



---

## 1. Démarrage du PC.

Au démarrage du PC, vous avez le choix entre plusieurs OS :

- **Windows** : ...
- **Linux** : autre système d'exploitation de type unix
- **Linux** : autre version du précédent. C'est celui que vous choisirez. (Il s'agit d'un système sur lequel vous avez les droits "administrateur".)

Login graphique : Choisissez "Gnome" comme environnement graphique

**Login:** root (administrateur)

**Password :** TP\_reseau

**Remarque(s) :** vous pouvez également vous logger en mode "texte" en appuyant simultanément sur Ctrl, Alt et F1. Pour revenir à l'interface « graphique », il faut utiliser Alt-F7...

## 2. Utilisations des Menus

Pour vous familiariser avec l'environnement Gnome, on vous demande d'utiliser les différents menus à votre disposition et éventuellement lancer des applications telles que le navigateur internet, les éditeurs de texte ou l'outil de configuration Gnome. N'hésitez pas à modifier quelques options ...

## 3. Utilisations du Terminal

La plupart des utilitaires sont accessible depuis le "shell". Pour la suite du TP, lancer l'application "terminal" depuis les menus.

Voici une petite liste des commandes importantes :

- **man** : c'est une application qui permet d'obtenir le manuel d'un programme particulier. Par exemple, "man ls" vous donnera le manuel d'utilisation de la commande "ls".
- **ls** : cette commande affiche la liste de tous les fichiers, répertoires et autres dans le répertoire courant. (Voir plus loin pour la liste de répertoires traditionnels sous Unix.). N'hésitez pas à vous familiariser avec les arguments de cette commande (dont ls -l)
- **cd** : cette commande permet de changer de répertoire courant (cd=changedirectory). Pour aller dans le répertoire "/usr/bin/", il faut taper "cd /usr/bin".
- **rm** : pour effacer un fichier. Par exemple, "rm essai.txt" supprimera le fichier essai.txt dans le répertoire courant.
- **rmdir** : la même chose, mais pour effacer un répertoire.
- **mkdir** : pour créer un répertoire.
- **pwd** : permet d'afficher le chemin du répertoire courant. (pwd = print working directory)

**Remarque(s) :** Certaines commandes (comme "cd" ou "pwd") ne sont pas de véritables programmes (elles ne correspondent pas à des fichiers exécutables). Pour obtenir l'aide de ces commandes, il faut faire "man bash-builtins".

- **chmod** : permet la gestion des droits sur les fichiers.

### □ Questions :

1. Quelle commande utilisez-vous pour obtenir le manuel de la commande "man" ?
2. Utiliser la commande "pwd" pour savoir dans quel répertoire vous vous trouvez. Notez le résultat.
3. Déplacez-vous dans le répertoire "/bin/". Est-ce que ce répertoire contient un fichier appelé "ls" ? Est-ce qu'il contient un fichier appelé "cd" ? Qu'en pensez-vous ?
4. Retournez dans le répertoire précédent. (Celui dont vous avez noté le nom en b). Une manière rapide de retourner dans l'ancien répertoire courant est d'utiliser la commande "cd -". Essayez... Est-ce que ça marche ?

5. Retourner dans votre répertoire personnel. (Une autre manière de rejoindre votre répertoire personnel est de taper la commande "cd", sans aucun nom...) Créez un répertoire qui s'appelle "mon dossier". Est-ce que ça marche ?

## 4. Droits d'accès

Chaque fichier possède des droits qui permettent de restreindre les actions possible sur le dit-fichier. On peut avoir les droits de :

- lecture (pour visualiser le fichier)
- écriture (pour modifier le fichier)
- exécution (pour l'exécuter)
- une combinaison arbitraire de ces droits.

### □ Questions :

Pour un traitement plus fin, on peut utiliser des droits différent pour l'utilisateur propriétaire du fichier, le groupe propriétaire du fichier, et tous les autres utilisateurs.

1. Créer un fichier vide (ou un répertoire) avec un nom que vous aurez choisi (« test »). A quel groupe appartient il ?
2. Changer les droits de ce fichier ((user, group et others); (lecture, écriture, exécution)) à l'aide de paramètre à représentation symbolique puis en représentation octale.
3. Supprimez les droits de lecture sur le fichier "test". Essayer de l'ouvrir avec un éditeur de texte. Remettez les droits de lecture, mais supprimez les droits d'écriture. Essayez de l'ouvrir avec un éditeur de texte et rajouter une ligne au fichier et essayez de le sauvegarder.

## 5. Chemins d'accès

La plupart des programmes se trouvent dans un des répertoires suivants :

- /bin/
- /sbin/
- /usr/bin/
- /usr/sbin/

Lorsque vous tapez un nom de programme, le système vérifie que le programme existe dans un de ces répertoires. Si le fichier existe, il est exécuté. Sinon, un message d'erreur apparaît.

