I-C1 TD n°1

EXERCICE 1 - étude des processus UNIX

1- A l'aide de la commande ps, afficher la liste de tous les processus tournant sur votre machine, avec les informations suivantes :

USER nom de l'utilisateur propriétaire du processus
PID numéro d'identification
%CPU
%MEM
STAT Etat
START Date de début
TIME
COMMAND Commande utilisée pour lancer ce processus

(vous vous aiderez du manuel (man ps) et du résumé (ps --help).

1. A quoi correspond l'information TIME ?
2. Quel est le processus ayant le plus utilisé le processeur sur votre machine ?
3. Quel a été le premier processus lancé après le démarrage du système ?
4. A quelle heure votre machine a-t-elle démarré ?
5. Pouvez-vous établir le nombre approximatif de processus créés depuis le démarrage ("boot") de votre machine ?

2- Sous UNIX, chaque processus (excepté le premier) est créé par un autre processus, son processus père. Le processus père d'un processus est identifié par son PPID (Parent PID).
– Trouver une option de la commande ps permettant d'afficher le PPID d'un processus.
– Donner la liste ordonnée de tous les processus ancêtres de la commande ps en cours d'exécution.

3- Reprendre la question précédente avec la commande pstree.

4- Essayez la commande top, qui affiche les mêmes informations que ps mais en rafraichissant périodiquement l'affichage.
1. La touche ? permet d'afficher un résumé de l'aide de top. Afficher dans top la liste de processus triée par occupation mémoire ("resident memory") décroissante.
2. Quel est le plus "gros" processus sur votre machine ? A quoi correspond-il ? (rappel : vous pouvez utiliser man truc pour découvrir ce que fait truc...).

EXERCICE 2 - Arrêt d'un processus.

1- Créer dans un fichier compteur.c un programme suivant l'algorithme :

- i <- 0
- Répéter infiniment :
  - i <- i + 1
  - si i est multiple de 100 000, afficher i et un saut de ligne (\n).

2- Lancer l'exécution de ce programme et vérifier qu'il fonctionne. L'arrêter en tapant CTRL-C.
3- En utilisant les fonctionnalités du shell (&, fg, bg), lancer quatre instances du programme compteur en même temps. Mettre au premier plan la troisième, l'arrêter (CTRL-Z) puis la relancer en arrière plan.
4- A l'aide des commandes jobs et kill %n, arrêter tous les compteurs.
5- Même question en utilisant les commandes ps et kill (avec un PID).

EXERCICE 3 - Gestion des signaux dans des programmes

Le comportement d'un programme lorsqu'il re¸coit un signal dépend du type de signal. Certains signaux peuvent être "déroutés", c'est `a dire que le programme peut spécifier à l'avance que le signal doit être ignoré ou traité par une fonction spéciale (signal handler) qui sera appelée lors de la réception du signal.
Vous traiterez les questions suivantes en langage C (voir le support de cours UNIX plus d'informations).
1- écrire un programme qui affiche indéfiniment "top" toutes les secondes. On utilisera l'appel système sleep pour bloquer le processus une seconde.
Lancer le programme, le stopper avec CTRL-Z, le relancer, puis le tuer avec CTRL-C.
2- Modifier le programme pour qu'il ignore le CTRL-Z. (man 7 signal donne la liste des signaux).
3- Modifier le programme pour qu'il affiche "signal X recu" lorsqu'il re¸coit le signal numéro X. Quel signal doit-on lui envoyer pour le tuer ?

EXERCICE 4 - Révision sur les tubes
1- Quelle est la différence entre tee et cat ?
2- Que font les commandes suivantes :

```
$ ls | cat
$ ls -l | cat > liste
$ ls -l | tee liste
$ ls -l | tee liste | wc -l
```