**Système d'exploitation Linux**

**• pwd:** Où suis-je ?

**• passwd :** Changer le mot de passe

**• cat /etc/passwd** : Liste des comptes existants

**• cat /etc/group :** Liste des groupes existants

**•su tp1 :** basculer vers le compte tp1

**• su :** basculer vers le compte root ( unbunto : sudo su )

**• groupadd gr1**: création d’un groupe gr1

**• useradd u1 –g gr1:** création d’un nouvel utilisateur u1 appartenant au groupe existant gr1

**• passwd u1:** changer le mot de passe de tp1

**• userdel -r u1:** suppression de l’utilisateur u1 avec ses fichiers personnels

**• ls –l /etc :** contenu du sous-repertoire /etc

- le répertoire « . »= représente un lien vers le répertoire courant.

- le répertoire « .. » = représente un lien vers le répertoire père.

- Le chemin peut être absolu ou relatif: cd /home/tp1/cours cd cours ( sous-rep du rep courant )

- Le caractère tilde (« ~ ») désigne le répertoire de connexion d’un utilisateur (le home directory)

**• cd** 🡪 retour au répertoire de connexion (home).

**• cd .. 🡪** monter d'un niveau dans la hiérarchie

• **cd ../.. 🡪** monter de 2 niveaux dans la hiérarchie

**• ls -l** 🡪 listé le dossier courant avec les attributs des fichiers

**• ls -a** 🡪 liste complète, y compris les fichiers et dossiers cachés, qui commencent

**• ls -lr** 🡪 liste "récursive".

**• ls -l -r 🡪** équivalente á ls -lr, ce qui signifie qu’on peut rassembler 2 options -l -r dans -lr.

**• ls --help** 🡪 pour tout savoir tout options et information sur la commande ls.

**• file fichier 🡪** renseigne sur la nature du contenu du fichier

**• mkdir rep** 🡪pour créer un sous\_rep du rep courant

**• rmdir rep** 🡪 pour supprimer un sous\_rep vide

**• ln fich1 fich2** 🡪 permet de créer un lien physique fich2 á partir de fich1.

**• ln -s fich1 fich2 :** permet de créer un lien symbolique de fich1.

**• touch fich1 fich2 :** crée une liste de fichiers vides, ici fich1 et fich2.

**• cat (texte court) ou less ou more (texte long) :** consulter un fichier texte.

**• vi ,pico, edit fichier :** (dépend de la version linux) permet de créer ou éditer un fichier.

**• find rep -name expression :** permet de rechercher les fichiers dans le rép (ou à défaut dans le rép. courant) avec une expression de sélection.

• **cp (copy, copier fichiers et répertoires):**

On distingue 2 usages :

– **cp [option] source destination**: copie d'un seul fichier, en précisant le chemin et le nom du fichier destination

– **cp [option] ens-fichiers-source rép-destination :** copie l'ensemble des fichiers dans le rép-destination.

**Option :**

* **-R,** recopie récursive, permet de copier toute une arborescence
* **-i** avertit l'utilisateur de l'existence d'un fichier du même nom et lui demande s'il veut le remplacer.
* **-p** effectue une copie en gardant le propriétaire et le groupe d'origine.
* **-v** affiche en clair le nom des fichiers copiés.

Exemple :

**cp -R /home /root/tmp** 🡪 crée une copie dans /root/tmp/home

**• rm [option] fichiers :** supprimer un fichier ou plusieurs, -r pour supprimer un répertoire, -f permet de supprimer même les fichiers protégés en écriture, -i pour afficher un message de confirmation á l’utilisateur.

Exemple:

**rm -rf /home/toto/tmp** 🡪 détruit sans préavis l'arborescence (si on en a le droit !)

**• mv [option] source destination :** si la source est un nom du fichier et la destination est un répertoire, mv déplace la destination dans le répertoire, si non si la destination ni un nom du répertoire, et ni un nom d’un fichier existant, alors mv changera le nom du la source vers la destination.

***Les droits:***

• Tout fichier du système appartient à la fois à un utilisateur (son "propriétaire") et à un groupe.

