ENSEA
Ecole Nationale
Supérieure
de l'Electronique
et de ses Applications

2e année – Majeure Informatique

TDm 2 : Jouons avec les Files

C. BARÈS

Les manipulations proposées dans ce TDm sont à réaliser sous GNU/Linux. Référez-vous en permanence aux pages de manuel des fonctions utilisées, section 2 ou 3. Pour accéder à ces pages vous pouvez :

- les rechercher sur internet, par exemple sur `http://man7.org/linux/man-pages/`
- taper dans un terminal : `man 2 stat` pour la 2e section du manuel de `stat`.

Les sections qui vont nous intéresser en programmation système sont :

2 → Appels systèmes
3 → Fonctions de la bibliothèque C standard

Remarque : Évitez les pages de manuel en français, elles ne sont pas à jour...

1 – Fichier

1.1 stat

Utilisez l'appel système `stat` pour déterminer la taille du tampon à utiliser lors des opérations d'E/S sur :

- le disque dur ;
- la console.

Indice : Il y a un exemple à la fin de la page de manuel de `stat`.

1.2 my_cp

Créez en C un programme `my_cp` qui prend en argument 2 noms de fichier :

```
$ my_cp fichier1 fichier2
```

et qui a le comportement suivant :

- `fichier1` doit être un fichier ordinaire ;
- `fichier2` ne doit pas exister ;
- si ces 2 conditions sont vérifiées, alors le `fichier1` est copié vers `fichier2` ;
- si `fichier2` n'est pas donné à la ligne de commande, alors `fichier1` est affiché dans la console.

Indice : Il faut utiliser les appels systèmes `stat`, `open`, `read` *et* `write` *, ainsi que le descripteur de fichier* STDOUT_FILENO.

2 – Tic & tac

2.1 Tic

Créez un programme qui prend en argument un nom de fichier et qui affiche à l'écran son contenu à l'envers, du dernier octet au premier.

2.2 Tac

Créez un programme qui prend en argument un nom de fichier et qui affiche à l'écran son contenu à l'envers, de la dernière ligne à la première. Pour optimiser le traitement, vous pouvez effectuer une première passe pour rechercher la position de chaque `'\n'`.

3 – FIFO 20

3.1 En bash

Créez un fichier fifo depuis le bash. À l'aide des commandes `ls` ou `stat`, relevez ses caractéristiques.

Ensuite, à l'aide des commandes `echo` ou `cat`, envoyez du contenu dans cette fifo (en utilisant la redirection « `> mon_fichier_fifo` »).

Depuis un 2e terminal, affichez le contenu de la fifo. Conclusion ?

4 – Fourchettes et macaroni (Si il reste du temps)

4.1 fork 1 – pipe 1

Mettez en place un pipe entre un processus père et son fils. Le père lit sur l'entrée standard et l'écrit dans le pipe, le fils lit dans le pipe et écrit sur la sortie standard.

Quand le père arrive à la fin de fichier (`<ctrl>-d`), il envoie un signal `SIGKILL` au fils qui se termine, puis attend la mort du fils.

4.2 fork 1 – pipe 2

Reprendre le programme précédent, mais maintenant, le fils compte le nombre de caractères envoyés par le père, et lui retourne ce nombre par un 2e pipe à chaque fois que le père transmet.

4.3 fork 2 – pipe 2

Reprendre le programme précédent, mais maintenant le travail du père est effectué par un 2e fils. Le père se contentera de surveiller la fin de ses 2 fils.