UNIX & Shell Programming

Date : 04/03/24
Page :

Input → [CPU] → Output

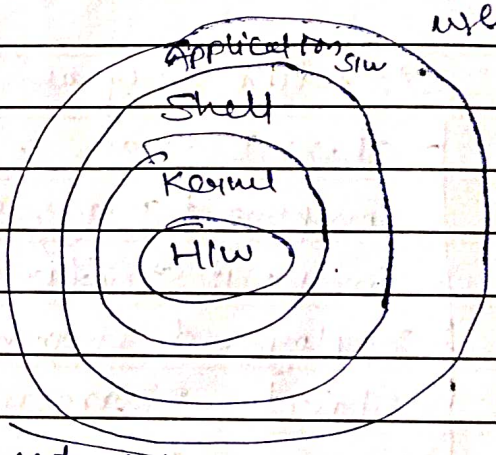
Application S/W
OS
H/W

Software → System software
Application software

OS :- Eg :- Windows, UNIX etc.

UNIX Architecture :- → 1970, Ken Thompson, Dennis Ritchie,

→ Command Line Interpreter
→ Not having G.U.I.
→ Multitasking / Multiuser



(a) Kernel :- (a) core of UNIX OS
(b) Loaded into memory when computer is booted
→ Manages resources
→ Keep track of memory
→ Manages files that carries out data.
→ Security

(b) Shell :- (a) Command Interpreter
(b) User → (commands) Shell → (Interpreter) Kernel → H/W

Shell script :- (1) Frequently used commands in files.
→ many shells out on Kernel

UNIX & Shell Programming

Date : 05/03/24
Page :

(iii) Types of shell :- 1) Bourne shell :- ① Developed by Steve Bourne.
② Simple to use
It is most compact
(iii) comes with every UNIX system

2) C shell :- ① Developed by Bill Joy.
② has advantage over Bourne shell, C syntax
(iv) It comes ~~with~~ has command history

3) Korn shell :- ① Developed by David Korn.
+ most efficient shell
→ lets you edit command line

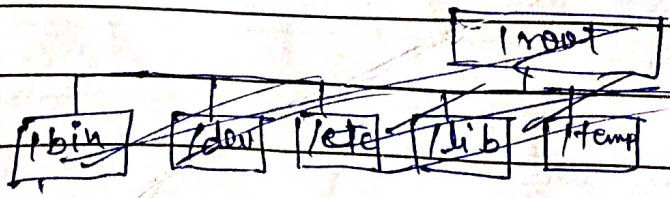
(v) UNIX file system :- File :- collection of related data.

3 types of files :- ① Ordinary file ② Special files
③ Directory files.

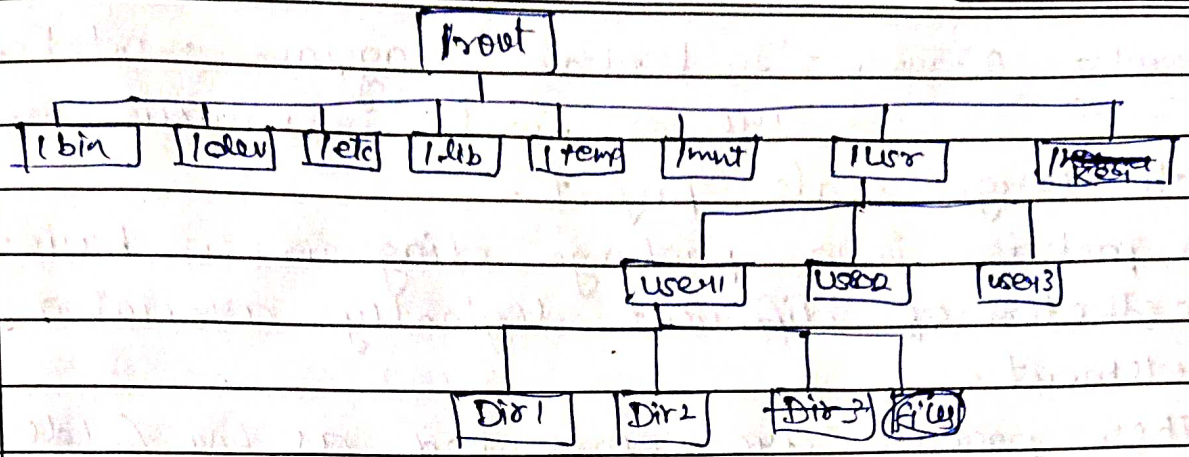
① Ordinary files :- text, binary files, files having attributes (file name, size, access permission, Inode no (unique))

② Special files :- files include physical devices info (hard disk, terminals)

③ Directory files :- contains names of files, information of access, permission.



