De la puce au web - LI105
TD-TME 3
Système

L'objectif de la séance de TME est de manipuler la distribution Debian du système d'exploitation Linux en se familiarisant avec la ligne de commande.

Pour cela, ouvrez un terminal depuis votre environnement préféré. Si vous travaillez sous l'environnement Gnome, utilisez l'application `Terminal` accessible depuis le menu Applications → Accessoires. Similairement, si vous êtes sous l'environnement Kde utilisez l'application `Émulateur de terminal` (ou `Konsole`) accessible depuis le menu Applications → Utilitaires.

1 Travail en ligne de commandes

1.1 Commandes de base dans le terminal

Exercice 1

Question 1 Tapez la commande `ls` suivi de la touche `<Entrée>`.

Solution : `ls` liste le contenu de votre répertoire personnel

Question 2 Entraînez-vous à copier-coller du texte dans la fenêtre bash :

1. Boutons copier / coller dans le menu édition
2. Sélection du texte à la souris (mise en surbrillance) pour copier et clic avec le bouton central de la souris (molette) pour coller
3. Tapez la commande `ls -l` suivi d'un nom de fichier copié à l'aide de la commande `ls` précédente, suivi de la touche `<Entrée>`
4. Tapez la commande `pwd` pour déterminer le répertoire dans lequel vous vous trouvez
5. Tapez la commande `cd /` puis `ls` et enfin `cd ~` puis `pwd`. Commentez ce que vous observez.

Solution :

1. Pas de réponse : il faut s'entraîner !
2. Idem
3. `ls -l` affiche le contenu du répertoire au format long (avec les détails sur les droits d'accès)
4. `pwd` → chemin absolu de votre répertoire personnel
5. `cd /` → aller à la racine, `ls` → contenu de la racine, `cd ~` → retour au dossier utilisateur, `pwd` → de nouveau affichage du chemin absolu jusqu'au répertoire utilisateur.

1.2 Parcours de l'arborescence des fichiers en ligne de commandes

Exercice 2

Question 1 Taper la commande suivante `nautilus &` (sous Gnome) ou `dolphin &` (sous Kde) dans le terminal et valider. Qu'observez-vous ?

Solution : Le navigateur de fichiers graphique est accessible depuis la ligne de commande. Il y a une interaction forte entre ces deux modes de fonctionnement.

Question 2 Taper la commande suivante `gnome-search-tool &` (sous Gnome) ou `kfind &` (sous Kde) dans le terminal et valider. Qu'observez-vous ?

Solution : L'utilitaire de recherche avancée de fichiers est également accessible depuis la ligne de commande !

Question 3 Saisir les commandes suivantes en alternance avec la commande `pwd` (pour déterminer dans quel répertoire vous êtes) :

1. `ls *`, `ls .`, `ls ..`, `ls ../..`, `ls ~` et `ls ~login` (où `login` représente un numéro d'étudiant valide), `ls /usr`, `ls /usr/*`, `ls /u*r/*`, `ls /u*/*`, `ls /usr/X*/b*/x[c-f]*`,
2. `cd`, `cd .`, `cd ..`, `cd ../..`, `cd /`, `cd ~` et enfin `cd ~login` (où `login` représente un numéro d'étudiant valide)

Solution : Il faut tester pour voir le résultat !

1.3 Gestion des droits en ligne de commande

Exercice 3

Question 1 Aller dans votre répertoire principal et créer un nouveau répertoire nommé `Essai2`.

Solution :
```
cd
mkdir Essai2
```

Question 2 Comme précédemment interdire l'accès aux membres de votre groupe ainsi qu'aux autre utilisateurs pour le répertoire `Essai2`.

Solution : `chmod go-rxw Essai2`

Question 3 Modifier les droits d'accès du répertoire `Essai2` pour que personne ne puisse y accéder (en lecture, en écriture et en exécution).

Solution : `chmod a= Essai2` ou bien en partant de la situation précédente `chmod u-rwx Essai2`

1.4 Complétion automatique des commandes

Introduction BASH (l'interprète de commandes intégré aux applications `Terminal` et `Konsole`) aide à compléter les commandes, le chemin ou le nom d'un répertoire ou fichier que vous êtes en train de taper. Pour cela, il suffit de commencer à taper le début de votre commande, chemin ou nom et d'appuyer ensuite sur la touche `<TAB>` :

- s'il n'y a aucune ambiguïté, `bash` trouvera la solution et complètera votre commande, chemin ou nom.
- Dans le cas contraire, il y a zéro ou plusieurs solutions possibles. Appuyer une deuxième fois sur la touche `<TAB>`. S'il y a plusieurs solutions, `bash` vous en proposera ou bien affichera le nombre de possibilités si celui-ci est grand. Sinon, `bash` ne fait rien.

