Lecture 14: Files and Directories

- UNIX stores all files in an identical manner
- File contents are treated as series of bytes
- UNIX treats devices also as files of special type
- File sizes can grow dynamically (file size limited only by disk space available)
- UNIX allows simultaneous access to any file by multiple users
- Each file is internally assigned a unique identification number, called inode
- Directories are files of special type in the UNIX system
- UNIX provides security measures for files

UNIX treats Input / Output devices as files. Programs access devices with the same syntax they used when accessing ordinary files.

UNIX stores all files as a stream of bytes. It is at the user's discretion to interpret the contents of the file.

Space is allocated dynamically according to the size of the file, which changes with the change in the number of bytes.

Several application programs can access a file concurrently to read or update its contents. **UNIX** provides methods to coordinate file access and maintain data integrity. It also provides the facility for file and record locking.

UNIX protects user files by setting certain access rights. These can be for reading, writing and executing the files.

Types of Files

- Ordinary File
- Directory File
- Special File
- Block Device File
- Character Device File
- FIFO files

Ordinary Files

These files are used to store data, executable programs, simple text _
Users, depending on his/her authority, can add data to a file, delete da1c or remove it entirely when the file is no longer needed. Files create: through editors, word processors, assemblers and compilers fall into ths category.

Directory File A directory file contains a list of files or sub directories under it. Eac_ directory entry contains the name, the inode number and other relata: information of that file.

Directories behave just like ordinary files except that some of 1:Pe operations used for manipulating ordinary files do not work with director; files and vice versa.

Special Files Also called as device files, these represent the physical devices. These files are read from and written to just like ordinary files, except tha:. request for read or write causes activation of the physical devices associated.

These files can be classified as Character Device files-and Block Device files.

Character and Block Device Files In a character device file, data is handled character by. character as in the case of **terminals** and **printers.**

In block device files, data is handled in "large chunks or blocks, as in case of disk and **tapes.**

FIFO files FIFO files allow unrelated processes to communicate with each other through **pipes,** where the communication path is only in one direction.

File Naming Conventions

- A file name can be a maximum of fourteen characters long
- No concept of primary and secondary name
- The name may contain alphabets, digits, dots and underscores
- No embedded spaces or tabs are allowed in file names
- System commands or UNIX reserved words can not be used for filenames
- UNIX treats upper case and lower case alphabets separately

Some of the valid filenames in UNIX are:

abc.doc.l
Report 1
reports
.login
rep.123.a
sales.rep
sales_rep

The UNIX system is case sensitive, i.e. UNIX distinguishes the lower case and upper case alphabets as different. So the files Employee and employee are two different files in UNIX system. .

Question: Identify the valid and invalid filenames

1. Emp.dat
2. sal.rep.95
3. .9512001
4. _sales.
5. sal-rep
6. –employee
7. file(3)
8. emp out
9. *IP
10. asc?dat

UNIX Directories

- Directories are special files containing names of other files and *lor* directories
- Home directory: A directory assigned by the super user in which a user normally works
- Current directory: The directory from which the user is currently issuing commands
- At login time, by default the current and home directory are the same
- User can not change the location of home directory