• Pour chaque fichier le monde de ses utilisateurs potentiels est scindé en 3 catégories, nommées :

- **u** : User (le propriétaire)

- **g** : Group

- **o** : Others.

- Linux a repris les 3 protections d'UNIX sur les fichiers et les répertoires. Leur notation symbolique est :

**-r :** lecture

**- w :** écriture

**- x :** exécution La présence d'un tiret

**- "-"** signifie l'absence complète de droits

**• chown [-R] nv-user fichiers :** Changer le propriétaire l'option -R (récursif) permet d'agir sur l'ensemble des sous répertoires.

Exemple :

**chown -R smi /home/stage1 :** le répertoire /stage1 appartient mnt á l’utilisateur smi

**• chgrp [-R] nv-groupe fichiers :** Changer le groupe propriétaire, effectué par root ou le propriétaire, à condition que celui-ci soit membre du nouveau groupe.

Exemple :

**chgrp -R smi /home/stage1 :** le répertoire /stage1 appartient mnt au groupe smi.

• Ajout, retrait ou fixation des permissions : Pour chaque fichier, on désigne par : u, g et o les 3 catégories (user, group, other), a pour all.

+ - = l'action d'ajouter, de retirer ou de fixer un droit, qui s'applique à chaque catégorie séparément  [u g o a] [+ - =] [r w x]

**- chmod u+x fichier** : "ajoute le droit d'exécution au propriétaire du fichier"

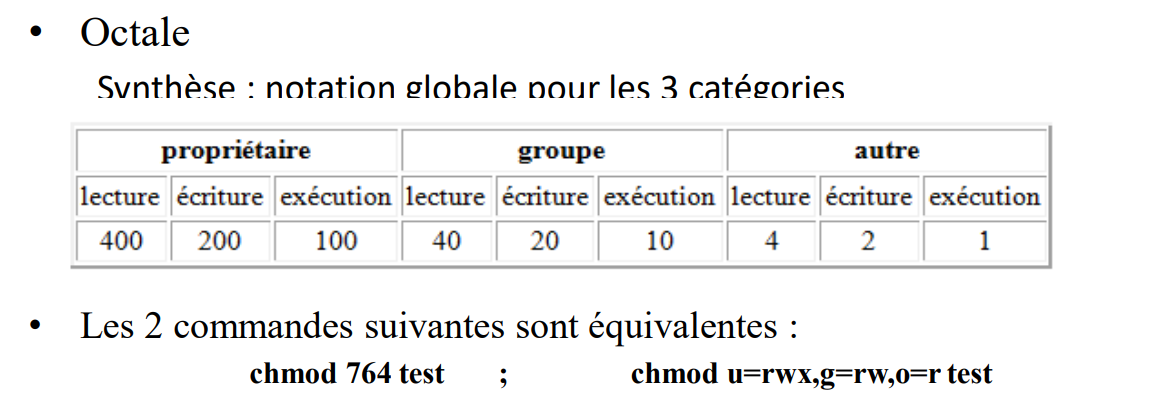
**- chmod go-rwx fichier** : "enlève tous droits d'accès à tous les utilisateurs, sauf au propriétaire"

**- chmod –R go-rwx /home/toto** : enlève toutes les permissions d'accès des fichiers du répertoire personnel de toto (et des sous-rép.), à tous sauf au propriétaire, c'est-à-dire toto.

•Notation absolue

**- chmod u=rwx,g=rw,o=r fichiers:** remplace les permissions précédentes des fichiers, en les fixant à -rwxrw-r--.

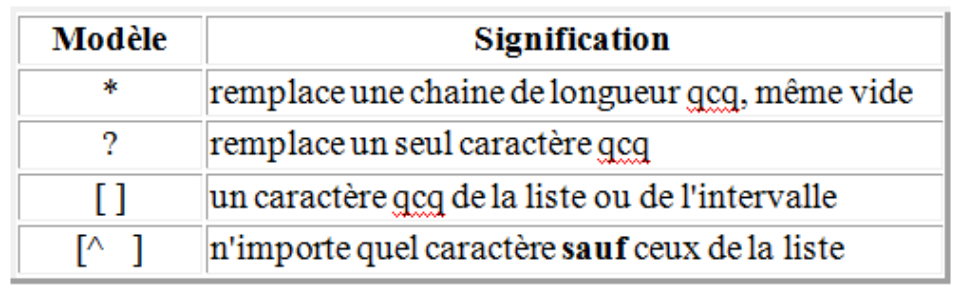
**- chmod u=rwx,g=r,o=** fichiers fixe les permissions à -rwxr-----.