Date :
Page :



(a) /bin ⇒ unix commands

(b) /dev ⇒ device files [for each device, their is a separate file]

(c) /etc ⇒ additional programs & data files

(d) /lib ⇒ resulable function & routines

(e) /temp ⇒ temporary files

(f) /mnt ⇒ where storage device other than hard disk (floppy, CD) are mounted.

(g) /usr ⇒ home directory for users

(h) /kernel ⇒ kernel specific code.

UNIX & Shell Programming

Date : 07/03/24
Page :

Q. UNIX file system components :-

⇒ UNIX divide physical disk into logical disk. called partitions.

① Each partition is a standalone file system.

⇒ 4 components

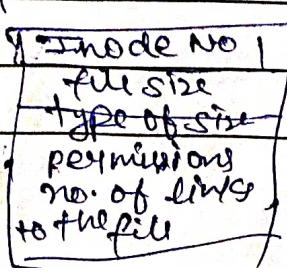
Boot Block	Super Block	Inode Block	Data Block

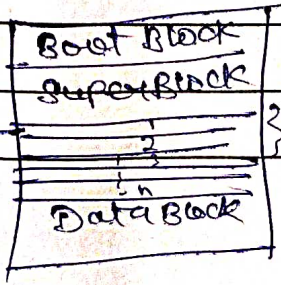
① Boot Block :- It is the first block of UNIX file system. and contain bootstrap programming. It is loaded into memory and executed when boot it up. It is fetched from the boot block of a file system.

② Super Block :- Contains parameters of file system like total size, total no. of data blocks, No. of Inodes, block size of file system.

③ Inode Block :- (Index node) :- When a file is created, (UNIQUE) and inode is also created to keep the information about all the attributes of particulare file. be presents the maximum no. of (owner) in UNIX system.

It stores honor, type, modified time, access permission size of file etc.