Exercice 4

Dans les questions suivantes, vous observerez le comportement de complétion de commande de `bash` en appuyant plusieurs fois sur `<TAB>` si nécessaire et indiquerez s'il y a 0, 1 ou plusieurs commandes possibles. Comment décider du nombre de commandes `/` chemins existants ?

Question 1 Pour les commandes suivantes :

1. "l"`<TAB>` ("l" comme "L"ucie en minuscule) puis "s" `<TAB>`
2. "x" `<TAB>` puis "c" `<TAB>` puis "l" `<TAB>` puis "o" `<TAB>`

Solution :

Quand on appuie sur `<TAB>` une fois : si la commande est complétée, il n'y avait qu'une seule solution possible ; s'il ne se passe rien il y a 2 cas : soit il n'y a pas de solution, soit il y en a plusieurs : il faut de nouveau appuyer sur `<TAB>` pour forcer l'affichage d'une invite pour toutes les solutions.

Pour les commandes :

1. “l”<TAB> : rien : 0 solution ou plusieurs solutions : <TAB> trop de solutions ; “y” pour les voir, “q” pour quitter. Si on tape “s” puis <TAB> : commande “ls”
2. “x” <TAB> : rien : 0 ou plusieurs solutions : puis “c” <TAB> : idem si de nouveau <TAB> affichage de toutes les solutions pas trop nombreuses : puis “l” <TAB> : idem puis “o” <TAB> : proposition de “xclock”

Question 2 Pour les noms ou les chemins :

1. “ls /us” <TAB>, puis “b” <TAB>, . . . , “konquer” <TAB>
2. “ls /us” <TAB>, puis “loc” <TAB>, . . . , “f” <TAB>, puis “f” <TAB>
3. “ls /usr/X*” <TAB>
4. “ls /usr/X*R6” <TAB>
5. “ls /usr/X*R*” <TAB>
6. “ls /usr/X ? ?R*” <TAB>
7. comparer “ls /usr/X” <TAB> puis “/b” <TAB> <TAB> “[d-s]*” et “ls /usr/X*/b*/mkfont[d-s]*”
8. “ls /usr/X” <TAB> puis “/b” <TAB> <TAB> “[a-cs-z]*”
9. “ls /usr/X*R” <TAB> . . . “6” <TAB>

Solution : Pour les noms ou chemins :

1. “ls /us” <TAB> : 1 suite possible `ls /usr/` puis “b” <TAB>, `ls /usr/bin` <TAB> : trop de solutions : puis “konquer” <TAB> : propose de lancer konqueror.
2. idem précédemment mais permet de lancer firefox
3. “ls /usr/X*” <TAB> : 1 solution `ls /usr/X11R6/`
4. “ls /usr/X*R6” <TAB> : 1 solution `ls /usr/X11R6/`
5. “ls /usr/X*R*” <TAB> : 1 solution `ls /usr/X11R6/`
6. “ls /usr/X ? ?R*” <TAB> : 1 solution `ls /usr/X11R6/`
7. comparer “ls /usr/X” <TAB> puis “/b” <TAB> <TAB> “[d-s]*” et “ls /usr/X*/b*/mkfont[d-s]*” : à tester !
8. “ls /usr/X” <TAB> puis “/b” <TAB> <TAB> “[a-cs-z]*”
9. “ls /usr/X*R” <TAB> . . . <TAB> plusieurs fois : rien et pourtant si on tape “6” <TAB> on obtient une solution : `ls /usr/X11R6/` : fonctionnalité par encore implémentée dans `bash`.

2 Utilisation de commandes avancées

2.1 Commande `cut`

Exercice 5

Soit le fichier `proverbe.txt` contenant le texte suivant :

Rien ne sert de courir
Il faut partir à point.
La Fontaine.

Question 1 créer le fichier `proverbe.txt` dans votre répertoire personnel.

Solution :

```
cd
cat > proverbe.txt
Rien ne sert de courir
Il faut partir à point.
La Fontaine.
^D
```

Question 2 donner la commande pour n'afficher que les 5 premiers caractères de chaque ligne

Solution : La commande `cut` permet d'afficher une partie de chaque ligne d'un fichier.

```
cut -c 1-5 proverbe.txt
```

On choisit sur chaque ligne les caractères (option -c) compris entre le caractère de la position 1 jusqu'au caractère de la position 5 (1-5 comme valeur choisie pour l'option -c).

