

Basic

UNIX

course

Conducted by Ong Zhiyang

UNIX



A*STAR

IME

The UNIX environment

- To work efficiently, become comfortable with the UNIX environment
- Work in a UNIX environment whenever you are able to
- Use Cygwin on Microsoft Windows to get a UNIX look-and-feel.
- Else, try running the Linux or Mac OS operating systems

Course Objectives

The course objectives are to learn:

- How to log in and out of the system
- The fundamental commands of the UNIX operating system to access and manage files as well as directories
- How to change permission on files and directories
- How to list and control the processes running on the system

UNIX Structure

- Operating System
- The UNIX File System
- UNIX Directories, Files and Inodes
- UNIX Programs

UNIX Operating System

UNIX is a three-layered operating system consisting of the following:

- Hardware – provides services for the UNIX operating system
- Kernel – interacts directly with the hardware and provides the services to the user programs,
- User programs – Software used by the user to perform various tasks.

For example: Open Office, xedit, or Cadence

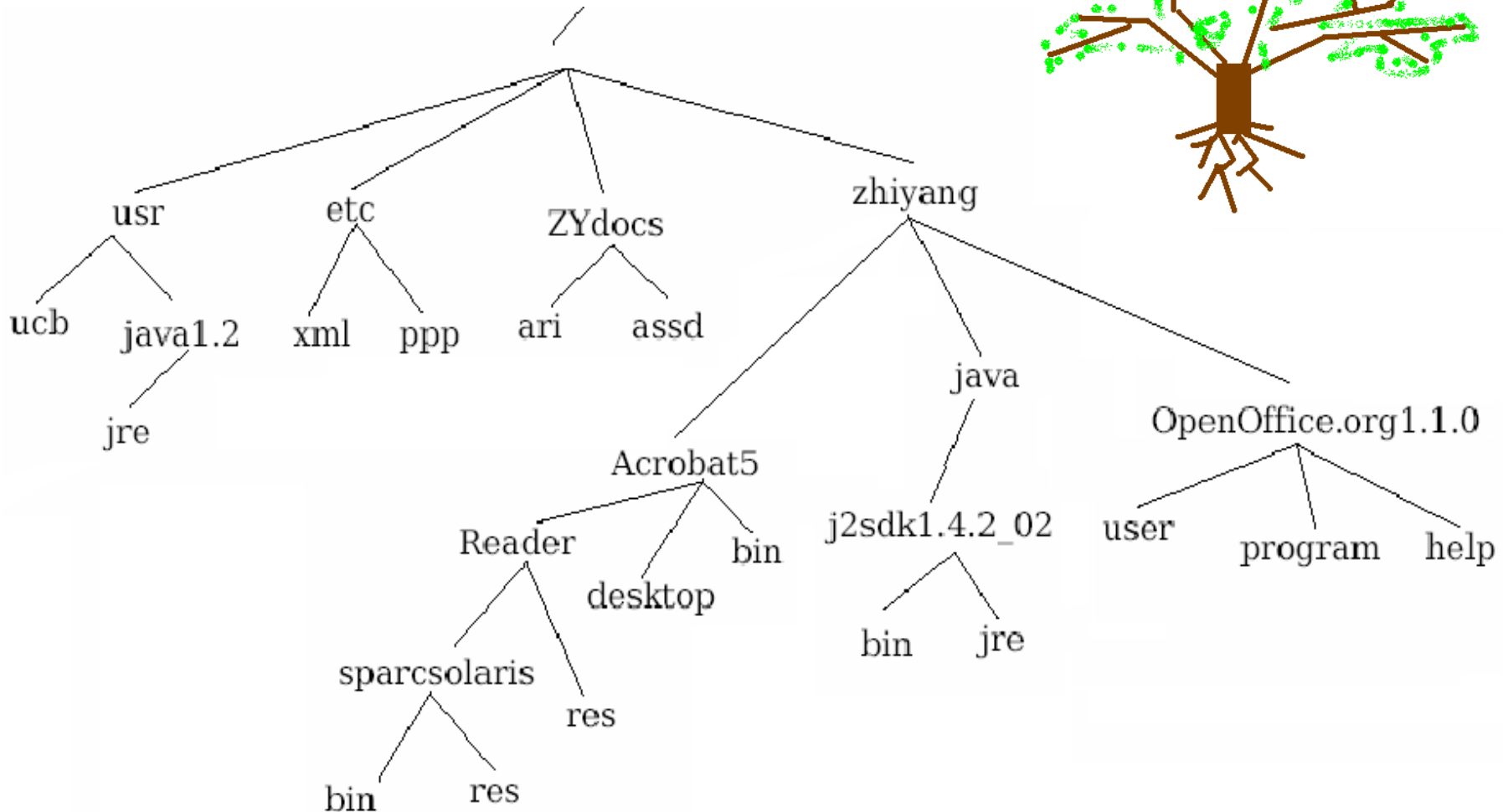
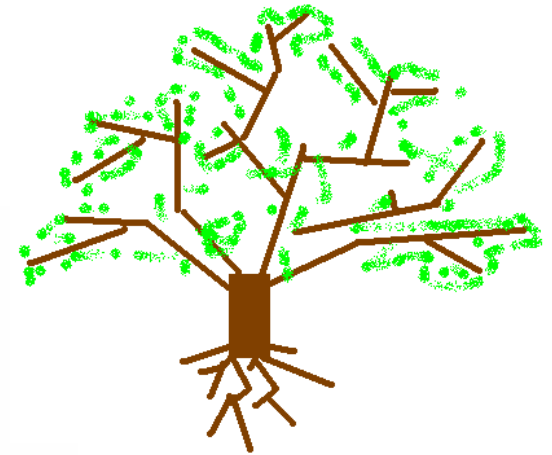
User programs