•La commande umask permet de définir les droits par défaut pour les fichiers et répertoires nouvellement crées.

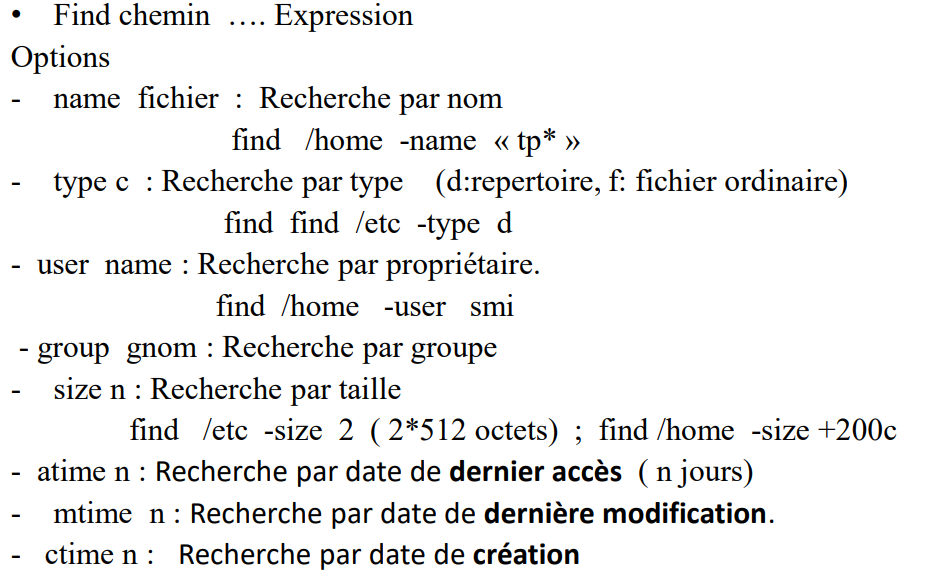
* Umask : affiche le masque de l'utilisateur actif
* umask masque : fixe les permissions ultérieures de création des fichiers de l'utilisateur actif, conformément à masque, en notation octale
* Masque par défaut est 022.

On a souvent à manipuler des ensembles de fichiers : on utilise des caractères spéciaux (appelés aussi métacaractères) Il existe quatre constructeurs de modèles \*, ? , [] et ^.



Désigner un ensemble de fichiers :

* **ll /\*/\*.d** : tous les fichiers d'un rép de / qui se terminent par .d ]\
* **ll -d /home /\*** : tous les sous-répertoires de /home
* **rm \*** : supprime tout le rép courant !
* **cp /lib/modules/\*/\*/\*.? /home/toto :** toto copie tous les pilotes dans son répertoire personnel
* **ll [a-dA-D]\* :**liste les fichiers du rép courant dont le nom commence par a, b, c ou d minuscule ou majuscule (y compris les sous-rép.)
* **cp ventes1[00-50].xls /home/toto/bilan**: copie tous les fichiers ventes100.xls jusqu'à ventes150.xls
* **ll [a-n]\*:** liste les fichier du rép courant dont le nom commence par a jusqu’à n.
* **ll [an]\*:** liste les fichier dont le nom commence par a ou n.
* **ll [^an]\*** lest les fichier dont le nom ne commence ni de a ni de n.



**Combinaison entre commandes :**

* **Pwd ; whoami ; date**  le ";" permet d’exécuter une séquence de commandes.
* **Commande1 && commande2**  "&&" dit au Bach que si la première commande a fonctionné, alors exécute s'il te plait également la deuxième commande.
  + **cd /home/tp1 && rm \*** : on supprime tout les fichiers si et seulement si le dossier /home/tp existe déjà.
* **Commande1 || commande2**  "||" signifie que la commande2 sera exécutée que si la commande1 a échoué de s’exécuter.
  + **cd tp || mkdir tp**  essayé de se déplacer vers le répertoire tp, si non crée le .

