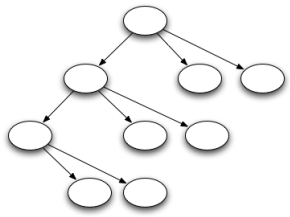
Examen «Systèmes» - M1

Exercice 1

Écrire un programme C dont l'exécution permettra de créer un ensemble de processus correspondant à la généalogie suivante :



Exercice 2

Écrire un programme C qui permettra lors de son exécution :

1. de saisir une suite de caractères au clavier
2. de créer un processus fils
3. de charger le processus fils d'exécuter la commande dont le nom correspondra à la suite de caractères précédemment tapés
4. de terminer le processus père de sorte qu'il renvoie la valeur de retour du processus fils

Exercice 3

Écrire un programme C, qui lors de son exécution devra :

1. créer une arborescence de processus afin de réaliser un *job* de type `cmd1 | cmd2 | cmd3` (trois processus communiquant à travers des tubes)
2. faire en sorte que la réception par le père de l'un des signaux `SIGHUP` ou `SIGTERM` provoque la terminaison immédiate et certaine de tous les processus créés
3. faire en sorte que la terminaison ordinaire du *job* provoque la terminaison du père et qu'il renvoie la valeur de retour du processus correspondant à `cmd3`

Exercice 4

On souhaite créer deux processus capables de s'échanger des données à travers une mémoire partagée. L'un d'entre ne fera qu'y écrire, l'autre seulement lire; mais on souhaite assurer les propriétés suivantes:

1. toute lecture devra être précédée d'une écriture
2. toute écriture (sauf la première) devra être précédée d'une lecture

Q1. décrire les mécanismes que vous allez utiliser afin de garantir les propriétés ci-dessus
Q2. écrire le(s) programme(s) implémentant les mécanismes décrits à la question précédente

Problème

On se propose de réaliser un crible parallèle de nombres premiers. La méthode de base sera celle d'Ératosthène et qui date du 3° siècle avant J.C et fonctionne ainsi :

1. écrire les nombre de 2 à n, les uns à la suite des autres
2. à partir de la gauche prendre le premier nombre non barré (disons k)
3. noter ce nombre qui est un nombre premier
4. barrer tous les nombres (y compris celui-ci) en sautant de k cases à chaque fois
5. recommencer en 2.

Ici, il s'agit donc de paralléliser cet algorithme de sorte que les opérations 4. et 5. représentent le début d'une exécution parallèle et non une simple séquence. Les nombres devront impérativement être placés dans une mémoire accessible à toutes les entités de calcul. On devra assurer une synchronisation correcte des entités de calcul (processus ou threads) de sorte que le lancement d'une nouvelle entité de calcul garantisse bien que le nombre premier suivant sera trouvé effectivement.

1. décrire avec soin les problèmes à résoudre
2. écrire le programme correspondant