- The user program does not require any information about the hardware
- That is, the hardware is encapsulated for the user programs.
- Hence, programs can be reused in other machines running the UNIX operating system.
- User programs interact with the kernel by using a set of standard system calls

Multi-User, Multi-Processor UNIX

- UNIX is a multi-user, multi-processor operating system.
- That is, UNIX allows many users, whom are running many programs each, to be logged into a system simultaneously.
- The kernel's job is to keep each process and user separate as well as to regulate access to system hardware.

UNIX File System



UNIX File System

- The UNIX file system resembles an inverted tree structure
- Its root, the root directory, is denoted by “/” (In Microsoft Windows Platforms, the slashes are backwards, “\”)
- It resembles the root of a tree
- The tree’s branches are the paths, where directories/subdirectories (internal nodes) indicate a branch splitting off into two.
- Its leaves are files, which are “external nodes”

- A subdirectory is the child of the current working directory
- With exception of the root directory, every file and directory is listed in its parent directory.
- The parent of the root directory is itself.
- Each node is either a file (external nodes) or a directory (internal nodes) of files that can contain other files and subdirectories.

Specification of a Directory/File

- A file or directory can be specified using an absolute or relative path name.
- An absolute file name is also known as the full path name.
- A full path name begins with the root, /, and follows the branches of the file system, each separated by “/”, until you reach the desired file (or directory), “acroread.txt”.
- For example, if the user is in the root directory:

`/zhiyang/Acrobat5/Reader/sparcsolaris/bin/acroread.txt`

- A relative path name specifies the path relative to another, usually following the current working directory the user is at.
- Special directory entries:
 - “.” indicates the current directory
 - “..” indicates the parent of the current directory.
- For example, if the user is at the directory “zhiyang/OpenOffice.org1.1.0/user”, a relative path to get to the “acroread.txt” file is:

`../../Acrobat5/Reader/sparcsolaris/bin/acroread.txt`

UNIX Directories, Files and Inodes

- A directory is a file that contains a table listing the files contained within it.
- In that table, each filename is assigned to the inode number in the list.
- An inode is a special file designed to be read by the kernel; it provides information about each file.
- The information included are: its permissions, ownership, the physical location of the data blocks on the disk containing the data as well as dates of creation, of last access and modification

- The UNIX file system does not require any particular structure for the data in the file itself.
- The file can be ASCII, binary, or a combination of both
- That is, UNIX does not yield information about the type of the files in a directory since it treats files of all types as the same.

UNIX Programs

- A **program**, or **command**, interacts with the kernel to provide the environment and perform the functions called for by the user.
- A program can be a:
 - Shell script
 - Built-in shell command
 - Compiled source
- A **shell** is a command line interpreter (CIL) and facilitates the user interaction with the kernel.

Getting Started

- Logging In and Out
- UNIX Command Line Structure
- Control Keys
- Getting Help
- Direction Navigation and Control
- File Maintenance Commands
- Display Commands

Logging In

- A user of the UNIX server has a unique login name as well as a changeable password that must be kept confidential to the user.
- To login in:
 - Enter the username at the login prompt
 - Enter the password at the password prompt
- UNIX is **CASE SENSITIVE**

Passwords

- Users must choose a password that they can remember as well as one that is not obvious
- This prevents other people from trying to gain access to the user's UNIX account/UNIX system.
- Passwords:
 - Use at least 6 characters
 - Use a mixture of Upper and lower cases
 - Use a mixture of character types

