TD Programmation Système
Série 2 : Signaux

Exercice 1

Ecrire le programme C qui permet d'ignorer le signal SIGINT (ctrl^C) pour un certain nombre de seconde passé en paramètre.

Appels Système : signal(), kill(), sleep()

Indication
-Redéfinir le signal SIGINT en utilisant SIG_IGN
-Mettre le processus en attente par la fonction sleep()
-Revenir au comportement initial de SIGINT en utilisant SIG_DFL

Exercice 2

1. Ecrire le programme C "Boucle.c" qui permet de redéfinir le comportement d'un signal passé en paramètre. Le nouveau comportement consiste à afficher le numéro du signal. Le programme affiche son pid et rentre dans une boucle infinie.
2. Ecrire le programme C "Envoi.c" qui permet d'envoyer un signal à un processus. Le numéro du signal et le pid du processus sont passés en paramètres.

Appels Système : kill(), signal(), getpid()

Indication
-Exécuter le programme "Boucle" dans un terminal en lui passant un numéro du signal en paramètre
-Récupérer le pid du programme "Boucle"
-Lancer un deuxième terminal
-Exécuter le programme "Envoi" dans le nouveau terminal en lui passant en paramètre le pid de "Boucle" et le même signal redéfini par "Boucle"

Exercice 3 : Synchronisation Père/Fils

Ecrire le programme C sous Unix qui permet à un processus père de créer un processus Fils. Le père et le Fils doivent s'exécuter en parallèle en cordonnant leurs exécutions et en permettent l'affichage suivant :

```
Fils: 2 4 6 8 10
Père: 3 6 9 12 15
Fils: 12 14 16 18 20
Père: 18 21 24 27 30
........ ...........
........ ...........

Père: ..............99
```

Appels Systèmes : fork(), signal(), Kill(), pause()

Indication : Chaque fois qu'un processus affiche un multiple de 5 il se bloque et envoi un signal à l'autre processus pour le réveiller