Sujet du TP n° 2

Préambule

Les ressources du cours (supports, exemples, et autres documents) et des TP (les sujets, les fichiers de travail ainsi que les corrigés) se trouvent à l'URL : http://www.emse.fr/~lalevee/ismin/pse, dénommé **[site]** dans les documents fournis.

Ce TP suppose que vous avez installé l'archive **PSE.tar**. Si ce n'est pas le cas, consultez le sujet du premier TP.

Placez-vous dans le répertoire **PSE/TP2**.

Entrées-sorties disque (transparents 47 à 51)

Exercice 1

- copiez le fichier **exemples/slide051.c** et renommez-le **exercice1.c.**
- compilez ce fichier et testez-le.
- ajoutez des commentaires là où une entrée-sortie est effectuée, en particulier indiquez des exemples d'erreur d'entrée-sortie. Consulter **man** pour avoir des précisions.
- modifiez-le afin qu'il affiche le nombre d'octets lus et écrits.

Exercice 2

- écrivez un programme **exercice2.c** qui lit un fichier et qui l'éclate en fichiers de taille fixe, en nommant ces fichiers **F001**, **F002**, etc.
- la première lettre (ici **F**) et la taille seront données en paramètre du programme.

note : si **lettre** contient la première lettre, **numfichier** contient le numéro du fichier, et **chaine** est un tableau de 5 caractères, pour générer le nom de fichier, vous pouvez utiliser :

```
sprintf(chaine, "%c%3.3d", lettre, numfichier);
```

Voici quelques commandes qui vous permettront de tester votre programme :

```
$ ./exercice2 exercice2.c 100 F
$ cat F* > FICHIER
$ diff -s exercice2.c FICHIER
Les fichiers exercice2.c et FICHIER sont identiques.
```

La première éclate le contenu du fichier source **exercice2.c** en fichiers nommés **F001** à **Fnnn** de 100 octets chacun, sauf éventuellement le dernier. La seconde regroupe tous ces fichiers (opération inverse) dans le fichier **FICHIER**. La dernière enfin vérifie que le fichier source **exercice2.c** et le fichier après fusion sont identiques.

les tubes nommés (transparents 52 à 54)

Pratique 1

- créez un tube nommé **fifo** en utilisant la commande **mkfifo** (consulter le manuel pour connaitre les options à indiquer), de telle manière que le propriétaire et les membres de votre groupe puissent lire et écrire.

- retrouvez le type de fichier avec la commande **ls -l**.

Pratique 2

- pour illustrer le principe des tubes, tapez la commande **cat < fifo** dans une première fenêtre **terminal**
- ouvrez une seconde fenêtre **terminal** et placez-vous dans le même répertoire courant que celui de la première fenêtre
- tapez la commande **cat exercice2.c > fifo**
- expliquez ce qui s'est passé.

Lisez-moi 1

- nous allons utiliser un module de la bibliothèque
- il s'agit du module **ligne**, qui lit des lignes de texte à partir d'un fichier, et donc en particulier d'un tube nommé.
- une ligne de texte est définie comme une chaine de caractères de taille maximale **160** (voir constante **LIGNE_MAX**), terminée par un **'\n'**.
- consultez le fichier **ligne.h** présent dans le répertoire **include**.
- consulter le fichier **ligne.c** présent dans le répertoire **modules**.
- la fonction **lireLigne()** lit des octets à partir du fichier spécifié par le descripteur (ce fichier doit être ouvert avant l'appel de la fonction) et les stocke dans le buffer indiqué jusqu'à rencontrer le caractère fin de ligne
- la fonction **ecrireLigne()** est l'opération inverse.
- ces fonctions retournent le nombre d'octets lus ou écrits, ou -1 en cas d'erreur. Pour **lireLigne()**, **0** signifie fin de fichier.

Exercice 3

- écrivez un premier programme **serveur.c** qui lit des lignes de texte à partir du tube nommé **fifo**
- ce programme écrit ensuite ces lignes à la fin du fichier **journal.log**.
- certaines lignes de texte ont une signification particulière :
  "fin" : arrêt du programme
  "init" : initialisation du fichier **journal.log** (remise à zéro)

Pour tester le serveur, vous pouvez utiliser le **shell** comme client. Pour cela, dans votre terminal, tapez des commandes comme celles-ci :

```
$ ./serveur &
$ cat exercice2.c >fifo
$ ...
Serveur. ligne de 22 octets ecrite dans le journal.
Serveur. ligne de 2 octets ecrite dans le journal.
$ less journal.log
$ echo init > fifo
$ Serveur. remise a zero du journal demandee.
$ echo fin > fifo
Serveur. arret demande.
```

Exercice 4

- écrivez ensuite un programme **client.c** qui lit des lignes au clavier avec la fonction **fgets()** et les envoie au **serveur** par l'intermédiaire du tube nommé **fifo**.
- il s'arrête soit en fin de fichier (appui de **CTRL-D**), soit lors de l'envoi de la ligne **"fin"**.

Question 1

- lancez et utilisez plusieurs clients dans des fenêtres différentes
- que constatez-vous ?

Question 2

- que se passe-il quand vous tapez **"fin"** dans l'un d'entre eux ?
- pour corriger ce problème, ajouter l'instruction suivante en début du programme client :

```
signal(SIGPIPE, SIG_IGN);
```

(c) Philippe Lalevée, 2013-2014.