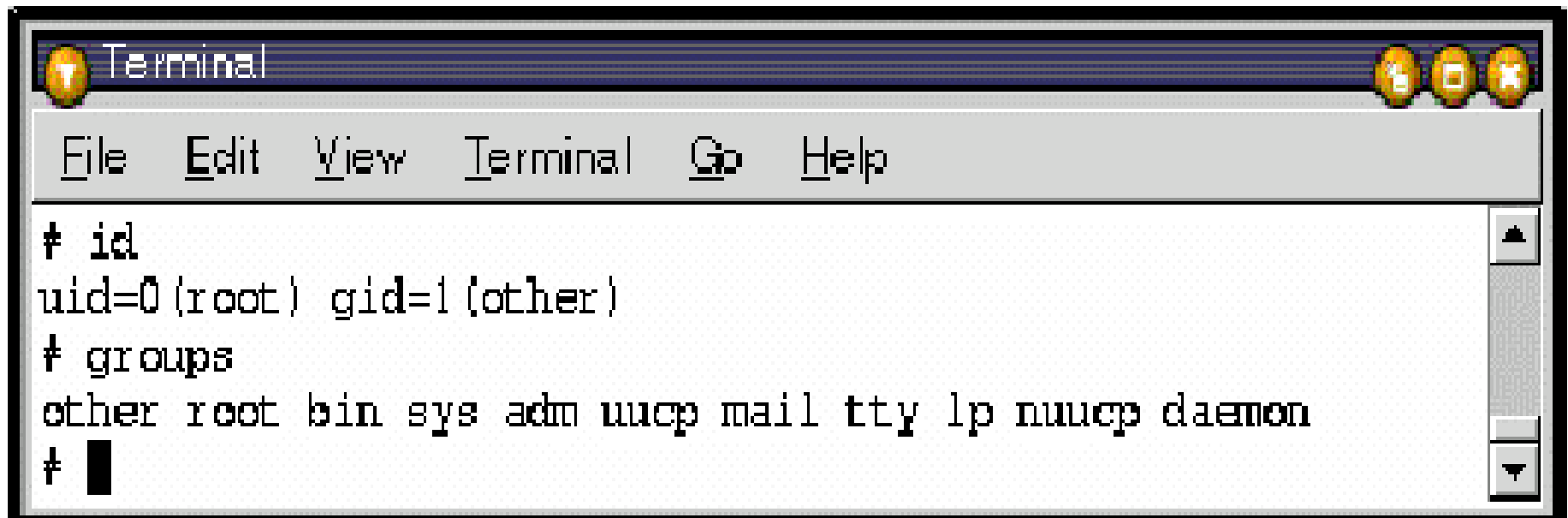
Changing/Forgetting your Password

- If a user forgets his/her password, the user will be issued a temporary password by the system administrator
- The user will have to change this password by using the *passwd* command.
- Passwords should be changed often to avoid break-ins to the systems by hackers

- Each user has a memory space quota allocated by the user's department system administrator.
- Once this quota has been exceeded, the user will not be able to login to the system.
- Use the *du* command to check the disk usage of your UNIX account.
- Always log out after each UNIX session to protect your terminal
- This will prevent unauthorized users from taking advantage of your access.

Exiting and Identity

- Use the `logout` command to leave the system
- Use the `exit` command to leave the shell
- The UNIX system identifies each user by the user's user number (`userid`) and group numbers (`groupid`)
- They are assigned to that user by the system administrator.
- Use the *`id`* command to find out the `userid` of the user and the *`group`* command to find out the groups that user belongs to.

A screenshot of a terminal window titled "Terminal". The window has a menu bar with "File", "Edit", "View", "Terminal", "Go", and "Help". The terminal content shows the execution of two commands: "id" and "groups". The output of "id" is "uid=0 (root) gid=1 (other)". The output of "groups" is "other root bin sys adm uuucp mail tty lp uuucp daemon". The prompt character is "#".

```
# id
uid=0 (root) gid=1 (other)
# groups
other root bin sys adm uuucp mail tty lp uuucp daemon
# █
```

UNIX Command Line Structure

- A **command** is a program that tells the UNIX system to do something. Its syntax is:

command [options] [arguments]

- where an option modifies the way the command performs
- whilst an argument indicates what the command will perform its action on

- The syntax of a command with multiple options can either be:

`command -[option1][option2][option3]`

or

`command -option1 -option2 -option3`

Control Keys

- **Control keys** are used to perform special functions on the command line or within a text editor.
- Hold down the **Control** key and another key simultaneously to type these control keys.
- For example,
- S – Tell the terminal to stop accepting input
- Q – Tell the terminal to start accepting input
- U – Erase the entire input line
- D – Indicates the end of the data stream; a user can log off.
- C – Interrupt the current process and bring up the shell prompt on the shell terminal again.

Getting Help

- UNIX has a manual, called the **man pages**, available on-line to explain the usage of the UNIX system and its commands.
- The syntax for the **man** command is:

`man [options] command_name`

- An alternative to the man pages is the help option available for most commands. Its syntax is:

`command_name -help`

Directory Navigation and Control

- System and user directories are organised under the **root**.
- The user does not have a root directory in UNIX since they usually log into their own **home** directory belonging to their account.
- Users are able to create other directories under their home directories.

Basic commands to navigate & control

Command	Consequent Actions
cd	Change directory
ls	List the contents of a directory
mkdir	Create (make) a directory
rmdir	Remove an empty directory
pwd	Displays the location in path that is the current (present) working directory.

Examples...

- ***cd /full/path/name/from/root*** – Changes directory to absolute path named
- ***cd path/from/current/location*** – Changes directory to path relative to current location
- ***cd ~username/directory*** – Changes directory to the named username's indicated directory

File Maintenance Commands

- File maintenance commands are used to create, copy, remove and change permissions on files.

- The *cp* command is used to copy one file to another.

- Its syntax is:

```
cp [options] old_filename new_filename
```

- The *mv* command is moved from one file into another.

