Système TD 4

EXERCICE 1 : signal

Écrire un programme C qui attrape tous les signaux. Dès qu'il attrape un signal, il affiche un message dans le terminal avec le numéro du signal.

Utiliser la commande ***kill*** dans le terminal : *kill -numSignal numPid*

EXERCICE 2 : kill

Écrire un programme C qui demande un **PID** et envoie tous les signaux possibles au premier programme.

EXERCICE 3 : génocide des fils

Écrire un programme C qui génère *n* fils (*n* est passé en argument au programme : argv).
Chaque fils écrit son PID ainsi que son numéro de création toutes les secondes.
Le père tue tous ses fils après 5 secondes (signal 9).

EXERCICE 5 : discussion entre père et fils (tube)

Écrire un programme C qui crée un processus. Le père demande un texte dans le terminal, l'envoie au fils et le fils l'affiche dans le terminal (en précisant qu'il est le fils).
Voir le rappel.

EXERCICE 6 : discussion en entre père et fils avec réponse (double tubes)

Écrire un programme C qui crée un processus. Le père demande un texte dans le terminal, l'envoie au fils et le fils affiche qu'il l'a bien reçu, puis le renvoie au père qui l'affichera.

EXERCICE 7 : discussion entre frère

Écrire un programme C qui crée un processus. Le premier demande un texte dans le terminal, l'envoie au second fils. Le second fils affiche qu'il l'a bien reçu, puis le renvoie au premier fils qui l'affichera.

EXERCICE 8 : discussion entre père et fils

Écrire un programme C qui crée *n* fils. Le père demande un texte dans le terminal et l'envoie à tous ses fils, puis chaque fils affiche le message en indiquant qui il est.

EXERCICE 9 : tube nommé

Faire deux programmes C.
Le premier demande un message sur le terminal et écrit le message dans le tube.
Le second attend le message du tube pour l'afficher. (voir rappel)

Les deux programmes doivent être utilisés dans deux terminaux différents.

EXERCICE 10 : discussion avec tube nommé

Les deux programmes précédents (exercice 9) doivent poursuivre la discussion.
C'est-à-dire que le second programme demande un message, à son tour, sur le terminal et l'envoie au premier. Et ainsi de suite…

EXERCICE 4 : ping pong

Donner un programme C qui crée 2 fils.
Quand le premier fils reçoit le signal 3 alors il envoie le signal 4 au second fils (en affichant ***ping***).
Le second fils envoie le signal 3 quand il reçoit le signal 4 (en affichant ***pong***).
Le père commence la partie de ping pong en envoyant le signal 3 au premier fils.
Le père arrête la partie au bout de 10 secondes.

RAPPELS

Création et utilisation d’un tube anonyme

```c
int tube[2]; // déclare le tube
pipe(tube); // crée le tube anonyme
read(tube[0], tampon, 10); // lire 10 caractères dans le tube
write(tube[1], tampon, 10); // écrire 10 caractères dans le tube
close(tube[0]); // ferme le tube côté lecture
close(tube[1]); // ferme le tube côté écriture
```

Création d’un tube nommé

```c
// création du fichier pour l'utiliser en TUBE
if (mkfifo("fichier.tube", 0644) != 0) {
    //Si création du tube échouée
    perror("mkfifo");
    exit(1);
}
```