Toute commande retourne une valeur finale au shell : 0 pour lui indiquer dans le cas de reussie ou un autre nombre dans le cas d’erreur. Cette valeur numérique est stockée dans la variable spéciale $?`

* **echo "`whoami`, nous sommes le `date` "**  "**`**"permet d’exécuter une commande au sein d’une zone textuelle.
* **echo "$(whoami), nous sommes le $(date)** " "$(variable)"permet d’exécuter une commande au sein d’une zone textuelle.

**Redirections des entrées-sorties**

Toutes les commandes sont dotées par le système de 3 canaux de communication :

• entrée standard (stdin=standard input, clavier) pour lire des données

• la sortie standard (stdout, Écran) pour envoyer des résultats.

• et la sortie des erreurs (stderr).

– < fichier : utilisation de fichier comme entrée standard, lire des données á partir du fichier.

– > fichier : utilisation du fichier comme sorite standard, écrire dans le fichier.

Attention ! le fichier est créé et écrase sans préavis le fichier existant portant le même nom.

– >> (append) fichier : redirection de la sortie standard à la fin du fichier s'il existe déjà sinon il sera créé (sans écrasement)

– 2 > fichier : redirection de la sortie d’erreur vers fichier

– 2 >> fichier :la sortie d’erreur sera ajoutée á la fin du fichier

Exemple s

**- find /home –name tp > fichier\_sortie 2 > fichier\_erreur:** le résultat sera dirigé vers le fichier\_sortie et les messages d’erreur vers le fichier\_erreur

**- cat f1 f2 f3 >f4 :** f4 contiendra la concaténation des fichiers f1 f2 f3

**Les tubes de communications (pipe)**

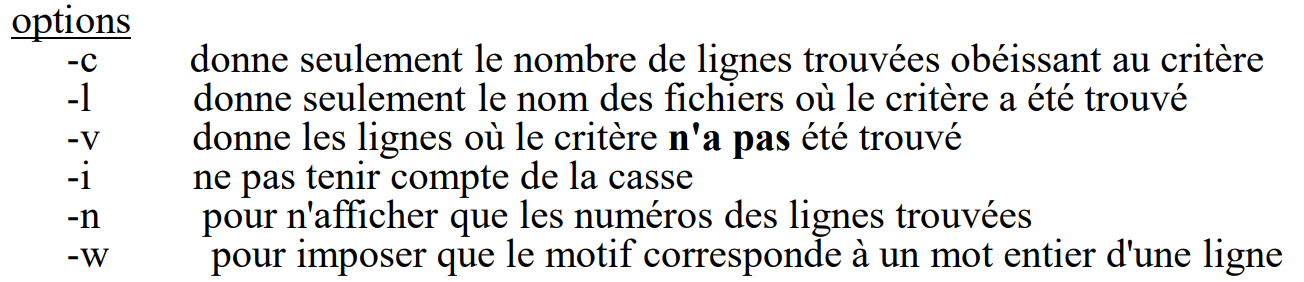
• Le symbole |, appel´e encore ”tube” ou ”pipe”, est utilisé pour relier 2 commandes entre elles. La sortie standard de la commande à gauche du symbole sera utilisée comme entrée standard de la commande de droite : Commande 1 | commande 2 | commande 3

* + **ls -l | wc –l**  affiche le nombre de ligne de la commande ls -l.
  + **ll | grep "rwxr-xr-x"**  affiche que les lignes qui ont les droit "rwxr-xr-x"

