the most common and basic Linux commands that you may use in this course. On most systems more information about a command can be found by typing [man command]; man being taken from the word manual is the best tool for finding information about the usage of commands.
You will need to be root to use some of these commands; also be sure to check your path. If there is no path to the command then you will likely get a "command not found" error. Check the command to be sure you have typed it correctly. Be extremely careful as root, you can make your system unusable. This is very important to understand. If you are using a dual boot system you may not be able to access either system if you make a mistake as root and your system is not bootable. Before you type any command as root be absolutely certain of what you are doing.

- **help:** print help information regarding a command. Many shell-based utilities (not all) will print a help screen by adding the **--help** switch to the end of the command. For example:
  ls –help

- **info:** Provides information about a command.
  info mkdir

- **man:** Type man followed by a command to get detailed information about how to use the command. Ex:
  man ls
  Type **man -k** followed by a word to list all of the commands and descriptions that contain the word you specified. Ex:
  man -k user

- **apropos:** Search Help manual pages (man -k)

- **mkdir:** Make Directory. "mkdir /home/someuser/new" creates a directory named new in someuser's home directory.If you are currently in the directory you want to make the new directory in you can just do "mkdir new" to make a directory named "new".

- **ls:** List files in the current working directory except those starting with. And only show the file name.
  ls –al List all files in the current working directory in long listing format showing permissions, ownership, size, and time and date stamp.

- **rm:** The **rm** command is used to remove or delete files or directories.

  Ex: rm myfile

- **clear:** Clears the screen.

- **cat:** Sends file contents to standard output. This is a way to list the contents of short files to the screen.

- **more:** Type more followed by the name of a text file to read the file's contents. Why do we emphasize using this on a "text" file? Because most other types of files will look like garbage! Ex:
  more testfile.txt

- **less:** Similar to the more command, but the user can page up and down through the file. The example displays the contents of textfile.
  less textfile

- **cd:** Change directory cd /home changes the current working directory to /home. The '/' indicates relative to root, and no matter what directory you are in when you execute this command, the directory will be changed to "/home".

- **cd ..:** Move to the parent directory of the current directory. if you are currently inside your home directory then this command will make the current working directory "/home".

- **cd ~:** Move to the user's home directory which is "/home/username". The '~' indicates the users home directory.

- **pwd:** Shows the name of the current working directory, it's the abbreviation of "present working directory".

- **logout:** Logs the current user off the system.

- **cp:** Copys files, cp myfile yourfile will copy the files "myfile" to the file "yourfile" in the current working directory. This command will create the file "yourfile" if it doesn't exist. It will normally overwrite it without warning if it exists.

- **nano:** Typing nano will start a basic text editor on most Linux systems.
  Type nano followed by the filename you wish to edit. This basic editor is quick and easy to use for beginners. However, it is very important that you also learn about other text editors available on Linux and UNIX systems.

- **Passwd:** followed by a user name will change this user's password. Example: **passwd** chap you can perform this command only when you have root permissions. If you only type passwd this will change the currently logged in user's password.

- **useradd:** followed by a username this command will add a user to the system. Same as adduser command.

- **The su command**

  Many a times you might have logged in as a normal used and might need to be root to install a software or for some other small task. You could logout then login as root complete the work logout and login back as a normal user. Instead, you can just use the su command. The format is:
  Ex: su