Standard directories of a UNIX system

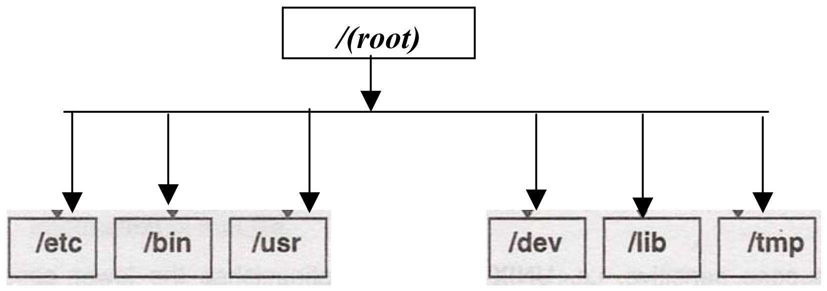


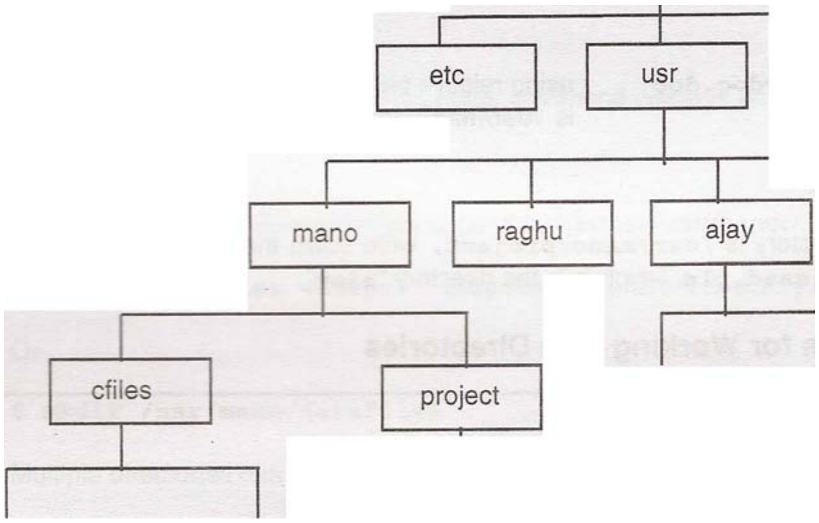
Fig: A typical **UNIX** Directory structure consists of the following directories

/etc	stores the system administration utility files.
/bin	stores the most commonly used **UNIX** commands.
/usr	stores all user home directories and some **UNIX** commands.
/dev	stores all the device files.
/lib	stores library files for C programming.
/tmp	used for temporary storage.

Pathname

- The exact location of a file in a tree hierarchy
- Referring to a file from the root directory : absolute path name
- Referring to a file from the current directory : relative path name
- Directories in path name separated by 'I' (slash)
- '.' (Single dot) represents the current directory

- '..' (double dots) represent the parent directory



A file is identified by its name plus its exact location in the directory structure.

Pathname represents the full name of the path from the root through the tree of directories to a particular file. .

File referencing can start from the root directory followed by all levels of sub directories below it, separated by slashes ('/'). Begin with a slash to indicate that the name starts from the root. This is called **absolute referencing** and the 'path in this case is an absolute **pathname.**

The **relative** path name begins with the current directory and describes the path of directories to the target file. Referring a file using relative path is called **relative referencing.**

Examples:

/usr/mano/ctiles/sales.c gives the full or absolute path name for the file sales.c.

if the current directory is */usr/mano/cfiles,* the file reports, which is in the sub directory project, can be referred to, using relative path name as:

../project/reports

Using absolute path name, the reference will be:

/usr/mano/project/reports

Here '..' (double dots) indicate the parent directory of the current directory.

The file mydoc.doc can be referred to as:

/usr/ajay/mydoc.doc	: using absolute pathname
or	
../../ajay/mydoc.doc	: using relative pathname, if the current directory is */usr/mano/cfiles.*

Question: If the current directory is /usr/mano/project, write down the absolute and relative path names for the file ssad.pln, which is in the directory "ajay".

Basic Commands for Working with Directories

➢ pwd	: print working directory
➢ mkdir	: creating new directories
➢ cd	: changing the directory
➢ is	: listing the directory contents
➢ rmdir	: removing a directory

.pwd

The pwd command is used to display the full path name for the current or the working directory. The path gives information about the location of the working directory in the hierarchical directory structure.

This command displays the absolute path name for the current directory.

The command line format is:

```
$ pwd <Enter>
```

Example:

```
$ pwd <Enter>
The output may be
/usr/mano
```

.mkdir

This command is used to create one or more directories. The new directories created will be empty except for the dot and the double dots entries, which represent the current and the parent directories respectively.