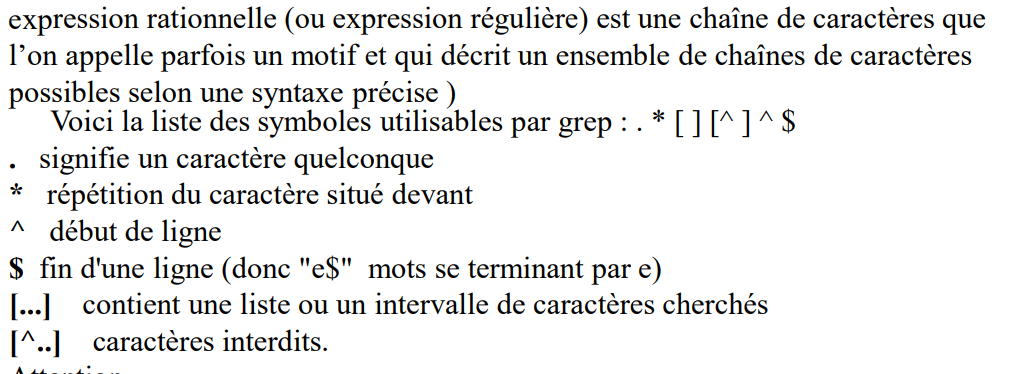
**La commande grep : sélection de lignes**

Cet commande sélectionne toutes les lignes qui satisfaisants une expression régulière.

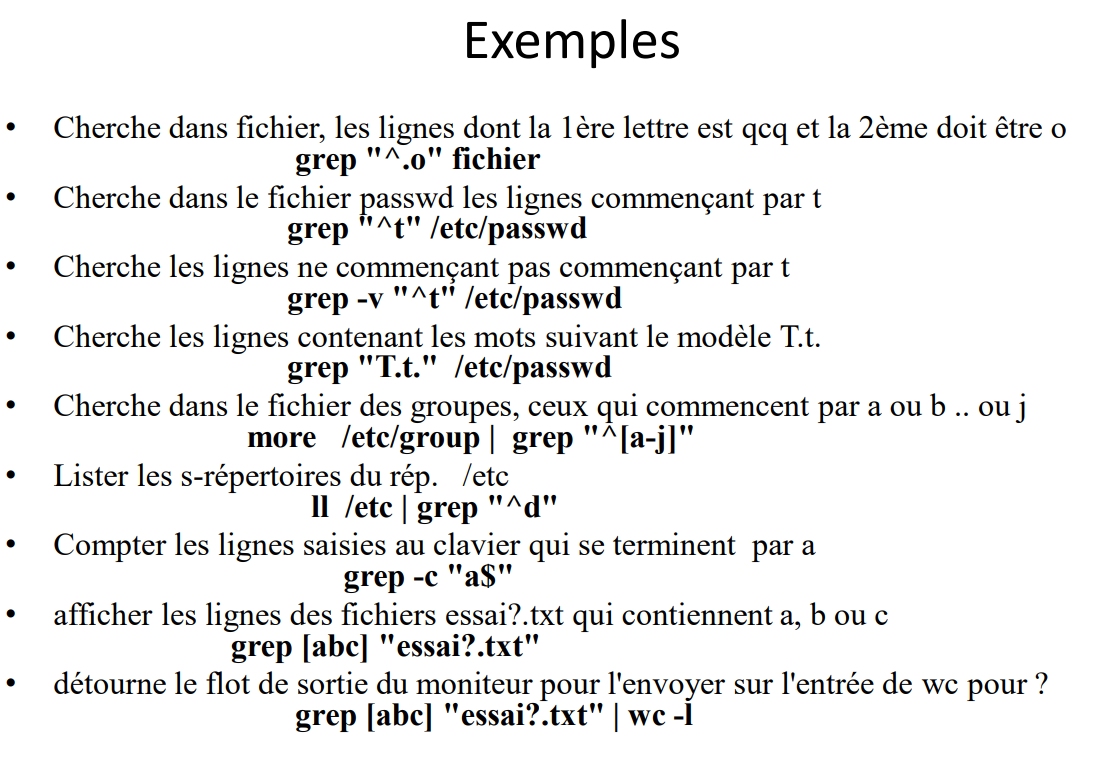
* Syntaxe : **grep [options] expreg [fichiers]**



**Expression régulier :**



**Attention**: Pour éviter une confusion entre les interprétations de ces symboles spéciaux par grep ou par le shell, il est indispensable de verrouille expreg en la plaçant entre guillemets " "