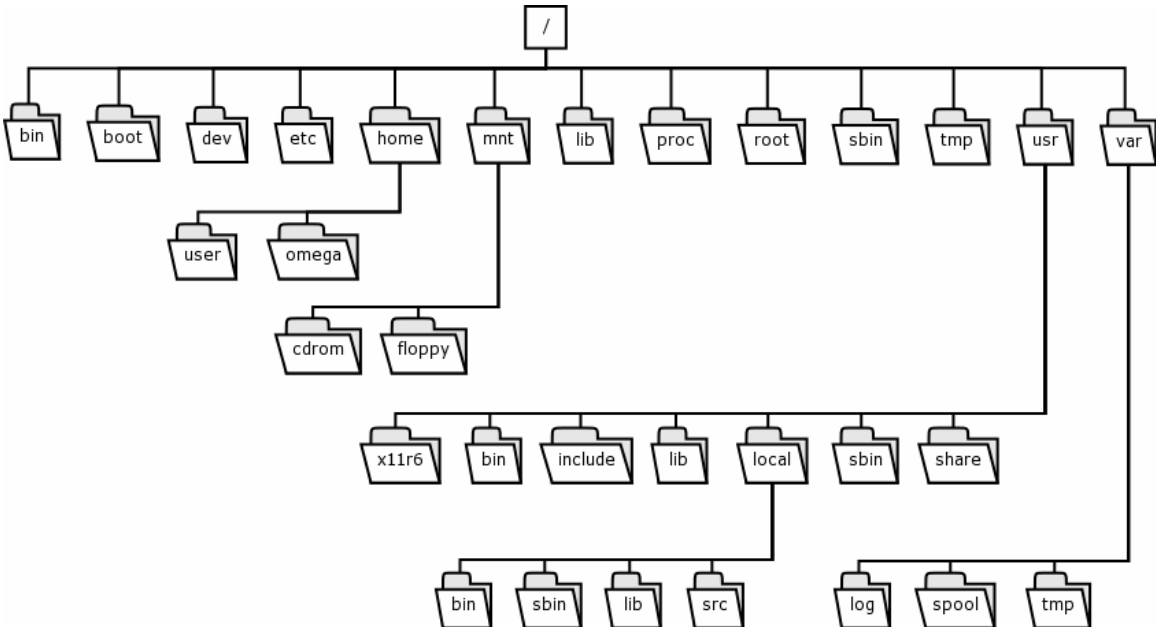
  This command lets you become root or "Super User" after you type it and hit enter you will be prompted for the root password, enter the correct password and you are root. Be careful you now have the power to destroy your system.

To get back to your own account do: [exit]
you can 'su' to become root from a normal user, But if you are root, you can use 'su' to become any user without using a password. Once your work is finished,
Ex: su username

Use 'exit' to become yourself.

Structure of Directories, Files

The filing structure of the UNIX/Linux operating system is hierarchical. It is a tree structured system, completely open (assuming necessary permissions) to every user on the system, with everything emerging from / (root) at the top.
Figure 1.2 illustrates this concept.



Every directory has a parent, and--possibly--one or more children. Those children can in turn be parents.

A directory is a special type of file. A file and a directory of the same name within the same directory is therefore impossible.

Everything in UNIX is considered a file; a directory is a special kind of file, and so is the keyboard (/dev/kbd).

Linux Directories

/ [Root]

/ by itself is the root or the top, the beginning of the file system. "Super user" is usually the only user allowed to write to that directory.
This is the home directory for the super user (root). This directory is not viewable from user accounts. The /root directory usually contains system administration files.

/bin [Commands]

The /bin and /usr/bin directories contain public commands. Those commands may be in binary, or in shell format.

/boot

Stored in this directory are files that are required for the Linux boot process. Such files include vmlinuz, the Linux kernel file.

/dev [Devices]

/dev contains device files required for interfacing with hardware. Devices in UNIX are either block or character devices. Examples of character devices are your keyboard, mouse and serial port. Block devices can include the floppy drive, CD-ROM drive and hard disk. In UNIX, every device is referred to as a file. For writing to even the keyboard writes to a file (/dev/kbd) which is read by the operating system.

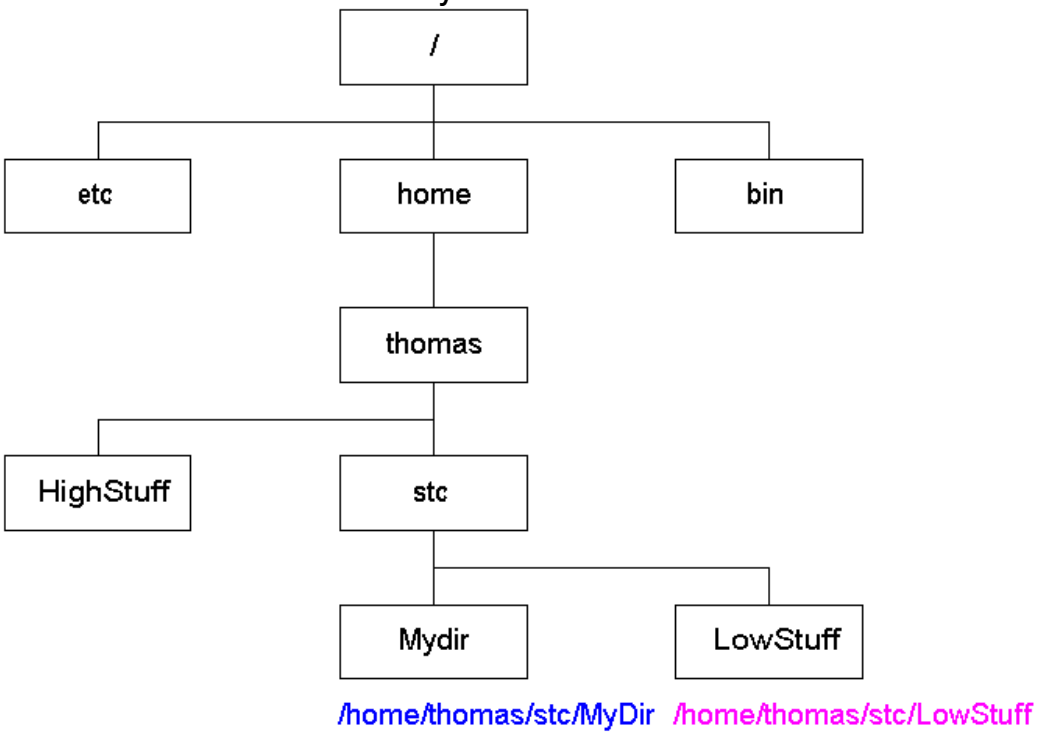
/etc [Management]