Question 3 donner la commande pour n'afficher que les 2 premiers mots de chaque ligne

Solution : On choisit comme délimiteur de champ (`-d` pour delimitor en anglais) le caractère `<espace>`. On entoure ce dernier de guillemets simples (quote ' '). En effet, sans être entouré de guillemets, le caractère `<espace>` sera pris comme séparateur d'option pour la commande `cut`. On dit que l'on déspécifie le caractère `<espace>`. On peut aussi utiliser le caractère \ pour déspécifier le caractère `<espace>`. Ce qui donne :

- `cut -d' ' -f 1-2 proverbe.txt`
- ou bien `cut -d\ -f 1-2 proverbe.txt`

Attention, il y a 2 caractères `<espace>` : le caractère \ est suivi d'un premier `<espace>` (déspécifié et utilisé donc comme délimiteur) puis d'un autre `<espace>` (séparateur de l'option suivante `-f`). On choisit ensuite tous les caractères compris entre le champ 1 (`-f` pour fields en anglais) jusqu'au champ 2 (1-2).

Question 4 quel sera le résultat de la commande : `cut -f 2 -de proverbe.txt` ?

Solution :

- `f 2` : 2ème champ de chaque ligne
- `de` : le caractère 'e' sert de délimiteur de champs

On affiche le deuxième champ de chaque ligne du fichier proverbe.txt soit :

- Ri/e/n n/e/ s/e/rt d/e/courir = {Ri} {n n} { s} {rt d} {courir} = {n n}
- Il faut partir à point. = pas de 'e' : toute la ligne est renvoyée = {Il faut partir à point.}
- La Fontaine. = {La Fontain} {.} = {.}

2.2 Commandes `tail` et `sort`

Soit le fichier `pantheon.csv` contenant le texte suivant :

```
Nom; Prénom; Naissance
Page; Jimmy; 1944
Van Halen; Eddie; 1955
Vaughan; Stevie Ray; 1954
Clapton; Eric; 1945
Lagrène; Birelli; 1966
Beck; Jeff; 1944
McLaughlin; John; 1942
Vai; Steve; 1960
Hendrix; Jimi; 1942
```

Exercice 6

Question 1 Donner la commande qui permet d'obtenir le fichier `liste.csv` qui est une copie de `pantheon.csv` sans la première ligne.

Solution : On utilise la commande `tail` qui donne les dernières lignes d'un fichier. Pour obtenir les lignes comptées depuis la fin du fichier la commande tail requiert l'indicateur - suivi du nombre de lignes : `tail -1 pantheon.csv` donne la dernière ligne du fichier. Pour obtenir les lignes comptées depuis le début on utilise l'indicateur + suivi du numéro. Ainsi, `tail +2 pantheon.csv` donne toutes les lignes depuis la 2ème jusqu'à la dernière.

Pour créer le fichier `liste.csv` on aura donc :

```
tail +2 pantheon.csv > liste.csv
```

La commande `tail +2 pantheon.csv` produit son résultat à l'écran. Pour mettre cela dans un fichier, on utilise la redirection `>` vers un fichier (ici `liste.csv`).

Question 2 Donner la commande pour obtenir un affichage de la liste des guitaristes par noms croissants.

Solution : `sort liste.csv`

Question 3 Donner la commande pour obtenir un affichage de la liste par dates décroissantes.

Solution : `sort -r -t“ -k 2 liste.csv`

- L'option `-r` (pour reverse en anglais) pour un ordre décroissant.
- L'option `-t` pour choisir le caractère `<espace>` (déspécifié par “) comme délimiteur de champ.
- L'option `-k` pour choisir le champ (ici 2) pour le tri.

Il est également possible ici de choisir le caractère ';' comme délimiteur : `sort -r -t';' -k 2 liste.csv`

3 Variables d'environnement

On poursuit dans cet exercice l'exercice portant sur les variables d'environnement du TD3. Si celui-ci n'a pas été abordé, commncez par le traiter avant de poursuivre sur les questions suivantes.

Exercice 7

Question 1 En utilisant la variable `HOME`, écrire une commande pour afficher la liste de tous les répertoires dont le père est commun avec votre répertoire personnel.