This command has the following format:

```
$ mkdir <pathname>
```

If the directory name already exists then the mkdir command displays an error message

Examples:

To create a sub directory datafiles under the directory *lusr/mano:*

```
$ mkdir datafiles <Enter>      assuming the present directory is /usr/mano.
```

Or

```
$ mkdir /usr/mano/datafiles    <Enter>
```

Multiple directories can be created with a single mkdir command:

```
$ mkdir src exec report        <Enter>
```

will create src, exec and report sub directories under the current directory *lusr/mano.*

Question: Write the commands to create a subdirectory ADCS under the directory /usr/jamil. Your present directory is *lusr/mano* (refer to prev. fig).

.cd

This command is used to change the current directory. Useful to traverse through the directory structure.

```
$ cd [ pathname ] <Enter>
```

where pathname is either an absolute or relative pathname to the target directory.

The cd command in UNIX without an argument will change to the HOME directory of the user whereas the DOS cd command without any arguments works like the pwd command of UNIX.

The DOS cd command when used without the pathname is equivalent to the pwd comrra.-: - UNIX.

```
cd..     : changes to the parent directory of the current directory.

cd../..  : changes to the parent's parent directory.
```

Examples (refer to fig):

```
$ cd !usr        <Enter>        will change the current directory to /usr.
```

To change the current working directory to */sr/mano/project:*

```
$ cd   /usr/mano/project <Enter>
```

To change to the directory ajay from the current directory */usr/mano/project:*

```
$ cd ../../ajay   <Enter>        using relative path name

Or

$ cd /usr/ajay   <Enter>         using absolute pathname
```

Question: Write a single command to change the directory to jamil, if the current directory /usr /mano/project. Do it in two ways, one using absolute pathname and then relative pathname.

.ls

- listing the directory contents
- options with Is command:
    - -l list in long format.
    - -a all files including those which begin with a dot (.)
    - -i lists the inode number in the first column.
    - -x sorts multi column output across the page
    - -C displays files in columns (wider format)
    - -R recursively list all directories and sub-directories

This command is used to list the contents of a directory. **It** can also display information about a group of files given as argument(s).

The **Is** command alone displays all the files and directories in the current directory.

The output is sorted alphabetically on the file name.

The general command line format is:

```
$ Is     [ option...] [pathname]
```

Examples:

To view the contents of the directory ***lusr/ajay***

```
$ cd   /usr/ajay     <Enter>      make ajay as the current directory
$ Is                 <Enter>
```

The output may be:

```
mydoc.doc
ssad.pln
$
```

The output of the above command is an alphabetically ordered list of the files and directories.

The **-I** (lower case L) option with **Is** command displays a long listing giving the file names along with other information about each file.

```
$ Is -1         <Enter>
total 391
-rwxrwxrwx  3     ajay   ru    2080   Oct 10       09:34 mydoc.doc

-rw-rw-rw   1     ajay   ru    3578   Oct 9        11:55 ssad.pln
```

The **-a** option displays the hidden files i.e. the filenames starting with a dot.

The **-i** option displays the inode numbers in the first column and the file names in the next column.

```
$ Is   -i    <Enter>
```

the output may be:

```
8315   mydoc.doc
8312   ssad.pln
```

The -R option displays the content of the directory and the contents of the sub directories present, if any.

The -C option is used to display file names in columnar format. Similar to **DOS DIR/W** command.

The options of **Is** command can be combined:

Is -Ii will display a long listing of files with their inode numbers.

Is -Ia will display the hidden files along with other files in a long format.

.Meta characters or Wild cards

“*”	Match any string of zero or more characters in file name
"?"	Match any single character
[list]	Match anyone character from the list

Filename expansion is the ability to expand a shorthand notation for a group of filenames to a complete set of explicit filenames. The special filename-matching characters are called **Metacharacters,** to suggest patterns or templates to match various classes of filenames.

.The “*” metacharacter

The “*" stands for any combination of **zero or more characters.** It will not match a **dot** at the beginning of the filename.

1s * is equivalent to entering **Is** followed by every filename in the directory.

1s *.c will display all files ending with '.c'

.**The "?" metacharacter**

The '?' (question mark) stands for any single character in a filename.

