TP № 4: Tubes

Samir Aknine, Antoine Gréa & **Rémy Grünblatt** 22/04/2020

Compétences à acquérir:

- Savoir faire un tube, en C
- Connaître les opérations de duplication de descripteurs de fichier

Information: Il est extrement conseillé d'avoir recourir à une machine sous GNU/Linux (par exemple, Ubuntu). Il est possible d'utiliser `https://shell.univ-lyon1.fr` en secours, en ouvrant plusieurs terminaux (notamment dans le cas du processus zombie), mais cela ne devrait pas être la première option pour vous.

Important: Tous les TPs feront l'objet de rendus, deux semaines après le TP, et seront notés; les modalités de rendu seront précisées par email ou sur le chat discord de la classe. Il convient donc de bien noter les commandes exécutées, par exemple dans un fichier texte ou markdown, et leur résultats, afin de préparer le rendu du TP.

Exercice 1: Faire un tube

Information: Sous Unix, le système se base sur les fichiers pour la presque totalité des opérations (on peut souvent lire « sous Unix, tout est fichier »). Par exemple, les processus possèdent des descripteurs de fichiers spéciaux qui sont l'entrée standard, la sortie standard, la sortie d'erreurs. Ces descripteurs peuvent être connectés à l'aide d'un opérateur appelé « tube » (ou « pipe » en anglais). En shell, lorsque l'on écrit `cat fichier | grep "machin"`, la barre verticale représente un tube, et permet de rediriger la sortie standard de la commande `cat` vers l'entrée standard de la commande `grep`. Les données transportées par un tube sont appelées « flux » (stream, en anglais). On peut utiliser les constantes `STDIN_FILENO`, `STDERR_FILENO` et `STDOUT_FILENO` pour faire référence à `stdin`, `stderr` et `stdout` (ou 0, 2 et 1 respectivement).

Question 1: Étude d'un programme

Étudier, compiler le fichier « `tube.c` », et expliquer son fonctionnement, en particulier le comportement de `write` et de `read` en fonction de l'ordre d'exécution.

Question 2: Perroquet

Créer un programme `perroquet` qui lit l'entrée standard (par exemple, avec la fonction `getchar`) et qui les envoie à un autre processus qui les affiche (pour ce dernier processus, vous connaissez normalement déjà un programme qui permet d'afficher son entrée standard).

Question 3: Double Rainbow

Modifier le programme `tube.c` pour que le père affiche « Noooooooonnnnn ! » et le fils affiche « Je suis ton père », en utilisant un unique pipe. À quoi doit-on bien faire attention ?

Exercice 2: Duplicata

Information: L'appel système `dup` (pour duplicate) permet de dupliquer un flux vers un autre descripteur de fichier. Cette primitive permet de créer un alias d'un descripteur de fichier vers le premier descripteur libre (variante `dup`), ou bien vers un descripteur spécifié (variante `dup2`), un peu à la manière d'un lien. Une fois dupliqué, on peut utiliser indifféremment les deux descripteurs qui font référence au même fichier.

Le programme `self` a besoin du contenu du fichier `number.txt` sur son entrée standard pour s'exécuter correctement. Cependant, il s'avère que le fichier contient des caractères qui ne sont pas des chiffres...

Question 1: Nettoyage

Réaliser un programme qui affiche (sur la sortie standard) le contenu du fichier dont le nom est passé en paramètre, en enlevant tout ce qui n'est pas un chiffre.

Question 2: Redirection

Modifier ce programme pour rediriger sa sortie standard dans un tube dont la sortie sera connectée à un processus fils qui exécutera le programme `self` et lui passera les données sur son entrée standard.

Question bonus: Shell

Refaire les deux questions précédantes en shell (un *one-liner* evidemment).