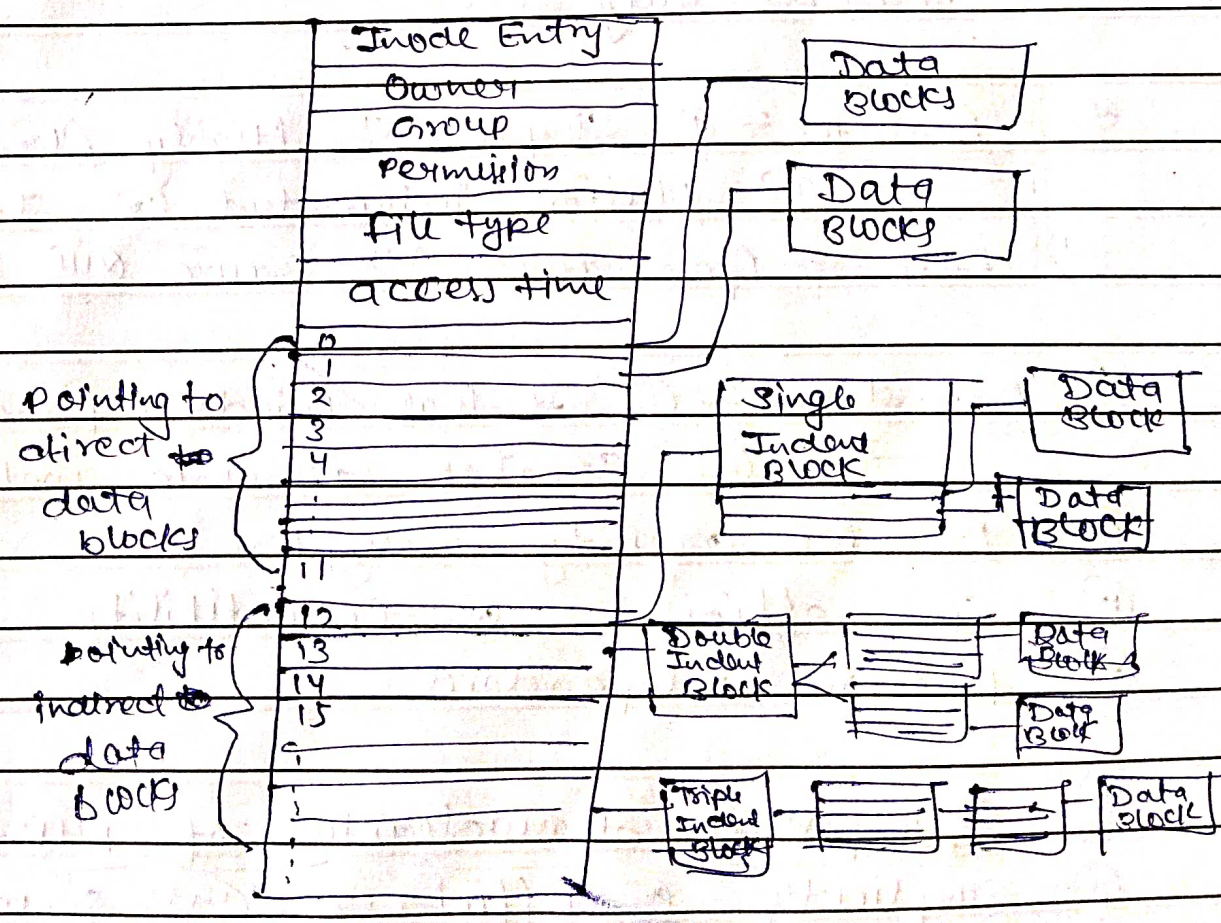


Date :
Page :

(v) Data Block : The Data blocks contains the actual data contain in the files.

The blocks containing the data are called direct blocks. If a file wants to expand it require some more block and if the adjust data block are not free then it has to locate a free space from the disk.

Inode keep track of all direct block addresses but it can store only of them. Therefore file system also contain in direct blocks. (which do not contain the data but contain the address of direct block).



Unix basic commands :-

1. pwd :- present working directory
   $ pwd ↵ (output :- /home/bca
2. who :- displays all the user who have logged into the system.
   $ who :- user1 2024-03-09 9:42

3. mkdir :- make directory
   $ mkdir Science
4. rmdir :- Remove directory (empty)
   $ rmdir science
5. → rm -r <directory name> (to remove non-empty directory)
   (recursive)

6. cd → change directory
   $ cd <directory>
   → cd ..... → to come out
   → cd / → to reach upto the root
7. touch :- create file with zero bytes
   Syntax $ touch <filename>

8. cat :-
   Syntax : $ cat > <filename>
   cat > file1
   to exit press [ctrl+d]

→ cat file1 } display content of file1
cat < file1 }

Date :
Page :

→ cat file1 file2

→ cat file1 > file3
content of file1 also gets into file3.

→ cat > test << BCA → EOF (End of file) (End marker)
> Hello unix
> Hello BCA 6th sem
> BCA ↵
The movement you will get BCA at new blank line it will stop and give you to the entered text.

→ append file :- cat > file1
hello unix (ctrl+d)

(2) cat >> file1
hello java (ctrl+d)

(3) cat file1
→ Hello unix
hello java.

→ remove file :- rm <filename>
rm -i <filename>
↳ interactively removing
remove <filename> ? Y/N

→ ls → listing

→ ls -l → long format (detailed information about file & directory)

→ ls -a → represents all files included hidden files

UNIX & shell Programming

Date : 11/03/24
Page :

ls :- ① ls -l ② ls -a

ls -t → short files or directories by there last modification time, displaying the most recently once first

$ ls -t

ls -r :- Display the content in reverse order.

ls -s :- listing along the size of file.

ls -R :- Display file & directories recursively, including subdirectories.

ls -i :- Display files with inode no.

ls -n :- which user created UserID

wc :- count the words, lines and characters.

Syntax :- wc <filename>

eg :- wc file1 ↵

output :- 2 8 35 file1

wc -l <file name> → no. of lines

wc -w <file name> → no of words

wc -c <file name> → no. of bytes

wc -L <file name> → print longest line length.

wc file1 file2

output :- 2 8 35 file1

3 15 40 file2

5 23 75 Total

head :-

Syntax → $ head -n <filenames>

↳ no. of top line in the file

Date :
Page :

eg:- head -2 file1
↳ display top 2 lines

without any option it displays only first 10 lines in the file

$ head file1 file2

output:- ⇒ file1 ⇐
top 10
⇒ file2 ⇐
top 10

(ii) $ head -q file1 file2
↳ quiet

output:- file1 + file2 together

⇒ tail

$ tail -n <file name>
↳ no. of bottom/last lines in the file

⇒ cp :- to copy a file

Syntax:- $ cp <source file> <destination file>
If destination file doesnot exist then it is created.
Or if destination file already exist it is over written without any warning.

eg:- $ cp file1.txt file2.txt

Date :
Page :

→ copy files to directory :-

$ cp f1.txt f2.txt

→ copy directories :- $ cp -R <source.dir> <destination.dir>
↓
Recursive

$ cp -R dir1 dir2

if destination directory exist it copies all the contents
= if destination directory doesn't exist cp command creates it and copies the content recursively

⇒ mv (move) — move files from to another.

