Université Grenoble Alpes **Année 2025/2026**
L3 MIAGE

Introduction au Système
Planche de TP n°1

NB. En cas de problème pour effectuer les exercices de cette planche sur la machine de la salle de TP, connectez-vous au serveur mandelbrot : faire un ssh sur
im2ag-mandelbrot.univ-grenoble-alpes.fr
Si besoin, voir ici pour l'utilisation de ssh :
https://phoenixnap.com/kb/ssh-to-connect-to-remote-server-linux-or-windows

1. Configuration de votre environnement de travail

1.1. Lire les informations concernant ce qui se passe au login, et l'importance du fichier **.bashrc** ici : *http://teaching.idallen.com/cst8207/19w/notes/350_startup_files.html*
Voir particulièrement la section 3, dont :
`A BASH shell that is not interactive reads the .bashrc file when it starts up...`

En utilisant l'option *-a* de la commande *ls*, voyez-vous **dans votre répertoire de login** des scripts de configuration, notamment `.bashrc` ? Si oui, dans la suite vous l'éditerez, si non vous le créerez.

1.2. Il est important de savoir enrichir ses **fichiers de configuration**. Nous étudions d'abord la définition d'aliases.
Tout d'abord, **lire** les informations concernant la définition "d'**aliases**" pour vos commandes ici :
https://doc.ubuntu-fr.org/alias

Ajouter dans votre fichier `.bashrc` l'alias suivant (pas très pertinent, c'est pour l'exercice !) :
```
alias p='echo $PWD'
```
puis exécuter ce fichier par la commande **`source .bashrc`**
Quel est alors l'effet de la commande `p` ?

Vérifier que la commande `rm` est redéfinie ("aliasée") en `rm -i` dans votre fichier de configuration, sinon **ajouter cet alias** (quel est l'intérêt de ceci ? voir le man de `rm`).

Vous **définir un alias** `lc` pour obtenir un listing long (`ls -l`) de tous les fichiers sources C.

NB. Vous pourrez à tout moment vous définir tout autre alias que vous jugerez utile…

1.3. Variables d'environnement, configuration du PATH :

** Comparons les variables USER, HOME, et PWD :
Placez-vous dans votre répertoire de login et **affichez les valeurs de vos variables USER, HOME et PWD**.
Allez dans le répertoire `/bin` et **refaites afficher les valeurs de ces variables**.
Conclusion ? (assurez-vous d'avoir bien compris ce que représentent ces variables)

NB. Vous pouvez en apprendre plus sur les variables par exemple ici :
http://tldp.org/LDP/Bash-Beginners-Guide/html/sect_03_02.html

** Faites afficher la valeur de votre **variable PATH**, que contient-elle ? Que signifie-t-elle ?
Faites maintenant l'expérience suivante :

- Que se passe-t-il si vous exécutez la commande `bienvenue.sh` ?
- Ce script est dans `/home/p/pierrlau` ! **Ajouter** ce chemin dans votre PATH (pas

dans le fichier de configuration, mais simplement dans le shell courant)

• Que donne maintenant l'exécution de la commande ci-dessus ? Pourquoi ?

• Et que donne l'exécution de :

```
which bienvenue.sh
```

(consulter le man de la commande *which* pour en comprendre l'utilité)

• Pouvez-vous toutefois voir le contenu de `/home/p/pierrlau` ? Pourquoi ? (rappelez-vous l'explication sur les permissions des répertoires…)

NB. Rappelez-vous désormais que vous pourrez enrichir à tout moment votre variable **PATH** dans votre **fichier de configuration** si nécessaire (chaque fois que vous aurez besoin d'ajouter un nouveau chemin d'accès à un exécutable).

2. Tubes (pipes) et substitution de commande

On rappelle que :

- le **tube** (pipe) permet de rediriger la sortie standard d'une commande sur l'entrée standard d'une autre commande (il n'y a donc aucun sens à l'utiliser sur les commandes qui ne travaillent pas sur les E/S standard !)

  Exemple : `ls | wc -l` donnera le nombre de fichiers du répertoire courant

  → assurez-vous de comprendre pourquoi !

