TD : Les tubes

Objectifs :

Utilisation des tubes anonymes ou nommés pour échanger des données entre deux ou plusieurs processus, en filiation ou indépendants

---

Exercice1 :

Écrire un programme permettant de réaliser une communication entre un processus père et son fils.

Le processus père lit des caractères au clavier et les communique à son fils afin que ce dernier les affiche à l'écran. Le fils n'affichera que les caractères alphabétiques.

Exercice2 :

Modifier le programme de l'exercice 1, afin que le fils compte le nombre de caractères alphabétiques, les affiches en majuscules et renvoi ce nombre au père.

Exercice3 :

Le but de cet exercice est de faire communiquer de façon symétrique deux processus fils entre eux. Le père ne communique pas avec ses fils. Pour réaliser cela le processus père doit :

- ✓ Créer deux tubes,
- ✓ Créer le fils ***FilsA,***
- ✓ Créer le fils ***FilsB,***
- ✓ Attend la fin de ses fils avant de se terminer,

Dans le cas où la seconde création échoue, il faut terminer le processus ***FilsA*** en lui envoyant le SIGKILL (***kill(pid_a, SIGKILL)***).

Dans le fils ***FilsA*** :

- ✓ Lit des caractères au clavier et les envois au ***FilsB,***
- ✓ Attend que le ***filsB*** lui renvoi le nombre de caractères alphabétiques qu'il a reçus,

Dans le fils ***FilsB*** :

- ✓ Lit les caractères envoyés par le fils ***FilsA***,

- ✓ Les affiches à l'écran,
- ✓ Compte le nombre de caractères alphabétique,
- ✓ Renvoi ce nombre au ***FilsA,***

On veillera à :

- ➢ Synchroniser le père et ses fils,
- ➢ A fermer tout ce qui doit l'être au bon moment et au bon endroit,

---

Exercice4 : **Tubes nommés.**

Le but de cet exercice est de simuler le fonctionnement d'un *Client-Serveur.* Dans ce type d'application, le serveur attend une requête d'un client dans un tube. Il traite la requête et répond dans un autre tube où le client peut donc lire cette réponse. Normalement, il y a un serveur et plusieurs clients. Pour qu'un client puisse lire sa réponse et non celle d'un autre client, le client indique par exemple, son *pid* dans sa requête puis s'endort. Le serveur écrit la réponse dans le tube puis réveille le client en lui envoyant un signal.

Dans cet exercice, on ne va pas gérer les signaux et on va avoir un seul client.

- ➢ Programme ***Serveur*** :

Le programme ***Serveur*** devra :

- ✓ Créer deux tubes nommés,
- ✓ Se mettre à l'écoute d'un tube,
- ✓ Lire la requête du client,
- ✓ Renvoyer un message à ce client via l'autre tube,
- ✓ Fermer et détruire les tubes.

- ➢ Programme ***Client*** :

Le programme ***Client*** devra :

- ✓ Envoyer un message au serveur,
- ✓ Lire la réponse,

Le message que le client envoie au serveur sera passé en ligne de commande, il faudra penser à vérifier que ce paramètre est bien passé.

Exercice 5 : tubes et recouvrement (pour les plus rapides)

Le but de cet exercice est de créer un programme ***Pere*** permettant à un père d'échanger avec un fils. Le père va envoyer une chaine de caractères au fils. Le fils est un programme exécutable Fils qui lit la chaine, la transforme en majuscules puis la renvoie au père.

Le fils étant un programme exécutable, il faudra utiliser le recouvrement. Lors du recouvrement, le fils perd les variables que lui transmet le père. Il faudra donc passer les descripteurs de tubes comme paramètres au programme Fils1. Les paramètres passés à un programme sont des chaines de caractères et les descripteurs des entiers, il faudra donc faire attention. Les deux processus ferment les descripteurs dès qu'ils n'en ont plus besoin.

➢ Ecrire le code du programme Fils.

➢ Ecrire le programme Pere.

➢ Vérifier le bon fonctionnement de l'ensemble.