Solution : La variable `HOME` contient le chemin absolu de votre répertoire personnel. Le `..` indique le répertoire père. On écrit donc : `ls $HOME/..` ;

Pour vous, ce sont normalement les répertoires personnels des étudiants dont les identificateurs se terminent par le même chiffre que votre identificateur. En règle générale, le nom d'un répertoire personnel d'un utilisateur est le même que son identifiant.

Question 2 Modifier la commande précédente pour afficher la liste de tous les utilisateurs dont le nom comporte un ou plusieurs 3.

Solution : `ls $HOME/.. | grep 3` un ou plusieurs 3. Donc, un 3 suffit.

Question 3 Modifier la commande précédente pour afficher la liste de tous les id des utilisateurs dont le nom comporte un ou plusieurs 3 triée en ordre décroissant.

Solution : `ls $HOME/.. | grep 3 | sort -r`

4 Écriture de scripts simples

Exercice 8

Question 1 Rappeler ce qu'est un script.

Solution : C'est un ensemble de commandes à exécuter (voir cours). En général, ces commandes sont écrites dans un fichier que l'on exécute (il faut donc qu'il soit exécutable, cf droits d'accès).

Question 2 Ecrire un script qui ne prend pas d'arguments et qui renvoie le nombre de dossiers ou fichiers de votre répertoire personnel contenant le motif “jpeg”.

Solution :

```
cat > monScript
cd
ls *jpeg* | wc -l
^D
chmod a+x monScript
cat > monScript
cd
ls | grep jpeg | wc -l
^D
chmod a+x monScript
```

Question 3 Transformer le script précédent en un script qui accepte pour argument le chemin absolu d'un répertoire à analyser.

Solution :

```
cat > monScript
cd $1
ls *jpeg* | wc -l
^D
chmod a+x monScript
cat > monScript
cd $1
ls | grep jpeg | wc -l
^D
chmod a+x monScript
```

Question 4 Ecrire maintenant un script acceptant 2 arguments : un nom de fichier et un motif et qui renvoie le nombre de lignes de ce dernier dans lesquelles apparaît le motif.

Solution :

```
cat > monScript
grep $2 $1 | wc -l
^D
chmod a+x monScript
./monScript monFichier.txt ette
```

5 Expressions rationnelles

Nous étudions ici quelques expressions rationnelles (aussi appelées "régulières" du fait de leur nom en anglais `regular expression`).

Exercice 9

On supposera que le répertoire de travail est `/usr/include/`.

Question 1 Lire la page `man` de la commande `egrep` et indiquer son rôle.

Solution : Il faut regarder la page `man`

Question 2 Utiliser la commande `egrep` (syntaxe : `egrep -n <motif> <fichier>`) avec les motifs suivants, en terminant à chaque fois par la touche `<Entrée>` :

1. `ff` : avec comme fichiers `*.h`, puis `X.h`. Les noms de fichiers sont aussi affichés.
2. `[0-9]` : avec comme fichier `X.h`
3. `0x[a-f0-9A-F]+` : idem
4. `0x[a-f0-9A-F]` : ce motif est suffisant pour donner le même résultat que le motif précédent.
5. `0x[a-f0-9A-F]+` : avec comme fichier `*.h`

6. `0x[a-f0-9A-F]` : idem.

Solution : Il faut manipuler pour voir le résultat

6 Processus, commande `ps` et `kill`

Exercice 10

Question 1 Lancer le terminal (ci-après la fenêtre 1) (taper la commande `terminal &` ou `konsole &`) et taper la commande `ps -Al` pour afficher en format long tous les processus. Repérer le processus bash correspondant à cette instance du terminal en cours d'exécution et dessiner le sur un brouillon comme le nœud d'un arbre avec son `PID` et son `PPID`.

Question 2 À partir de la fenêtre 1, ouvrir un nouveau terminal comme précédemment pour créer une nouvelle fenêtre (ci-après la fenêtre 2). Taper la commande `ps -Al`. Repérer les nouveaux processus de terminal et bash engendrés et compléter l'arbre ci-dessus avec ces nouveaux nœuds.

Question 3 Dans la fenêtre 2, tuer le processus bash ou terminal de la même fenêtre avec la commande `kill -9 PID-du-processus`.

Question 4 Recommencer tout le processus de création des deux fenêtres et depuis la seconde arrêter le bash de la première. Que remarquez-vous ?

Solution : Si un processus fils demande l'arrêt de son père, cela va l'arrêter également.