- Its syntax is:

```
mv [options] old_filename new_filename
```

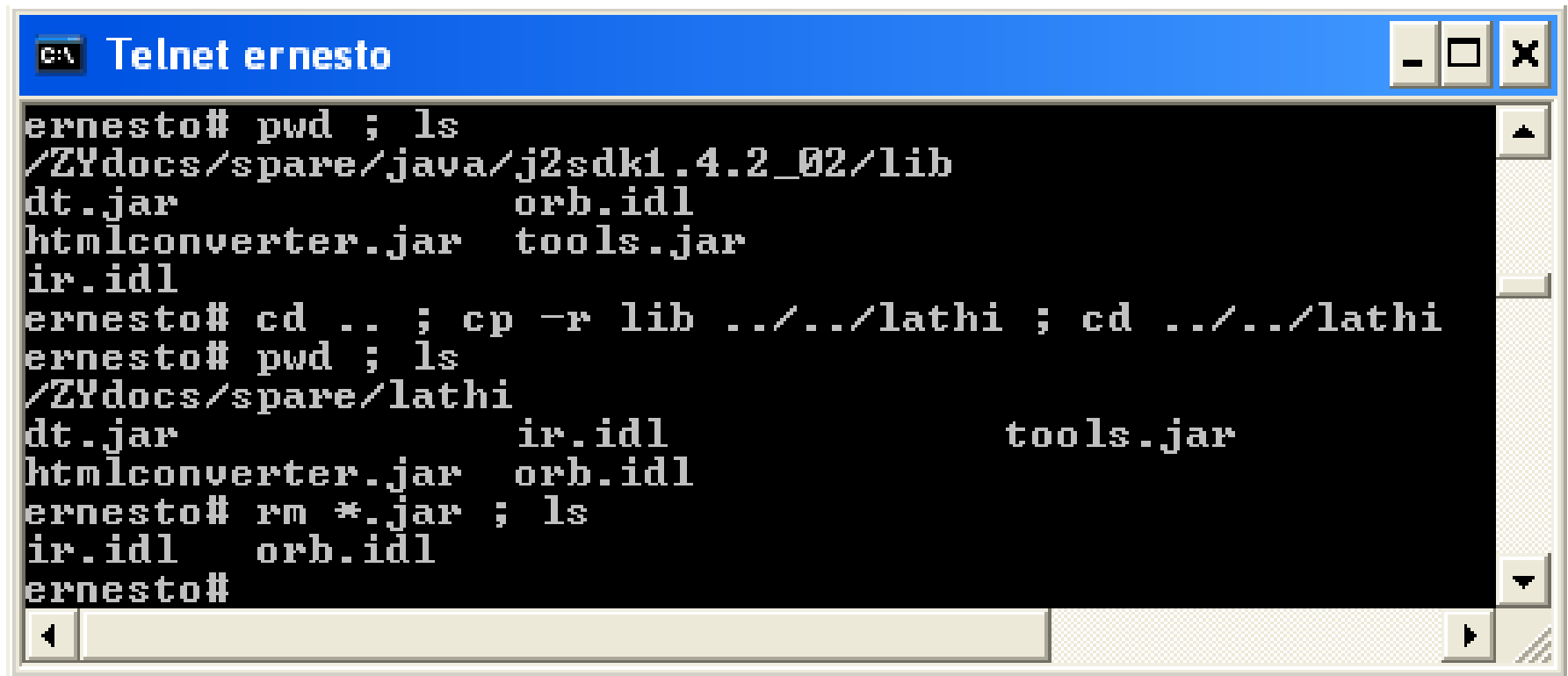
- The *rm* command is used to remove a file.

- Its syntax is:

```
rm [options] filename
```

- Use the `-r` option to allow these commands to be performed recursively on all the subdirectories and files
- Use the `-i` option for interaction
- Use the `-f` option to disable prompts
- For example:
 - `rm -r -f`
 - `mv *.java ../../lakos/projects/avvt`
 - `cp -r anand/ ./pardhi/ccp/prac2`
 - `du -a -k`
 - `ls -a -l -p`

```
Telnet ernesto
ernesto# ls -p
CHANGES                               Welcome.html
COPYRIGHT                              bin/
ControlPanel.html                     javaws/
LICENSE                                lib/
README                                 plugin/
THIRDPARTYLICENSEREADME.txt
ernesto# pwd
/Zydocs/spare/java/j2sdk1.4.2_02/jre
ernesto# cd .. ; rm -r -f jre
ernesto# pwd ; ls
/Zydocs/spare/java/j2sdk1.4.2_02
COPYRIGHT                               demo
LICENSE                                 include
README                                  lib
README.html                            man
THIRDPARTYLICENSEREADME.txt           src.zip
bin
ernesto#
```

```
ernesto# pwd ; ls
/ZYdocs/spare/java/j2sdk1.4.2_02/lib
dt.jar          orb.idl
htmlconverter.jar  tools.jar
ir.idl
ernesto# cd .. ; cp -r lib ../../lathi ; cd ../../lathi
ernesto# pwd ; ls
/ZYdocs/spare/lathi
dt.jar          ir.idl          tools.jar
htmlconverter.jar orb.idl
ernesto# rm *.jar ; ls
ir.idl  orb.idl
ernesto#
```

```
C:\ Telnet ernesto
ernesto# du -a -k
1      ./orb.idl
18     ./ir.idl
20
ernesto# ls -a -l -p
total 42
drwxr-xr-x  2 root  other    512 Dec 15 10:20 ./
drwxr-xr-x  5 root  other    512 Dec 15 10:19 ../
-r--r--r--  1 root  other  18381 Dec 15 10:19 ir.idl
-r--r--r--  1 root  other   429 Dec 15 10:19 orb.idl
ernesto#
```

```
C:\ Telnet ernesto
ernesto# pwd ; ls -p
/ZYdocs/spare/java/j2sdk1.4.2_02/demo
applets/  jfc/      jni/      jpda/     plugin/
ernesto# mv ../../../../*.sxw ../../ambardar
ernesto# cd ../../ambardar ; pwd ; ls
/ZYdocs/spare/java/ambardar
UNIXcourse.sxw  UNIXcourse01.sxw  UNIXcourse02.sxw  UNIXcourse03.sxw
ernesto#
```

- **Be very careful when overwriting a file or deleting it.**
- **UNIX does NOT have a “unremove” command.**
- **The system administrators may not necessary have backups of all your files.**
- **Hence, data lost will mean that you have to redo some of your work!**
- The *rm*, *mv* and *cp* commands can be used to remove, move or copy files from one directories to another directories on a different level
- Use `../` to move up one directory level
- Use `./subDirName` to access the subdirectory “subDirName”.

