Assignment 7a

→ **Aim:** Interprocess Communication in Linux using pipes.

→ **Objectives:**

Implementation of Full Duplex communication between parent & child processes. Parent process writes a pathname of a file on one pipe to be read by child process & child process writes the contents of ~~the~~ on the second pipe to be read by parent process & display on standard o/p.

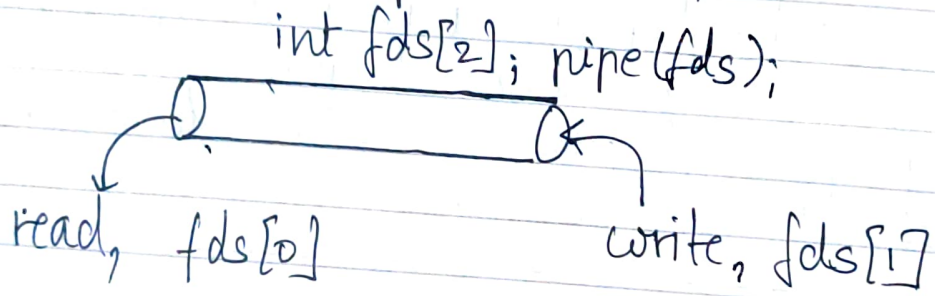
→ **Theory:**

Pipes:

- Pipe create an inter-process channel.
- The pipe() function shall create a pipe & place 2 file descriptors, one each into the arguements field [0] & field [1], that refer to the open file descriptors for the read & write ends of the pipe.
- Data can be written to the file descriptor field [1] & read from file

descriptor [0].

- A pipe can be ~~used~~ used in Unix command line, the first process is assumed to be writing to stdout & second assumed to be reading from stdin. So it is common practice to assign the pipe write device descriptor to stdout in the 1st process & assign the pipe read device descriptor to stdin in the 2nd process.



- For Full-Duplex communication we create 2 different pipes.
- IPC between child & Parent Process.

```
...
int fileds[2];
const int BSIZE = 100;
char buff[BSIZE];
ssize_t nbytes;
int status;
status = pipe(fileds);
switch(fork()) {
case -1: break
case 0: close fileds[1];
        read(fields[0], buf,
             BSIZE);
        close fileds[0];
        break;
defaut: close fileds[0];
        write(fileds[1], "Hello", 5);
    } close fileds[1];
} return 0;
```

→ Conclusion :

Topics Covered :

1. Pipes.
2. Half - Duplex & Full Duplex Communication
3. IPC between parent & child.