The /etc directory is the "management" directory. /etc contains programs used by the system administrator to administer and configure the system to the needs of the users.
It contains configuration files which are local to the machine.

Programs store configuration files in this directory and these files are referenced when programs are run.
One of the most important files of the /etc directory is the passwd (pronounced "password") file. The passwd file contains basic information about each user's account, including the logon and, in some cases, the password (encrypted) of every user (in the other cases, the encrypted passwords are in a file called /etc/shadow which only the super user has access).

/home [Home Directories]

The HOME directory is the directory in which the user lands when logging into the system.
This directory contains user account directories. Each user created by the system administrator will have a subdirectory under /home with the name of the account. This is the default behavior of Linux systems. E.g. User account for Anna is created; her home directory will be located in /home/anna. All her personal files will reside in this directory. All participants in this class are using the home directories of their respective user accounts.
login refers to the user's account name whereas machine name refers to the name of the system where the account resides.



/lib [Libraries]

This area Contains shared object library files that are necessary to boot the system as well as programs not directly available to users but used by other programs, such as compilers. They also contain databases (fonts, conversion units, etc.) used by programs.

/mnt

Used for mounting temporary file systems. When mounting a CDROM
for instance, the standard mount point location is /mnt/cdrom.
On the Debian GNU/Linux systems, the mount point has been changed to /cdrom.

/opt

This directory is reserved for all the software and add-on packages that are not part of the default installation. For example, StarOffice, Kylix, Netscape Communicator and WordPerfect packages are normally found here, all third party applications should be installed in this directory. Any package to be installed here must locate its static files (i.e. extra fonts, clipart, database files) in a separate /opt/'package-name' directory tree (similar to the way in which Windows will install new software to its own directory tree C:\Windows\Progam Files\"Program Name"), where 'package-name' is a name that describes the software package.

/proc

proc provides information about running processes and the kernel. A directory is provided for each running process. Useful system information such as the amount of Random Access Memory (RAM) available on the system as well as Central Processing Unit (CPU) speed in Megahertz (MHz) can be found within the /proc directory. The following commands will give you this information:
$ cat /proc/cpuinfo - Display CPU information of system
$ cat /proc/meminfo - Display RAM information as well as swap space capacity and usage.

/sbin

Similar to /bin, this directory contains executable programs needed to boot the system, however the programs within /sbin are executed by the root user. Contains system maintenance programs, examples of which are:

* ifconfig (interface configuration, use this command to add or remove a network interface)
* mkfs (make a file system on a partition)
* lilo (boot loader software, tells your Master Boot Record (MBR) where to find your operating system(s). Linux Loader (LILO) stores its working files in /boot.

/tmp [Temporary Directories]

/tmp is a necessary "temporary" directory. It is available to all users to read from and write into. But, along with /var/tmp and /usr/tmp, it is mostly used by compilers and by Linux as temporary storage for intermediate files.
This directory is used for temporary storage space. Files within this directory are often cleaned out either at boot time or by a regular job process. The Debian GNU/Linux operating system cleans up the /tmp directory at boot time. An example for using the /tmp directory would be when downloading the Open Office deb packages. By downloading these packages into the /tmp directory, the user can be assured the packages will be wiped off the system next time the machine reboots.