① renames files & directories

Syntax :- $ mv <source.file name> <destination.name>

eg :- ① $ mv f1.txt f2.txt (renaming of file)
$ mv dir1 dir2 (renaming of directories)

⇒ move a file to destination path
$ mv f1.txt f2.txt /home/bca/dir1/

Date : 12/03/24
Page :

Comparision b/w two files :-

1) cmp (compare) :- use to compare two files byte by byte and helps to you find whether two files identical or not.

eg :- f1.txt (This is a pen)
f2.txt (This is a pan)
$ cmp f1.txt f2.txt
output :- f1.txt f2.txt differ : char 9, line 2

If the files are identical :-
$ cmp f1.txt f3.txt (output + $ - - - -)

2) comm :- (common) :- Compare to sorted files line by line and write the output in 3 columns.

1st col	2nd col	3rd col
lines unique to first file	line unique to second file	line common to both files

eg :-
$ cat > f1
amit
deepak
priyanka

$ cat > f2

Date :
Page :

ajay
deepak
deepika (ctrl+d)

⇒ $ comm f1 f2 ↵
output: amit ajay deepak
priyanka deepika

-1 : Supress 1st col
-2 : " 2nd col
-3 : " 3rd col

$ comm -3 f1 f2 ↵
amit ajay
priyanka deepika

$ comm -1 2 f1 f2 ↵
deepak

3. diff (difference) :- It will display each and every difference of both files

⇒ $ diff <file1> <file2>
$ diff . f1 f2 ↵

output:
|c|
<amit
ajay>
3c3
< priyanka
deepika>

c → change
< → 1st file
> → 2nd file

Date :
Page :

(1) vi editor ⇒ visual editor (i). edit existing file (ii). create new file (iii). read a text file

(vim)

(2) open vi-editor

$ vi <file name> ↵

UNIX & Shell Programming

Date : 13/03/24
Page :

Vi - modes :- (i). Command mode :- When vi starts a it is in command mode

command-mode performs commands and navigate through content in file. In this mode you can move the cursor, delete text, search text and save a file..

eg:- $ vi file1↵

① press [esc] to ~~command~~ change mode

(ii) Insert mode / Input mode :- permits the insertion of new text, editing ~~and~~ the existing text. To come into insert mode. type 'i' (to come out from this mode, [Esc])

(iii) Ex-mode / Ex-command mode / Last Line mode :-

The bottom line of vi editor is command line. It is invoke by typing (:) esc

⇒ Steps :- → Press 'i' to move into insert mode.
→ Press Esc
→ :wq to quit & save
→ :q to quit

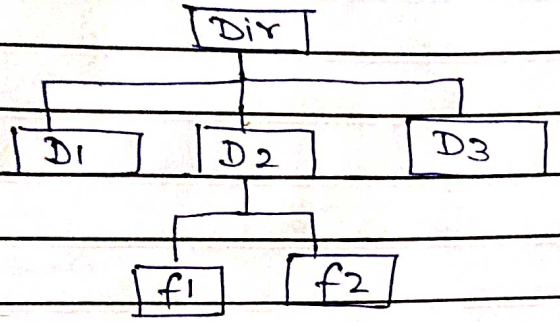
1) WASS that show the output like.
Today date is
The ~~cal~~ calender is

write a shell script

Date :
Page :

→ $ vi ss1.sh ↵
→ Press 'i'
→ echo "Today date is "
→ date
→ echo "The calender is"
cal (esc) → :wq
→ run $ sh ss1.sh ↵

2) WASS to create a structure



mkdir Dir
cd Dir
mkdir D1 D2 D3
cd D2
touch f1 f2

3) Enter marks of Hindi and English of a student and display them

echo "Enter the marks of Hindi"
read hindi
echo "Enter the marks of English"
read English
echo "Enter the marks of Hindi is $Hindi"

Date :
Page :

echo "Enter marks of English is $eng"