If the directory contains files, named as **chap1, chap2, chap3, chap4, chapA and chapB**

1s -1 will display a long listing of all the files whose first four characters are "chap"
chap? followed by any single character.

1s??.* will display all the files, having any two characters at the beginning, followed by a dot and upto a total of eleven characters

ls?.? will list all three-character filenames with an embeded ".".

The [ ] metacharacter

This metacharacter matches exactly a single character given in the list.

ls chap[123] will display files named as chap1, chap2 and chap3 only.

ls chap[ABC] will display the files chapA, chapS and chapC.

A range of values can also be given within the brackets.

ls chap[A-Z] will display the files chapA, chapS, chapC, chapD chapZ only.

Example:

1. To display all two-character file names:

```
ls??
```

2. A long listing of all filenames consisting of two lowercase letters:

```
ls -1 [a-z] [a-z]
```

3. display the filenames starting with lowercase c and ending in a digit:

```
ls c*[O-9]
```

4. listing the files having 'i' at the second position:

```
ls?i*
```

It is interesting to note that unlike DOS, which ignores the characters after "*", UNIX on the other hand reads them.

Question: The following files are present in the /unix directory

1. unix1.doc 2. unix1. chp 3. unixa1.doc 4.. unixa1.chp 5. unix2.doc 6. unix2.chp 7. unixa2.doc 8. unixa2.chp

Write the command to display only the files numbered as 3,4,7 and 8.

.rmdir

- Removes a directory
- Pre-requisites to remove a directory:
    - Must be empty before it is deleted
    - Should not be the current directory or a directory at a higher level
    - The directory name should exist
    - Must specify at least one argument with the command
    - Should not be the HOME directory of the user

.This command is used to remove directories.

```
$ rmdir          <pathname>
```

Example:

To remove the src sub directory from the */usr/mano* directory:

$ rmdir src <Enter> will remove the directory, if and only if a directory src exists immediately under a directory student1, which is the working directory and src is empty.

Note: Though it is valid to give a pathname in the rmdir command, it is recommended that rmdir should be only from the parent of the directory to be deleted.

Exercises

I. **Fill in the blanks:**

1. ______________is a way of organizing files by grouping them together.

2. File names in UNIX can be upto ____________characters long

3. The directory assigned in default to each user on logging in is known as_____________.

4. The current directory can be changed with______________command.

5._______________ command is used to create a directory.

6. ________________ command is used to display the path to the current working directory.

7.________________ command removes a directory file.

8. The ________________ command is used to display the files in the directories.

9. The _________________ command displays the directory listing in the long format.

10. ________________ command lists all the files which begin with a dot (.)

II. State true or false:

1. The shell executes the commands issued by the user.
2. The hierarchical filesystem was first used in the UNIX system.
3. The kernel acts as an interpreter for commands given by the user.

4. User can interact directly with the kernel without the help of the shell.
5. The type of the file can be obtained from the directory listings.

Practical session:

1. Login to the UNIX system.

2. Display the path to and name of your HOME directory.

3. List the names of all the files in your home directory.

4. Using the long listing *format* to display the files in your directory. Observe the output.

5. List the files in the columnar format.

6. List the files having 2, 7, 6, u, Y or Z as a character in their names.

7. Try to. list the files whose name is starting with a '-.' (hyphen).

8. Give appropriate command to create a directory called C-prog under your home directory.

9. Create the following directories under your home directory.

    newdir
    newdireetory

10. List the names of all the files, including the contents of the sub directories under your home

directory.

11. Remove the directory called new directory *from* your working directory.

12. Create a directory called temp under your home directory.

13. Remove the directory called newdir under your home directory and verify the above with the help of the directory listing command.

14. Create another directory new under the temp directory.

15. Change the directory to your home directory. What are the ways to perform this?

16. From your home directory, change the directory to directory new using relative and absolute path.

17. Verify that the directories have been renamed.

18. Remove the directory called C_Prog which is in your home directory.

19. Change to the directory /etc and display the files present in it.

20. List the names of all the files that begin with a dot in the /usr /bin directory.