Université Côte d'Azur
Faculté des Sciences - Département d'Informatique

Licence d'Informatique L2
Introduction aux Systèmes et Réseaux

TP n ° 1 : Introduction aux processus sous UNIX

1 Identification des processus

1. Testez les commandes `top(1)` et `ps(1)` pour afficher les processus s'exécutant sur la machine que vous utilisez.
2. Écrire un programme python3 qui affiche le numéro (`PID`) du processus qui l'exécute. Que remarquez-vous en le lançant plusieurs fois ?
3. Modifiez le programme pour qu'il affiche son `PID` et son `PPID`. Que remarquez-vous ?

2 Environnement des processus

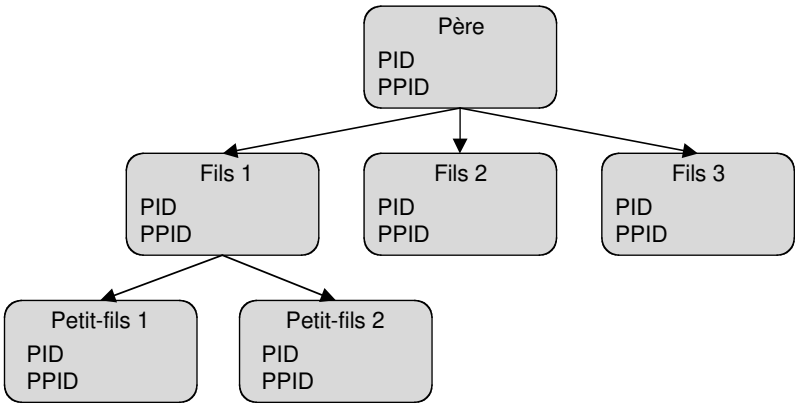
1. Testez la commande `env` pour afficher les valeurs des variables d'environnement courantes.
2. Examinez en particulier la valeur de la variable d'environnement `PATH`. Cette variable permet de localiser automatiquement un fichier exécutable (binaire) en définissant une suite de répertoires dans lesquels on le recherche, dans l'ordre indiqué. C'est ce qui permet, par exemple, à l'utilisateur de taper la commande `python3` au lieu (par exemple) de `/usr/bin/python3`. Pour connaître la localisation exacte d'une commande, on utilise la commande `which` (essayez par exemple `which ls` et `which python3`).
3. Exécutez la commande `export SAVEPATH=$PATH` (qui recopie dans la nouvelle variable `SAVEPATH` la valeur courante de `PATH`). Puis exécutez `export PATH=/:.` et tapez la commande `ls`. Que constatez-vous et comment l'expliquez-vous ?

   Pour restaurer la valeur de votre variable `PATH`, exécutez la commande `export PATH=$SAVEPATH`. Puis exécutez `unset SAVEPATH` pour éliminer la variable `SAVEPATH` de votre environnement. Question : comment se fait-il que la commande `export` continue de fonctionner malgré la modification du `PATH` ?
4. Depuis python, il est possible d'accèder directement aux variables de l'environnement. Celles-ci sont stockées dans le dictionnaire `os.environ`. Écrivez un programme qui affiche la valeur de chaque variable d'environnement.

   Le nombre de variables d'environnements listées par la commande `env` ou `printenv` est-il le même que celui contenu dans `os.environ` ? Utilisez `wc`.

3 Création de processus

1. Écrire un programme qui reproduit l'arbre généalogique représenté ci-après. Chaque processus doit afficher son `PID`, son `PPID`, et afficher les fils qu'il engendre.



2. Observez l'ordre d'apparition des messages à l'écran et commentez. Que se passe-t-il si on lance plusieurs fois le programme ?
3. Écrire les programmes (vus en TD, question 4) qui réalisent respectivement une chaîne de `n` processus, et un arbre de `n` processus (`n` est passé en paramètre à l'exécution du programme).

4 Synchronisation élémentaire de processus

1. Modifier le programme de la question précédente (3.1) pour que le fils 2 affiche son message avant les fils 1 et 3.
2. Modifier le programme pour que les petits-fils 1 et 2 affichent leur message avant les fils 2 et 3.
3. Reprendre et exécuter le programme vu à la question 1 du TD2.

5 Exécution de processus

1. (cf question 1 du TD3) : écrire un programme qui lance le programme de la question 1.3 (afficher le `PID` et le `PPID`), en modifiant certaines variables d'environnement. Faire afficher ces variables par le programme.
2. Reprendre la question 1.2 du TD3 : écrire un programme qui exécute une commande Unix qu'on lui passe en paramètre.

6 Travail maison

6.1 Fork

Implémentez les exercices 5, 6 et 7 du TD1.

6.2 Wait

Implémentez l'exercice 2 du TD2.

6.3 Exec

Implémentez les exercices 2, 3, 4, 5 du TD3.