Pour exécuter un programme qui n'est pas dans un de ces répertoire, il faut donner son chemin d'accès. Par exemple, la commande "/usr/local/bin/démineur" exécutera le programme "démineur" dans le répertoire /usr/local/bin (si le fichier correspondant existe).

Pour indiquer le chemin d'un fichier, on peut utiliser soit un chemin absolu (par exemple /usr/bin/nice) soit un chemin relatif (par rapport au répertoire courant : ./ permet de parler du répertoire courant, et ../ permet de parler du répertoire au dessus.)

**Remarque(s) :** la touche Tab permet, si cela est possible, de demander au système de compléter un chemin d'accès incomplet. Par exemple, au lieu de taper "cd /root/ABCDEFGHIJKLMNORSTUVWXYZ", vous pouvez taper "cd /root/AB" puis Tab. Le système rajoutera "CDEFGHIJKLMNORSTUVWXYZ"

### □ Questions :

1. Expérimentez avec la touche Tab. (En créant des répertoire et des fichiers avec des grand noms...). Que se passe-t'il si vous avez un répertoire « ABCD » et un répertoire « ABEF » ?
2. Copiez le fichier /bin/ls dans votre répertoire "mon dossier" en lui donnant le nouveau nom "LS".
3. Comment faire pour exécuter le programme LS ?
4. Si vous êtes dans le répertoire "mon dossier", quel sera l'effet des commandes suivantes ?
  - LS
  - ./LS
  - ../LS
  - ../mon dossier/LS
  - ../mon dossier/././LS

**Remarque(s) :** Le Filesystem Hierarchy Standard (FHS) disponible sur (<http://www.pathname.com/fhs/pub/fhs-2.3.html>) décrit les conventions de nommage et l'organisation des différents répertoires et fichiers systèmes dans les systèmes de la famille Unix (dont fait parti bien entendu Linux). Reportez vous à la documentation concernant le : /bin ; /sbin/ ; /lib/ ; /usr/bin/ ; /usr/sbin/ ; /usr/lib/ ;

/home/ ; /root/ ; /dev/ ; /proc/ ; /etc/ ; /tmp/ ; /var/ et n'hésitez pas à parcourir ces répertoires afin d'observer les fichiers qu'ils contiennent !!

## 6. Execution avancée de programme

Lorsque vous lancez un programme à partir du terminal, le terminal est bloqué jusqu'à ce que le programme en question se termine. Il est possible de dire au programme de s'exécuter "en arrière plan", c'est à dire que le terminal continuera de fonctionner pendant l'exécution du programme. La syntaxe est : "nom\_programme &"

La commande "sleep" ne fait rien d'autre que d'attendre un certain nombre de secondes avant de terminer. Ce nombre doit être donné en argument. (ex : "sleep 10" permet d'attendre 10 secondes.)

### □ Questions :

1. Essayez les commandes "sleep 4" et "sleep 4 &". Quelles différences constatez-vous ?
2. Dans le second cas, vous ne savez pas vraiment quand la commande se termine. Essayez les commandes "(sleep 10 ; echo "FIN")" et "(sleep 10 ; echo "FIN") &". Que constatez-vous ? (La commande "echo "FIN"" permet d'afficher "FIN" sur l'écran, et le point virgule ";" permet de séquentialiser des commandes...)
3. En utilisant le même schéma que dans le 2, lancez plusieurs commandes en arrière plan. (Par exemple, une commande qui attend 25 secondes avant d'afficher "FIN 25 secondes", une autre qui attend 1 minute avant d'afficher "une minute, c'est long !".)

Vous pouvez obtenir la liste des programmes qui s'exécutent dans le terminal en utilisant la commande "jobs".

4. Refaire si besoin le point c et utiliser la commande "jobs". Comment interprétez-vous le résultat ?

Pendant qu'un programme s'exécute en premier plan (pas de "&"), il est possible de l'arrêter en appuyant (en même temps) sur "Control" et "c". On peut suspendre un programme en appuyant sur "Control" et "z".

5. Lancez les commandes suivantes :

```
(sleep 60 ; echo "PREMIER")
```

```
Control-c
```

```
(sleep 60 ; echo "DEUXIEME")
```

```
Control-z
```

```
(sleep 60 ; echo "TROISIEME")
```

```
jobs
```

Comment interprétez-vous le résultat de la commande "jobs" ?

Si on a la liste des processus lancés à partir du terminal (commande "jobs"), il est possible de passer un programme en arrière plan avec la commande "bg %n" où n est le numéro du processus dans la liste. (bg=background) De la même manière, il est possible de passer un processus en premier plan avec la commande "fg %n".

