Programmation C
TP 1

L'objectif de cette séance est d'aborder les principales commandes LINUX.

Exercice O Préliminaires

1. Démarrer sous Linux et ouvrir un terminal. Le TP s'effectuera dans un terminal Linux
2. Afficher avec la commande `pwd` le chemin du répertoire actuel
3. Afficher le contenu de ce dossier avec `ls`, voire `ls -l`
4. Se déplacer dans l'un de ses sous-répertoires, puis revenir. S'exercer à se déplacer en utilisant les chemins relatifs aussi bien qu'absolues

Exercice 1 Un peu de compréhension

1. Expliquer ce que fait la suite de commandes suivantes (à tester) :

```
cd
mkdir ./tutu
mkdir tutu/titi
cd tutu
chmod u=r ../tutu/titi
mkdir ~/tutu/titi/toto
```

Exercice 2 Arborescence et droits

1. Donner la suite de commandes permettant de générer l'arborescence suivante (utiliser `touch` pour créer un fichier vide) :

```
+tp1/
        + dir1/
                -- f11
                -- f12
                -- titi
        + dir2/
                -- f2
```

2. Mettre les droits suivants :
   - tp1 : tous les droits pour tout le monde
   - dir1 : tous les droits pour tout le propriétaire et le groupe
   - dir2 : accès restreint au propriétaire

- f11 : exécutable par tous
- f12 : lisible par tous, modifiable par le propriétaire
- f2,titi : lecture et écriture pour le propriétaire

3. Donner la suite de commandes correspondant aux actions suivantes :
   - aller dans dir1
   - sans changer de répertoire, copier f11 vers tp1
   - renommer cette copie en tutu
   - aller dans dir2
   - déplacer ici tous les fichiers de dir1 commençant par 'f'

Exercice 3 Redirection des sorties, fichiers

1. Avec la commande `echo` (qu'est-ce que fait `echo` ?) et redirections, créer un fichier nommé `titi` contenant `Bonjour!`
2. En s'aidant de la commande `cat`, créer un fichier `toto` contenant deux fois le contenu de `titi` (il y a au moins deux façons de faire)
3. Que fait la commande `cat z » z` ?
4. Lister l'ensemble des fichiers contenus dans l'arborescence `/var/spool/` en stockant les messages d'erreurs produits dans un fichier nommé `err`. Afficher les 10 dernières lignes de `err`
5. Donner des commandes permettant de connaître la taille en octets de la page `man` de `mv` (indice: `ls` donne cette taille)

Exercice 4 Gestion des processus

1. Lancer le programme `xeyes`. Tenter de lancer une autre commande : le programme en cours l'empêche. Tuer-le donc (ctrl-c). Vérifier son arrêt (`jobs`, `ps`)
2. Lancer `xeyes` de nouveau, le seulement stopper cette fois. Qu'affichent `jobs`, `ps` ? Comment réactiver le processus ? Le faire et le tuer.
3. Lancer en simultané plusieurs instances de `xeyes`. Les yeux peuvent être déplacés par souris, en appuyant Alt-gauche. Regarder `jobs` et `ps`. Tuer une des instances de xeyes avec `kill`.
4. Ouvrir un nouveau fenêtre de terminal. Y lancer `jobs` et `ps`. Comment voir les processus ouverts depuis un autre terminal ? Tuer un des processus xeyes depuis ce nouveau fenetre.

Exercice 5 Scripts (petit bonus)

Le but de savoir contrôler un ordinateur par commandes texte n'est pas de faire snob. Cela permet de mieux comprendre comment le système d'exploitation fonctionne, mais il y a surtout d'autres avantages. Par exemple, il est beaucoup plus facile de travailler avec une machine à distance quand on ne dépend pas d'éléments graphiques. Mais notamment, le texte est une forme bien adapté pour automatiser des tâches. Parlerons des scripts shell.

Dans sa forme simple, un script peut être vu comme un fichier texte contenant une suite de commandes. Quand il est lancé, ces commandes sont exécutés. Cela permet d'effectuer de tâches potentiellement complexes par avec seule commande.

Pour créer un tel script, il suffit de créer un fichier texte, dont la première ligne sera `#!/bin/bash` et le reste constitueront les commandes désirés.

1. Créer un script control.sh qui, quand on le lance, affiche "bonjour" et liste les fichiers et répertoires dans votre répertoire courant
2. Executer ./control.sh. Que se passe-t-il ? Corriger.

Les scripts peuvent au fait être plus puissants que des 'simples' listes de commandes, mais nous n'allons pas le traiter dans ce cours.

Remarque finale

Il ne faut pas se laisser décourager si les commandes semblent difficiles à maîtriser au début. Il n'y a pas de honte de profiter d'une interface graphique. Avec de l'expérience, on remarquera quelles tâches sont plus faciles à accomplir en mode graphique et quelles en mode texte.

Il ne faut pas hésiter à chercher de l'aide dans les pages de manuel aussi bien qu'en ligne. Au début, les informations trouvées peuvent sembler difficiles à comprendre. Il faut persister, c'est une question d'expérience.

TP 1bis – exercices avancées

Seulement si vous êtes vites et à l'aise. N'est pas exigé.

Exercice 6 Ca va couper chérie ...

Preliminaires :
Avec des "pipes" (symbole |), il est possible de rediriger le sortie d'une commande vers l'entrée d'une autre, sans devoir passer par un fichier temporaire. Par example, la suite d'instructions suivante :

```
ls > tmp
wc < tmp
```

peut être remplacé par l'instruction suivante :

```
ls | wc
```

1. En utilisant la page man de la commande `cut` et en vous rappelant que la sortie de `ls -l` donne quelque chose comme:
   `-rw-r-r- 1 maaouni thesards 289 sep 11 02:48 test.c`
   donnez les commandes utilisant des pipes permettant d'afficher les choses suivantes:
   - juste la liste des droits de ce qu'il y dans le répertoire
   - les droits, propriétaires et noms des fichiers avec l'extension `.c`.
   - uniquement les droits en lecture des fichiers

   Attention le formatage du `ls -l` dépend de la configuration de la machine que vous utilisez. Votre solution peut ne pas correspondre avec celle de l'un de vous collèques. Pour filtrer certaines lignes inutiles, vous pouvez utiliser la commande `grep`.
2. Sachant que les lignes du fichier **/etc/passwd** sont de la forme :
   `talon:x:635:201:Marie-Pierre Talon:/home/compta/talon:/bin/bash`
   donnez une commande permettant de sauver la liste triée (commande `sort`) des noms et prénoms dans un fichier `tutu`
3. En utilisant `grep`, donnez une commande permettant d'afficher les logins des gens prénommés `Pierre`

Exercice 7 Le tube de la rentrée

1. Lancer en simultané plusieurs instances du programme `xeyes`
2. En utilisant des `pipes`, n'afficher que les numéros de processus correspondant aux `xeyes`
3. Ajouter un pipe avec la commande `kill` pour tuer ces processus. Que se passe-t-il ?
4. Lorsqu'on met une commande entre quotes inversées (`‘toto‘`) cette expression est équivalente au résultat affiché par la commande `toto` et va à son tour être évaluée. Exemple:

```
$>echo find me
find me
$>‘echo find me‘
find: me: No such file or directory
```

En utilisant des quotes inversées, modifier la commande précédente pour obtenir une ligne de commande qui tue tous les processus `xeyes`