Question :- Who is supporting pipelining mechanism ?

⟹ bash

Pipelining :-

It's mechanism between two commands.

⊛ Advantage of pipelining :-

① The cycle time of the processor is reduced.

② The pipeling is used then CPU Arithmatic Unit can be designed faster.

③ It increase computers overall performances.

⊛ disadvantages of pipelining :-

① pipelining involves adding hardware to the chip.

② Inability to continuosly run the pipeline at fullspeed

⊛ Advantage of dup() :-

⊛ $ man fcntl

fcntl → manipulate file discriptor.

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

int fcntl (int fd, int cmd, ---- /* arg */);
```

→ fcntl() performs one of the operations described bellow on the open file descriptor fd. The operation is determined by cmd.

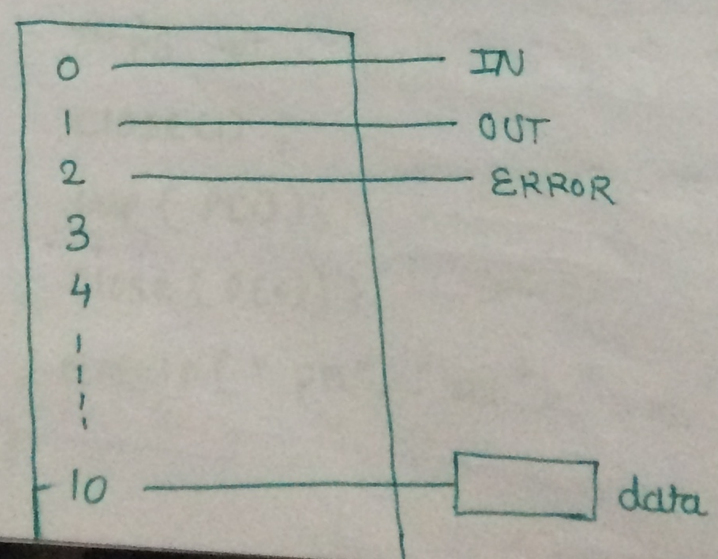
→ fcntl() can take an optional third argument whether or not this argument is required is determined by cmd.

> aim of the fcntl() is locking and Unlocking mechanism of file.

① F_DUPFD (int) :-

Find the lowest numbered available file discriptor greator than or equal to arg and make it be a copy of fd. This different from dup(a) which uses exactly the discriptor specified.

On success → new file discriptor is returned.



eg:- fd1 = fcntl (fd, F_DUPFD, 10);

o/p :- FD1 = 11

② F_GETFD :- (Void)

⇒ Read the file discriptor flags ; arg is ignored.

③ F_SETFD (int) :-

⇒ Set the file discriptor flags to the Value specified by arg.

④ F_GETFL (Void) :-

⇒ Get the File access mode and file status flags, arg is ignored.

⑤ F_SETFL (int) :-

⇒ Set the file status flags to the Value specified by arg.

⇒ File access mode (O_RDONLY, O_WRONLY, O_RDWR) and File creation flags (O_CREAT, O_Excl, O_NOCTTY, O_TRUNC) in arg are ignored.

⑥ Advisory locking :-

⇒ F_GETLK, F_SETLK and F_SETLKW are used to acquire, release and test for the existence of record locks (also known as file-segment or file region locks).

F_GETLK = Getting Information

F_SETLK = Set the lock.

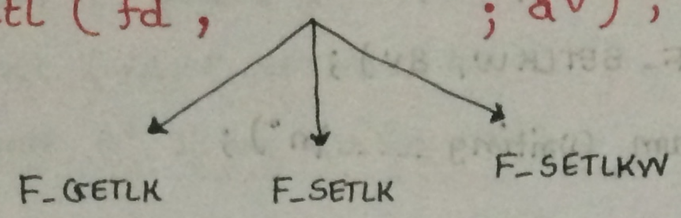
F_SETLKW = Set the Lock and lock is available if not wait for lock availability.

Program :-

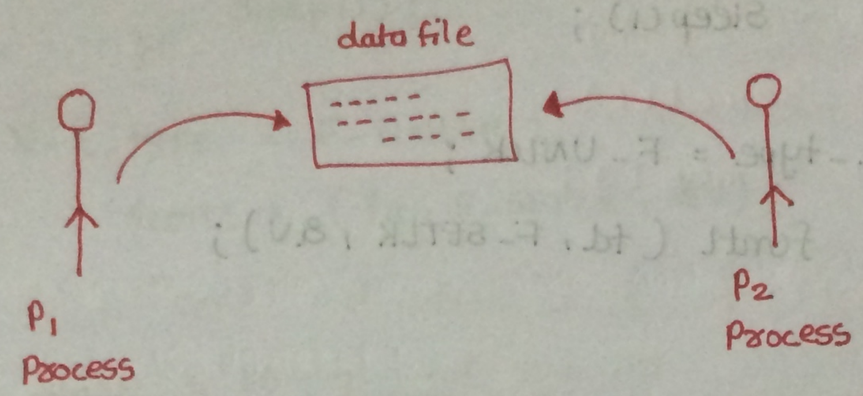
```
# include " header.h "
main()
{
    char ch = 'a' ;
    int  p[2] , c = 0 ;
        pipe(p) ;
fcntl ( p[1] , F_SETFL , O_NONBLOCK) ;
        while ( write ( p[1] , &ch , 1) != -1)
                c++ ;
Printf ( " %d \n", c) ;
Printf ( " Bye---- \n ") ;
}
```