6. Essayez d'utiliser ces commandes...

7. Essayez de comprendre ce que fait la commande "exec nom\_de\_commande".

## 7. Variables d'environnement

Lorsque le shell s'exécute pour la première fois, de nombreuses variables d'environnement sont créées. Ces variables d'environnement contiennent des informations sur le système, l'utilisateur ou les programmes. On peut en rajouter de nouvelles pour utilisation personnelle. Habituellement, les variables ne contiennent que des majuscules et le symbole \_ ; pour les afficher à l'écran, il faut utiliser "echo \$NOM\_DE\_VARIABLE" (ne pas oublier le dollar).

Voici quelques exemples de variables d'environnement :

- **HOME** : contient le chemin d'accès de votre répertoire personnel,
- **UID** : contient votre numéro d'utilisateur,
- **USER** : contient votre nom de login,
- **LINES (et COLUMNS)** : si vous êtes dans un terminal, LINES contient le nombre de lignes de votre terminal (pareil pour COLUMNS)
- **OLDPWD** : contient le chemin d'accès de votre précédent répertoire
- **PS1** : contient une description du prompt : ce qui précède chaque commande lorsque vous êtes dans un terminal
- **PATH** : contient la liste des chemins d'accès des programmes. Comme le chemin "/bin/" est dans PATH, vous n'avez pas besoin de taper "/bin/ls" mais vous pouvez vous contenter de taper "ls".

### □ Questions :

1. Quel est votre numéro d'utilisateur ? Quel est celui de votre proche voisin ?

2. Regarder la valeur de LINES et COLUMNS ; changer la taille de la fenêtre du terminal et recommencez. Vous pouvez modifier une variable en tapant "NOM\_DE\_VARIABLE=valeur". Par exemple, vous pouvez rajouter une valeur au PATH en utilisant "PATH=\$PATH:chemin".
3. Rajouter le chemin du répertoire contenant LS au PATH. Essayer d'exécuter la commande LS d'un répertoire quelconque.
4. La commande "cd -" permet de retourner dans le répertoire courant précédent. C'est en fait un raccourci qui veut dire "cd \$OLDPWD". Modifier la valeur de OLDPWD et utilisez la commande "cd -".

D'autres variables se rapportent au programme que l'on vient d'exécuter :

- **?** : contient la valeur de retour du programme précédent. (En général, c'est 0 si le programme c'est exécuté correctement, et une autre valeur sinon.)
- **\$** : contient le numéro de processus du processus courant.
- 5. Afficher le numéro de processus du shell dans lequel vous êtes ; vérifiez que c'est le bon en utilisant la commande "ps"

Pour affecter une variable, vous pouvez utiliser "NOM\_DE\_VARIABLE=valeur" ou bien "export NOM\_DE\_VARIABLE=valeur". Dans le premier cas, la variable est locale et n'est affectée que pour le processus courant ; dans le deuxième cas, elle sera "globale" et sera aussi affecté pour les processus fils.

6. Créez une variable locale "Variable\_temporaire" dont la valeur est "42" ; vérifiez que la variable est bien affecté. Lancer un second shell dans le même terminal en utilisant la commande "bash" et vérifiez que la variable n'existe plus. Quittez le processus fils (second shell) avec la commande "exit".
7. Même question , mais en utilisant une variable globale.

Vous pouvez supprimer une variable en utilisant la commande "unset NOM\_DE\_VARIABLE".

8. Supprimez la variable PATH. Que constatez-vous ?

## 8. Introduction aux scripts

Utilisez le logiciel nedit (ou un autre éditeur de texte) pour créer le fichier suivant :

```
#!/bin/bash
if (test -d "$HOME/Systeme-TP-1")
then echo "le répertoire existe déjà"
else echo "création du répertoire"
    mkdir Systeme-TP-1
fi
```

### □ Questions :

1. En utilisant la documentation, essayer de deviner ce que va faire ce fichier ?

Pour exécuter un script, vous pouvez soit :

- Utiliser la commande "bash nom\_du\_script"
- Soit rendre le fichier exécutable ("chmod u+x nom\_du\_fichier") et l'exécuter comme un programme normal. (Attention, pour pouvoir faire ça, il ne faut pas se tromper dans la première ligne du fichier...)

2. Exécuter le script plusieurs fois de suite. Est-ce que son comportement correspond à vos attentes ?
3. Modifiez le script pour que sa valeur de retour soit 0 quand le répertoire est crée et 1 quand le répertoire existait déjà. (Pour retourner la valeur n, il faut utiliser "exit n".) Testez le script.
4. Réaliser le script bsh de calcul du PGCD à l'aide de l'algorithme suivant :

```
Si diviseur == 0 Alors
    Afficher « erreur »
    Fin de programme
finsi
reste = 1
tant que reste != 0 faire
    reste = dividende mod diviseur
    dividende = diviseur
    diviseur = reste
finFaire
```