TP UNIX n°5

Fonctions utiles : signal, kill, alarm, sleep, pause

1°) Ecrire un programme qui fait une boucle infinie qui affiche un '.' à chaque tour de boucle.

a) Exécuter ce programme et vérifier qu'il peut être arrêté en tapant **Ctrl C**.
Après avoir arrêté le programme par Ctrl C taper la commande 'ps' et vérifier que le processus est bien terminé.

b) Exécuter à nouveau ce programme et vérifier qu'il peut être arrêté en tapant **Ctrl **.
Après avoir arrêté le programme par Ctrl \ taper la commande 'ps' et vérifier que le processus est bien terminé.

c) Exécuter encore ce programme et vérifier qu'il peut aussi être arrêté en tapant **Ctrl Z**.
Après avoir arrêté le programme par Ctrl Z taper la commande 'ps', le programme est toujours là ! Tuez le par la commande : kill –s SIGKILL suivie de son numéro
et vérifiez qu'il est bien terminé cette fois.

2°) Modifier votre programme de façon à ce qu'il ignore le signal SIGINT.

Exécuter ce programme et vérifier qu'il ne peut plus être arrêté en tapant Ctrl C mais seulement en tapant Ctrl \ ou en tapant Ctrl Z puis en le tuant par la commande :
kill –s SIGKILL suivie de son numéro.

3°) Ecrire un nouveau programme qui récupère le signal SIGTSTP par une fonction qui affiche le message "Loupé !!" puis fait une boucle infinie sans affichages (boucle vide).

Exécuter ce programme et vérifier lorsque l'on tape Ctrl Z pour la première fois on voit bien le message "loupé !!". Que se passe-t-il lorsque l'on tape plusieurs fois Ctrl Z ?
Vous arrêterez votre programme par Ctrl C.

4°) Modifier le programme du 3°) de façon à ce que lorsque l'on tape Ctrl Z pour la première fois on voit le message "loupé !!" mais lorsque on le tape la deuxième fois, le programme s'arrête normalement. Il faudra ensuite le tuer par la commande :
kill –s SIGKILL suivie de son numéro.

5°) Ecrire un programme qui crée un processus fils par fork.
- Le processus père attend 5 secondes, envoie le signal SIGUSR1 à son fils puis attend la terminaison du fils et se termine lui-même.
- Le processus fils initialise une variable globale 'attente' à 0 puis exécute une boucle while vide tant que cette variable est à 0. Lorsqu'il sort de cette boucle il se termine normalement. Le processus fils doit récupérer le signal SIGUSR1 par une fonction qui met la variable globale 'attente' à 1. Ceci le fera sortir de la boucle et se terminer.

Placer des traces dans les 2 processus pour voir si tout fonctionne bien, en particulier au moment où le père émet le signal et au moment où il se termine ainsi que lorsque le fils entre dans sa boucle d'attente, lorsqu'il reçoit le signal du père puis lorsqu'il se termine.