```
Struct flock
{
    . . .
    Short l_type ;  // type of lock : F_RDLCK, F_WRLCK,
                                      F_UNLCK
    short l_whence ; // How to interpret l_start
                        SEEK_SET, SEEK_END, SEEK_CUR.
    Off_t l_start ;  // Starting offset of the lock.
    off_t l_len ;  // number of bytes to lock.
    Pid_t l_pid ;  // Pid of process blocking our
                                lock
                        (F_GETLK only).
    . . .
};
```

fcntl (fd , ; &v) ;



Program :-



Process W1.c

```
# include " header .h "
main ()
{
    Struct flock V;
    Char a[] = " Savan. " ;
    int fd ;
        fd = open (" data " , O_RDWR | O_APPEND | O_CREAT,
                                                  0644) ;
        if ( fd <0)
        {
            Perror (" open ") ;
            return ;
        }
    V.l_type = F_WRLCK ;
    V.l_whence = SEEK_SET ;
    V.l_Start = 0 ;
    V.l_len = 0 ;
```

```
printf ("Before ----\n");
fcntl ( fd, F_SETLKW, &v);
Printf (" I am Writing -----\n");
    for (i=0 ; a[i] ; i++)
    {
        write (fd, &a[i], 1);
        Sleep(1) ;
    }
V.l_type = F_UNLCK ;
    fcntl (fd, F_SETLK , &v);
}
```

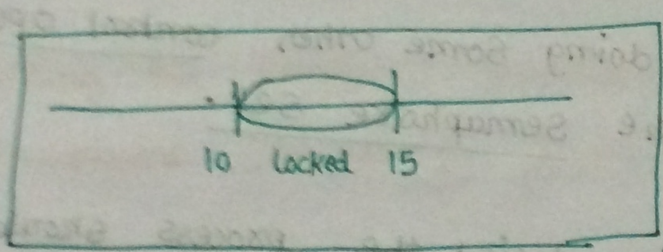
Process W2.c

```
#include " header.h"
main()
{
    Struct flock V;
    char a[] = " Bhalodiya " ;
    int fd, i ;
        fd= open ("data", O_RDWR | O_APPEND | O_CREAT,
                                        0644);
    if(fd<0)
    {
        Perror ("open") ;
        return;
    }
V.l_type = F_WRLCK ;
V.l_whence = SEEK_SET ;
V.l_Start = 0 ;
V.l_len = 0 ;
```

```
Printf (" Before ----\n");
fcntl (fd, F_SETLKW, &V);
Printf (" I am writing \n");
        for (i=0 ; a[i] ; i++)
        {
                write (fd, &a[i], 1);
                Sleep(1);
        }
    V.l_type = F_UNLCK;
        fcntl = (fd, F_SETLK, &V);
}
```

V.l_Start = 10
V.l_len = 5

means



data file

Start from 10 and len=5 that bytes are locked.

⇒ fcntl() is not used in shared memory because shared resource has not file discriptor fcntl (fd, --, --) fd is required.

⇒ lseek and fseek both are same [in lseek file discriptor] - [In fseek file pointer.]

**** Shamaphore ****

It is a small positive integer which is present in a kernel space and which is used to synchronize between multiple process when they are accessing shared resource.

1) semget();
2) semctl();
3) semop();

① semget() :- It is useful for creating or as well as opening the shemaphore set.

② semctl() :- It is useful for setting the semaphore value and getting the semaphore value and doing some other control operation on the semaphore set.

③ semop() :- It is desied the process should enter into the critical section or not. (It is also blocking function).

- In the place of semaphore, fcntl not used.
- In the place of fcntl, semaphore is used.