TP 4

SYSTEMS D'EXPLOITATION

DÉLIVRABLES

- Vos réponses dans un rapport PDF
- Code source commenté et fonctionnel
- Dossier à rendre : <Prenom1>_<Nom1>.<Prenom2>_<Nom2>.TP4.zip(ou tar.gz)

INFORMATIONS GÉNÉRALES

- Références au cours :
  - 7. Entrées / Sorties (I/O)
- Objectifs :
  - poser un verrou et l’enlever sur une partie d’un fichier;
  - comprendre les différents types de verrous

FLOCK

- Poser un verrou sur un fichier

Utilisation:

- flock(fd, cmd, … args …)
  - fd
  - cmd: LOCK_SH, LOCK_EX, LOCK_UN

FCNTL

- Poser un verrou sur une partie d'un fichier

Utilisation:
fcntl(fd, cmd, &flock)

- fd
- cmd: F_GETLK, F_SETLK, F_SETLKW

```
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_CUR, SEEK_END */
        off_t l_start;   /* Starting offset for lock */
        off_t l_len;     /* Number of bytes to lock */
        pid_t l_pid;     /* PID of process blocking our lock
                            (set by F_GETLK and F_OFD_GETLK) */
        ...
    };
```

EXEMPLE D'UTILISATION

```
$ ./your-programme bla.txt
Enter ? for help
PID=258> ?

    Format: cmd l_type start length [whence(optional)]
    'cmd' ---  'g' (F_GETLK), 's' (F_SETLK), or 'w' (F_SETLKW)
    'l_type' --- 'r' (F_RDLCK), 'w' (F_WRLCK), or 'u' (F_UNLCK)
    'start' --- lock starting offset
    'length' --- number of bytes to lock
    'whence' --- 's' (SEEK_SET, default), 'c' (SEEK_CUR), or 'e'
(SEEK_END)

PID=258> s w 0 5
[PID=258] got lock
```

EXTRAIT DE CODE

```
  for (;;) {          /* Prompt for locking command and carry it out */
     printf("PID=%ld> ", (long) getpid());
     fflush(stdout);

     // use fgets to read user input and then handle it

     // process user unput into the 'cmd' variable and the various elements of 'fl' struct

     status = fcntl(fd, cmd, &fl);           /* Perform request... */

     // interpret results of request and inform user
     if (cmd == F_GETLK) {                  /* F_GETLK*/
        // check status and handle errors (look at manual for possible errors)
        if (status ==0 ){
           // process results and print informative text
        }else if (errno == SOME_ERROR){
           // process results and print informative text
        }

     } else {               /* F_SETLK, F_SETLKW */
        // check status and handle errors (look at manual for possible errors)
        if (status ==0 ){
           // process results and print informative text
        }else if (errno == SOME_ERROR){
           // process results and print informative text
        }
     }
}
```