- Use the *ls* command to indicate that the file(s) and/or subdirectories no longer exist following a *rm* command.
- It can also be used to list file permissions as well by using:

```
ls -l
```

- The *chmod* command is used to change file permissions on an item (file, directory, etc).
- Its syntax is:

```
chmod abc [argument list] numeric mode
```

```
chmod [who]op[perm] [argument list] symbolic mode
```

- [argument list] contains the file, list of files or directories that you want to change.
- To indicate more than one file or subdirectory, append one after another with a space in between them.

- For the numeric mode:
- abc indicates the permissions that you want to set for the user (owner), group (that the owner belongs to) and other users.
- For symbolic mode,
- [who] is used to indicate the user (u), group (g) or other users (o)
- op indicates whether permissions for that argument list are
 - added to (+)
 - removed from(-)
 - set to (=)
- [perm] indicates what permission – read (r), write (w) and/or execute (x) - is given.

```
Terminal
File Edit View Terminal Go Help
ernesto# ls -l
total 994
-rw-r--r--  1 root    other    27643 Dec  3 13:18 UNIX.doc
-rw-r--r--  1 root    other    9727  Dec  3 12:58 UNIXcourse.doc.gz
-rw-r--r--  1 root    other   32158 Dec  5 09:59 UNIXcourse.sxw
drwxr-xr-x  2 root    other    512  Dec  4 17:32 asd
-rw-r--r--  1 root    other    4147  Dec  3 16:42 imelogo.gif
-rw-r--r--  1 root    other   14762 Dec  3 15:13 ooInstall.txt
-rw-r--r--  1 root    other     1  Dec  3 15:08 ooInstall.txt~
-rw-r--r--  1 root    other  145171 Dec  4 10:10 pkillGrep.jpg
-rw-r--r--  1 root    other  102970 Dec  4 09:55 ps&killEg.jpg
-rwxr-xr-x  1 root    other     27 Dec  5 10:03 temp.txt
-rw-r--r--  1 root    other   19456 Dec  4 08:34 test.doc
-rw-r--r--  1 root    other    1872 Dec  4 14:12 test.pdf
-rw-r--r--  1 root    other   63999 Dec  4 13:34 useridGrpID
-rw-r--r--  1 root    other   57755 Dec  4 13:42 useridGrpID.jpg
ernesto# chmod u-r--,go-rw- temp.txt
ernesto# ls -l
total 994
-rw-r--r--  1 root    other    27643 Dec  3 13:18 UNIX.doc
-rw-r--r--  1 root    other    9727  Dec  3 12:58 UNIXcourse.doc.gz
-rw-r--r--  1 root    other   32158 Dec  5 09:59 UNIXcourse.sxw
drwxr-xr-x  2 root    other    512  Dec  4 17:32 asd
-rw-r--r--  1 root    other    4147  Dec  3 16:42 imelogo.gif
-rw-r--r--  1 root    other   14762 Dec  3 15:13 ooInstall.txt
-rw-r--r--  1 root    other     1  Dec  3 15:08 ooInstall.txt~
-rw-r--r--  1 root    other  145171 Dec  4 10:10 pkillGrep.jpg
-rw-r--r--  1 root    other  102970 Dec  4 09:55 ps&killEg.jpg
-r--r--r--  1 root    other     27 Dec  5 10:03 temp.txt
-rw-r--r--  1 root    other   19456 Dec  4 08:34 test.doc
-rw-r--r--  1 root    other    1872 Dec  4 14:12 test.pdf
-rw-r--r--  1 root    other   63999 Dec  4 13:34 useridGrpID
-rw-r--r--  1 root    other   57755 Dec  4 13:42 useridGrpID.jpg
ernesto#
```

Display Commands

- The *echo* command is used to repeat the argument you give it back to the standard output device
- The *cat* is used to concatenate the contents of a list of files and display it on the shell terminal.
- Its syntax is:

```
cat file1 file2 file3
```

- The *more* command allows you to page through the contents of a file one screenful or line at a time. Some UNIX systems allow *less* or *pg* to be substitute commands.
- The *more* command has the following syntax:

```
more [options] [+/pattern] [filename]
```

- The *head* and *tail* commands can be used to display a determined number of lines from the top or the bottom of the file.



System Resources and Printing

- System Resources
- Print Commands

System Resources

- The *ps* command is used to determine which process is currently running (or not running)
- It also gets the following detailed information about each process at the time this command is typed.
- PID (Process ID) – Datum required to kill a process
- TTY (Control Terminal and Priority) – The controlling terminal for the process
- Time – Cumulative execution time for the process
- CMD (Command) – The command name (the full command name and its arguments, up to a limit of 80 characters, are printed under the -f option)

- The *kill* command eliminates a process that is currently executing.
- It is recommended that this command be used only if the normal way of quitting a process/application does not work.
- The syntax for using the *kill* command is:

kill -signal PID

- where signal is a number or name.

```
# ps
  PID TTY          TIME CMD
 1622 pts/5        0:00 ps
 1585 pts/5        0:00 sh
 1621 pts/5        0:00 xedit
# kill 1621
# ps
  PID TTY          TIME CMD
 1623 pts/5        0:00 ps
 1585 pts/5        0:00 sh
1621 Terminated
# xv &
xv: not found
1628
# []
```