- la **substitution de commande** par le **backquote** permet l'utilisation du résultat d'une commande dans une affectation ou en paramètre d'une autre commande

  Exemples :
  ```
  nbfic=`ls | wc -l`
  echo `ls | wc -l`
  ```

  → assurez-vous de comprendre ce qui se passe dans chaque commande !

2.1. Les commandes suivantes **ont-elles un sens** (quant à l'utilisation des notions vues ci-dessus : variables, tube, backquote) ? **Si oui**, que signifient-elles précisément, et **si non**, pourquoi ? Si nécessaire, créer des fichiers permettant de les tester.

```
cd $USER/Tp1
cd $HOME/Tp1
tail -10 pg2.c | grep "fin"
pwd | chmod u+x
chmod u+x `pwd`
```

Noter le code renvoyé par l'exécution de chacune de ces commandes (indication de son succès ou de son échec) en affichant la valeur de `$?` juste après son exécution, et vérifier que cela est bien cohérent avec le rôle de cette variable spéciale.

NB. Voir ici un rappel sur les variables dans le shell, et notamment `$?`
https://www.epons.org/shell-bash-variables.php

2.2. Ecrire la commande qui permet de placer dans une variable `nbficetc` le nombre de fichiers de type fichier simple contenus dans le répertoire `/etc`.

Pour reconnaître les fichiers simples, on utilisera le fait que dans ce cas la ligne obtenue par la commande `ls -l` commence par un –
Voir comment utiliser la commande `grep`, en particulier en début et fin de ligne, ici :
https://azurplus.fr/comment-utiliser-la-commande-grep-sous-linux/

Vérifier que l'exécution de cette commande a réussi : quel est son code de retour ? quelle est la valeur de la variable `nbficetc` ?

3. Utilisation de find

3.1. Ecrire une commande qui permet de chercher tous les fichiers de la sous-arborescence du répertoire `/usr` qui sont de type répertoire et dont le nom commence par la lettre `l`.

3.2. Ecrire une commande qui permet de chercher tous les fichiers de la sous-arborescence du répertoire `TP` se trouvant dans votre répertoire de login pour lesquels vous avez les droits de lecture et écriture et le groupe a le droit de lecture, et de supprimer ce droit au groupe pour chaque fichier ainsi trouvé.

3.3. Ecrire une commande qui permet de chercher, dans toute la sous-arborescence du répertoire `TP` de votre répertoire de login, les fichiers sources C, et de rechercher dans chacun d'eux la présence d'une variable nommée `x` ou `X`.
Dans un premier temps on cherchera seulement le caractère `x` ou `X` dans le fichier.
Partie optionnelle : Dans un deuxième temps, on précisera la commande en vérifiant que le nom complet de la variable est `x` ou `X` : en fait, on se contentera de chercher cette lettre précédée d'un blanc, d'un "=" ou d'une "(", et suivi d'un blanc, d'un "=", d'un ";" ou d'une ")".

Puis on rafinera cette commande en ne cherchant que les fichiers sources C qui ont été créés ou modifiés il y a 10 jours.

4. Vers un script shell

4.1. Ecrire une commande qui utilise `who` pour déterminer si un utilisateur est actuellement connecté sur le device `pts/0` : observer comment se présente la sortie de `who` et y rechercher le nom du device à sa place précise.
(changer le nom du device si celui utilisé ci-dessus ne vous permet pas de tester réellement votre solution sur mandelbrot)

Partie optionnelle : lire l'annexe du cours (programmation en shell) et faire les questions qui suivent.

4.2. *Ecrire une commande conditionnelle qui affiche « utilisateur present » ou « pas d'utilisateur » suivant qu'un utilisateur soit actuellement connecté sur le device `pts/0` ou pas. (changer le nom du device si celui-ci ne vous permet pas de tester réellement votre solution)*

4.3. *Ecrire une commande qui répète le traitement ci-dessus pour tous les devices de `/dev` (commande à exécuter sans vous déplacer dans `/dev` bien sûr).*

4.4. *Ecrire un script shell qui prend en paramètre un nom de répertoire (le script vérifiera qu'il s'agit bien d'un répertoire) et effectue le travail ci-dessus, pour tous les fichiers de ce répertoire (bien sûr on testera sur `/dev`).*

Important : pour ce TP comme pour tous les autres, les exercices (non optionnels) non finis en séance devront être **finis avant la séance suivante**.