/usr

This directory contains user applications and a variety of other things for them, like their source codes, and pictures, docs, or config files they use. /usr is the largest directory on a Linux system, and some people like to have it on a separate partition.

When installing an application on a Debian GNU/Linux machine, the typical path to install would be /usr/local. You will notice the directory structure within /usr appears similar to the root directory

structure. Some directories located within /usr include:

* /usr/doc - Documentation relating to the installed software programs.

* /usr/bin - Executable programs that are not required for booting or repairing the system.

* /usr/local/src - Source code for locally installed applications.

/var

this directory contains variable data that changes constantly when the system is running. Files in /var are dynamic and are constantly being written to or changed. Some directories located within /var include:

* /var/spool - files in the print queue
* /var/log - files containing logging information
* /var/run - files containing the process ID's for each current process.

Exercises

1. You want to see which demonstration programs you have on your system. In which directory are you more likely to find these programs?
2. Where would the most used commands be?
3. Where would your home directory be?

Permissions/File Access Modes

Permission defines the ownership of a file and the access of all users to that file (or directory).

Each file and directory has permissions associated with it. Those permissions are made up of three groups of three characters: rwx rwx rwx (read, write and execute). The first group represents the

owner permissions on the file, the second group represents the group permissions on the file, and the third group represents the world permissions on the file.

The owner is the user owning the file.

Each user has an internal user number (UID), and an internal group number (GID), set by the system administrator. Everyone with the same group number is said to be in the same group. Group permissions apply to everyone else with the same group number.

world is everyone else.

For example, the file phone.numbers may have permissions set as rwxrwxr-. This would mean that the owner of the file can read the file, write to it, and execute it. People in his/her group have the same privileges. Everyone else can only read the file; they cannot change, or execute it.
The permissions of a file/directory can be changed with the chmod command.

chmod: Change Mode (Permissions)

The chmod command allows a user to change the permissions of a file/directory. To use chmod, the user must be the owner of the file. The syntax of the command is:

chmod [-R] *mode filename(s)*
-R is an option.
-R
(Recursively) will cause all files and directories within (underneath) the file/directory whose permissions are being changed to take those permissions.

mode may be specified as three octal values (one for each of the three sets of permissions): if any of the permission bits r,w, or x is set, the corresponding permission is enabled: give it a 1. If not, give it a 0. Then, for each of the three permission groups, interpret the three binary numbers as an octal number.

Another way to explain it is to give different weighing factors to the different permissions: 4 to "r", 2 to "w", and 1 to "x". If the permission is set, add the weighing factor. Otherwise do not add anything to the group value.

Using the above example of the file phone.numbers, which had the rwxrwxr- permissions, it would translate to 111 111 100, or 774, or 4+2+1 4+2+1 4+0+0:

rwx rwx r-
111 111 100
7 7 4

If write permission for world (sometimes called others) is added, mode is changed to 776 (or 111 111 110, or 4+2+1 4+2+1 4+2+0).

Mode may also be specified symbolically as +r which would add read permissions to everyone, or as -w which would take away write permissions from everyone (all groups).
filename(s) may be one or more filenames, and/or one or more directories, separated by blank spaces.
For example, to change the permissions on file phone.numbers, from rwxrwxr- to rwxrwxrw-, the command

chmod 776 phone.numbers
or
chmod +w phone.numbers
or (because we are changing it only in the others group)
chmod go+w phone.numbers
which means "group others add write", could be used.

The general syntax, using the conventional method is

chmod [ugo]+| -[rwx] filename(s)
Where
u
:permission for user/owner.
g
: permission for group.

o
: permission for others.
r
: read permission.
w
: write permission.
x
: execute permission.

Any combination of ugo, rwx may be used. If none is used, then all three are assumed.

It is also acceptable to put a combination of "[ugo]+| -[rwx]", as long as they are separated by a comma (,), as in

chmod ug+x,o-x filename

Typing these commands you can get help.
chmod -help
or

man chmod

or

info chmod

chown this command will change the owner of a file or directory.

chgrp this command will change the group of the file or directory.

Exercises

1. Someone just created a file in his/her directory. Can I go and change permissions on it? Why, or why not?
2. I just created a new file in my directory, but I would like it to have rwx permissions for me, and r permissions for everyone else. What command can I use to change those permissions?
3. I have a file with permissions rwxrwxr-x. I want it to have

rwxr-xr-x permissions. What commands (I want two) can be used to change the permissions on it?
4. Take the same file. I want to remove all execute permissions for everyone, except myself. Again, which command can be used to achieve my goal?