- The *pgrep* command is used to examine the active processes on the system
- It also reports the process IDs of the processes whose attributes match the command-line argument.
- An example of the syntax for this command is:

`pgrep process-name`

```
Terminal
File Edit View Terminal Go Help
# ps
  PID TTY          TIME CMD
 1585 pts/5        0:00 sh
 1643 pts/5        0:00 xedit
 1644 pts/5        0:00 ps
# pgrep xedit
1643
# ps
  PID TTY          TIME CMD
 1585 pts/5        0:00 sh
 1643 pts/5        0:00 xedit
 1646 pts/5        0:00 ps
# pkill xedit
# ps
  PID TTY          TIME CMD
 1585 pts/5        0:00 sh
 1648 pts/5        0:00 ps
1643 Terminated
# █
```

- The *date* command is used to display the current date and local time as well as the time zone of that time.
- The *du* command is used to report the amount of disk space in use for the files or directories specified.
- Its syntax is:

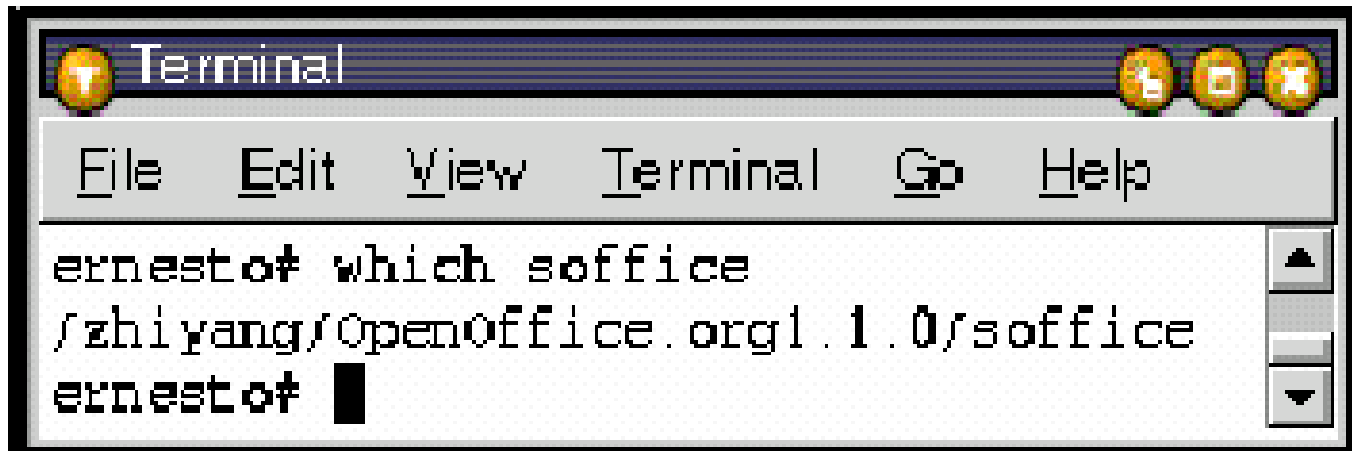
du [options] [directory or file]

- The *quota* command is an alternative command for the *du* command.
- It displays a user's UNIX file system disk quota and usage.
- The *-v* option is used to display the user's quota on all mounted file systems where quotas exist.

- The *which* command reports the path to the command or the shell alias in use.
- Its syntax is:

which command_name

- The *who* command reports who is logged in at the present time.
- The *ami* (*am i*) option displays the username of the user who is currently logged in.



A screenshot of a terminal window titled "Terminal". The window has a menu bar with "File", "Edit", "View", "Terminal", "Go", and "Help". The terminal content shows the command `ernesto# which soffice` and its output `/zhiyang/OpenOffice.org1.1.0/soffice`. The prompt `ernesto#` is followed by a cursor.

Print Commands

- The *lpr* or *lp* command is used to submit a specified file, or standard input, to a defined printer for printing.
- Each print job is given a unique request-id that can be used to follow or cancel the job while it is in the queue.
- Their syntax are:

`lpr [options] filename`

`lp [options] filename`

- The *lpq* or *lpstat* command is used to check the status of the user's print job
- It is usually used with the *-P* or *-p* status respectively to indicate the status of a specific printer.

- The *lprm* or *cancel* command is used to remove a user's print job from the print queue.

- A user cannot remove the print job of another user.

`cancel [request-ID (of the print job)] [specified printer]`

`lprm -Pspecified_printer [request-ID (of the print job)] [username]`

- where `specified_printer` is the name of the printer where the print job is sent to.

- The *pr* command is used to filter the file, and prints the header and trailer information surrounding the formatted file to the terminal.

- Its syntax is:

`pr [options] filename`

Shells

- The UNIX shell acts as an interface between the user and the operating system
- As a command-line interpreter, it reads typed commands and passes them on for further action to be taken by the UNIX operating system after translation.
- Two commonly used shells are the *Bourne* shell, sh, and the *C* shell, csh.
- The former uses the \$ symbol as its shell prompt whilst the latter uses the % symbol.
- Type csh at the Bourne shell prompt to change from the Bourne shell to the C shell.
- To enter multiple commands on the same line, type a semicolon (;) between each command

```
%
```

```
%cd MyDocs/JavaApps/
```

```
$
```

```
$du
```

```
#
```

```
#^C
```

```
#java MyFactorial
```

- A “not found message” will appear, followed by the shell prompt, if a command unknown to UNIX is entered.
- The *xterm* command allows a new shell terminal to be opened.
- The symbol **&**, following a command, runs a process in the background
- In the C shell, the *alias* command can be used to assign a name to a function.
- Following **^Z**:
 - The *fg* command is used to put a job into the background
 - The *bg* command is used to bring a job to the foreground.

