1DB18IS098
Vivek. V. Rao
5th B

Unix assignment - 2.

① What is a file? Explain different types available in UNIX or POSIX system. Also write the commands to create all the files.

→ File is a collection of records, so files are divided into 3 catagories

① Ordinary file: it contains only data as a stream of characters.

* Text file: contains only printable characters and you can often view the contents & make sense out of them. All c and Java files are example of text file.
* Binary file: it contains both printable & unprintable characters that cover the entire ASCII range (0 to 255)

② Directory file:

* contains no data, but keeps some details of the files & subdirectory that its contains.
* A directory file contains an entry for every file & sub directory that it houses. Each entry has two components.
* The file name
* A unique identification number for every file & directory.

③ Device file:

- used to represent a real physical device such as a printer, tape drive or terminal, used for I/O operations.
- Unix considers any device attached to the system to be a file - excluding your terminal:
- stdin and stdout will be discussed in more detail later
- usually only found under directories named /dev

(8) Explain the chown, fchown & lchown functions;

→ chown.

- This command can be used to change the ownership of a file.
- Syntax.

  chown username filename
- changing ownership requires superuser permission, so use su command $ su

  Password: ********
- after the password successfully entered, su returns a # prompt, same prompt used by root su lets acquire

  Eg:-

  $ ls -lnote

  -rwxr---x 1 Rumar metal 347 may 10

  $ chown sharma note

  $ ls -l note.

  -rwxr---x 1 sharma metal 347 may 10 20:30 note

fchown:

- The fchown() function has the same effect as chown() expect the file whose ownership is to be changed is specified by file descriptor rather than path name.

.fchown.

- The lchown() function has the same effect as chown() expect in the case where the named file is a symbolic link. In this case, lchown() changes the ownership of the symbolic link file itself

(1)

(3) Explain the following API's along with prototypes:
(i) open (ii) fcntl (iii) seek (iv) stat & fstat

→ (i) Open

- open() system call open a file for reading writing or reading and writing. The prototype for the open() system call is

```
int open (file_name, option - flags [mode])
char * file_name;
```

Algorithm for opening file

Input : file name
Type of open.
file permission

(ii) fcntl() function:

- This system performs file control and I/O control on the descriptors. It is an interface to the fcntl() & ioctl() unix routines.
- The prototype for the fcntl function is.

```
#include <unistd.h>
#include <fcntl.h>
```

(i) F_DUPED (ii) F_GETFD (iii) F_SETFD
(iv) F_GETFL (v) F_SETFL (vi) F_GETOWN
(vii) F_SETOWN

(iii) seek function:

- Read and write operations normally start at the current file offset to be incremented by the number of bytes read or written.
- The syntax for seek system call is position = lseek (fat, offset, reference);

fd → file description
offset → byte offset.
reference → indicates whether offset should be considered from the beginning of the file.

- The lseek system calls has nothing to do with the seek operation the positions a disk arm over a particular disk sector

(iv) Stat and fstat

- These two system call allows process to query the studies of files, returning information such as the file type, file owner, access permissions, file size, number of link, inode number & file access time

Fstat() is identical to stat(), except that the file to be stat-ed is specified by the field descriptor filedes

(4) Explain the chmod & fchmod functions:

→ * A file or a directory is created with a default set of permissions, which can be determined by umask
* To know the system's default permission create a file
cat > csc [ctrl d]
$ ls -l csc.
-rw-r--r-- (default permission for created file)
* The chmod (change mod) command is used to set the permissions of one or more filed for all 3 catogaries
* The command can be used in 2 ways:
* In a relative manner by specifing the change to the

(3)

current permissions.

* In an absolute manner by specifiy the ~~finalpermiss~~ finalpermissions.

Eg:- $ ls -l xstart

-rwo-r-- 1 Kumar metal 1906 Sep 23:38 xstart

here user is having the only read and execute permission using relative file permission need to add the execute permission to user

chmod category operation (-, +) permission filename

$ chmod u + x x start

$ chmod u+x start

$ ls -l x start

-rwxr--r-- 1 kumar metal 1906 Sep 23:38.

- Fchmod : The fchmod() function has the same effect as chmod() except that the file whose permissions are to be changed is specified by file descriptor rather than path name.

```
#include <sys/types.h>
#include <sys/stat.h>
int chmod(const char *pathname, mode_t mode);
int fchomd(int fides, mode_t mode);
```

(5) Explain the here command with an ~~explain~~ example;

→ * The << symbol can be used to read data from the same file containing the script. The file is called as a ~~be~~ here document

- The term 'here' signifies that the data is here rather than in the file
- syntax.

command << delimiter
document
delimiter.

Eg:

```
$ mailx kumar << MARK
explore
Dream
Discover.
MARK
```

- The string (MARK) is delimiter.
- The shell treads every line data delimited by MARK as input to the command mailx.

Eg:

```
$ wc -l << FEND
Decide.
commite
succeed.
end.
FEND
3                    // outputs number of lines = 3.
```