- The *set* command is used to set a shell variable whilst the *setenv* command is used to set an environment variable for this and subsequent shells.
- Typing **^Z** followed by *bg* brings an application to the background and typing *fg* brings it back to the foreground.

Useful Commands

- The *grep* command is used to search for generalized regular expressions occurring in UNIX files.

- Its syntax is:

```
grep [options] regexp [file[s]]
```

- The *diff* command is used to compare two files, or directories, and display the differences between the two (text/ASCII files only).

- Its output format is designed to report the changes necessary to convert the first file into the second.

- Its syntax is:

```
diff [options] file1 file2
```

- The *compress* command is a utility used to reduce the size of the named files.
- Except when the output is to the standard output, each file will be replaced by one with the extension *.Z*
- The same ownership modes, change times and modification times are kept by each *.Z* file.
- Its syntax is:

```
compress [ -fv ] [ -b bits ] [ file1 file2 file3 ... ]
```

```
compress [ -cvf ] [ -b bits ] [ file ]
```

- The *uncompress* command is a UNIX utility that will restore files to their original state after they have been compressed using the UNIX *compress* utility.

- The *find* command recursively descends the directory hierarchy for each path seeking files that match a Boolean expression written in the predicate list.

- A user can use the File Transfer Protocol (**FTP**) to transfer files from your account in one server to that in another
- Or from one PC workstation to another. To ftp into a server, type the following:

ftp server_name

- To copy those files from your account in the server to your workstation, try:

get filename

- If you want to put a file from your PC workstation, which runs Microsoft Windows, into your account in the server, and try:

put filename

- When you have finished transferring the files you desire, type *bye* to return to *MS_DOS* or *xterm/Terminal*.

- Type `ascii` to transfer files in the ASCII format
- Type `binary` to transfer files in binary format
- Usually text files are transferred in the ASCII format

- Similarly, to telnet into your account in the UNIX server, a user needs to login as the user would with a UNIX workstation.
- *Telnet* is a Internet telecommunications protocol that enables a computer to function as a terminal working from a remote computer.
- That is, it allows a computer to function as a terminal emulator.
- The command to telnet into a UNIX server is:

```
telnet UNIX_server_name
```
- Upon successful login, the user can use UNIX commands just like the user is able to on a UNIX Terminal or xterm window.

- To login remotely, use the rlogin command.

- Its syntax is:

```
rlogin [ -l username ] hostname
```

- A remote login from your terminal is established to the remote machine named hostname
- FTP only allows file transfer whilst rlogin and telnet allows you to access the remote machine via a terminal
- To export display:
 - Type “xhost +” at the shell prompt
 - Telnet to server
 - Set DISPLAY, an environment variable, to that of the IP address of that server

Other useful commands

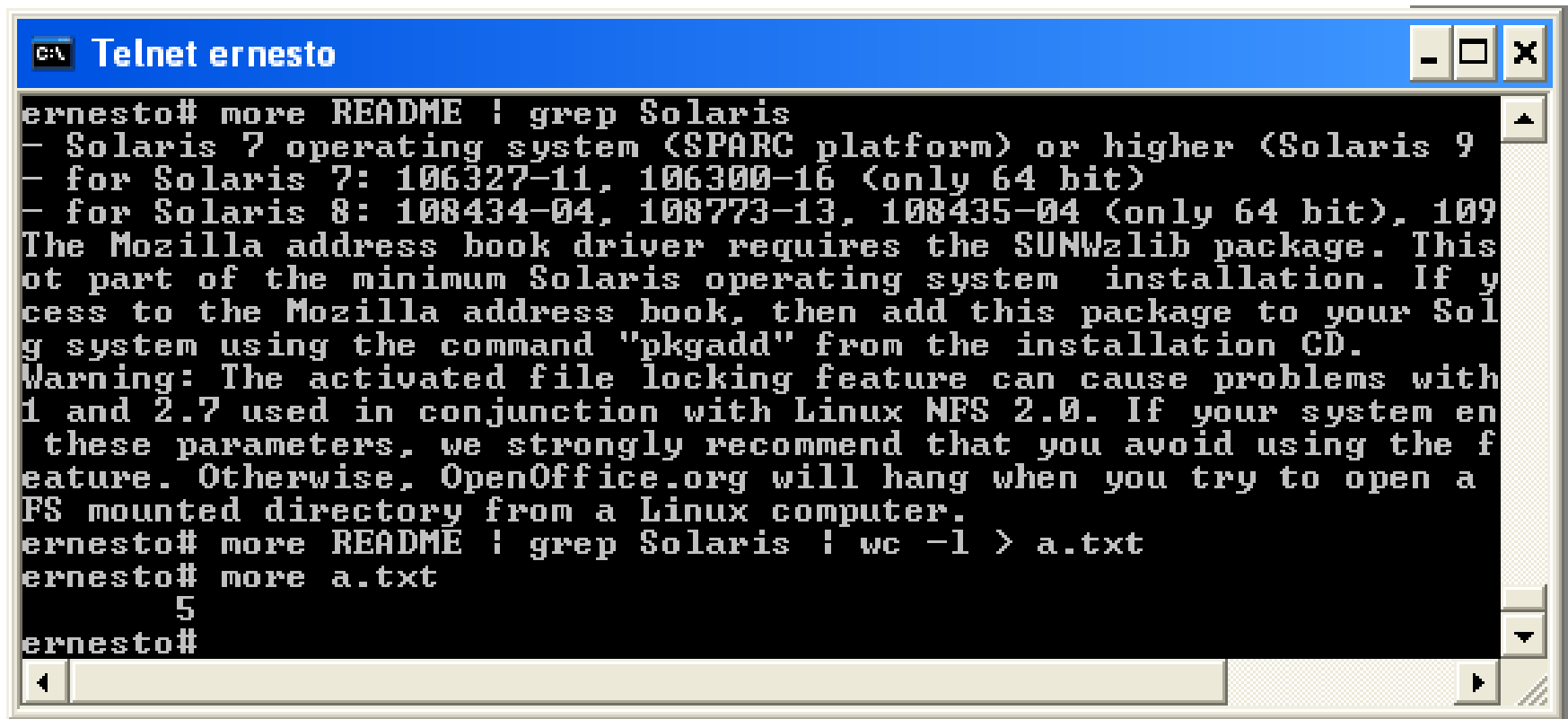
- Difference between these ps commands
 - /usr/bin/ps – includes PID, TTY, TIME and CMD
 - /usr/ucb/ps – includes the state of the process as well
- hostname – set/print name of current host system
- uname – print name of current operating system
- rcp – show host status of remote machines
- touch – change file access and modification times

```
C:\ Telnet ernesto
ernesto# hostname
ernesto
ernesto# uname
SunOS
ernesto# rup
  fe80::203:baff:   up   3 days, 21:49,   load average: 0.00, 0.01, 0.01
ernesto.ime.org   up   3 days, 21:49,   load average: 0.00, 0.01, 0.01
  fe80::a00:20ff:  up  40 days, 27 mins, load average: 0.00, 0.00, 0.01
  fe80::a00:20ff:  up  13 days,  1 min,  load average: 0.00, 0.00, 0.01
  fe80::a00:20ff:  up  13 days, 52 mins, load average: 0.00, 0.01, 0.01
  fe80::a00:20ff:  up  13 days, 24 mins, load average: 0.00, 0.00, 0.01
  fe80::a00:20ff:  up  13 days, 34 mins, load average: 0.00, 0.00, 0.01
  fe80::a00:20ff:  up   9 days, 17:02,  load average: 0.00, 0.00, 0.01
  fe80::a00:20ff:  up  16 days, 20:15,  load average: 0.01, 0.02, 0.02
jamp7800.ime.or   up  90 days, 23:50,  load average: 0.42, 0.38, 0.37
tianjin.ime.org    up   9 days, 17:02,  load average: 0.00, 0.00, 0.01
hangzhou.ime.or   up  16 days, 20:15,  load average: 0.01, 0.02, 0.02
ernesto# _
```

Command Piping

- Command piping involves sending the output of one command as the input to another command.
- The | symbol is used to pipe output to another command by filtering output of the first command to be the input of the second command.
- Its syntax is:

```
command1 | command2
```
- ~ is used to refer to the home directory of the current user in the C shell.
- ~*user* is used to refer to the home directory of the specified user in the C shell only.



```
ernesto# more README | grep Solaris
- Solaris 7 operating system (SPARC platform) or higher (Solaris 9
- for Solaris 7: 106327-11, 106300-16 (only 64 bit)
- for Solaris 8: 108434-04, 108773-13, 108435-04 (only 64 bit), 109
The Mozilla address book driver requires the SUNWzlib package. This
ot part of the minimum Solaris operating system installation. If y
cess to the Mozilla address book, then add this package to your Sol
g system using the command "pkgadd" from the installation CD.
Warning: The activated file locking feature can cause problems with
1 and 2.7 used in conjunction with Linux NFS 2.0. If your system en
these parameters, we strongly recommend that you avoid using the f
eature. Otherwise, OpenOffice.org will hang when you try to open a
FS mounted directory from a Linux computer.
ernesto# more README | grep Solaris | wc -l > a.txt
ernesto# more a.txt
5
ernesto#
```

finger | grep csxt | sort | pr | lpr -Pcsdn1

- finger lists the users on the system
- grep searches its input for lines containing the string "csxt"
- sort takes lines of input and sorts them alphabetically
- pr formats input into pages, with header information
- lpr sends its input to the printer.

Oops... the system has hung

- Telnet into your system account from another console and kill the appropriate processes
- As a last resort, contact your system administrator

- Note that when an application is not terminated properly, a *core* file of about 30 to 80 MB will be deposited in the current working directory
- This *core* file is ***USELESS***. Delete it!
- However, take care not to delete a core directory that contains your working and useful files (e.g. core design files)!
- To ensure that a core file is deleted instead of a core directory, use the *more* command to view the contents of the file.

- Files like **.pdv* and **.nfs* are useless temporary files that CADENCE creates to backup your files.
- If CADENCE is not shut down properly, they will not be automatically destroyed.
- Thus, you need to manually delete them from your directory.

The END