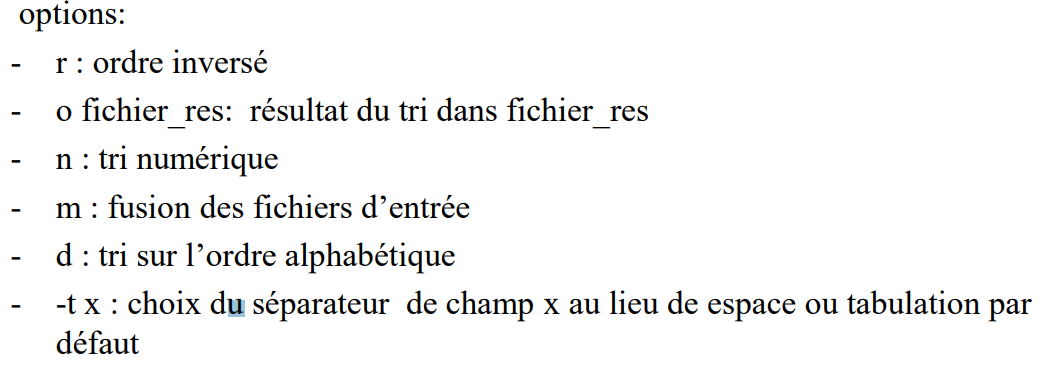
**Cut : sélection de colonnes, ou sélection des champs**

La commande cut présente 2 formes, forme de sélectionne des colonnes de caractères ou forme de sélection des champs séparés par un caractère bien précis.

* Sélection de colonne : cut -c(sélection\_colonnes) [fichiers]
  + **cut -c5 fichier**  affiche le 5ième caractère.
  + **cut -c5-10 fichier**  affiche du 5ième au 10ème caractères
  + **cut -c5,10 fichier**  affiche le 5ième et le 10ème caractères
  + **cut -c5- fichier**  affiche à partir du 5ième (jusqu'à la fin)
* Sélection des champs : cut -d "séparateur" -f(sélection\_champs) [fichiers].
  + **cut –d ": " –f1,3 /etc/passwd**  affichage le nom et le numéro (uid) de chaque utilisateur, en utilisant ": " comme séparateur.
  + **ls –l | grep " ^- " |tr –s " " | cut –d " " -f9**  liste des fichiers du répertoire courant
  + **tr -s** : supprime les espaces répétées.

**Commande Sort :**

Sort : permet de lire et de trier le contenu des fichiers.

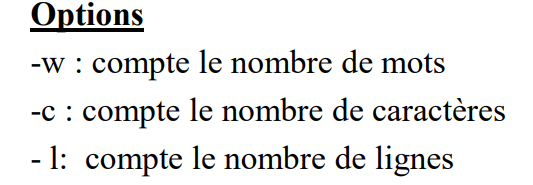


* **cat /etc/passwd |grep " bash$" |sort –o resultat**  affiche de façons triée les lignes qui se termine par Bash et mettre tout dans un fichier nommé résultat.

**Commande wc :**

Permet de compter le nombre des mots, caractère, ou ligne contenant dans un fichiers.

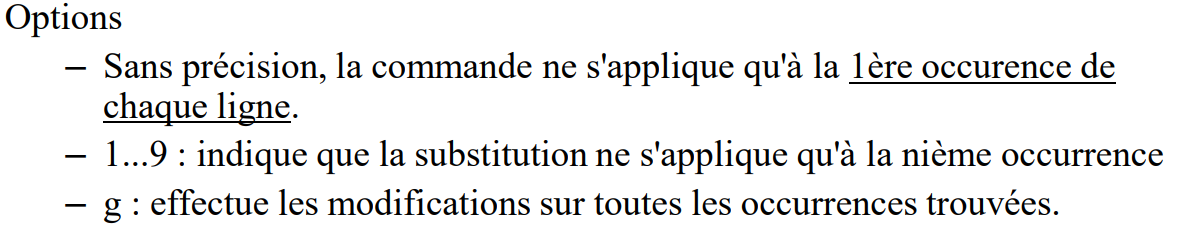
Syntaxe : **wc [options] [fichier]**



**La commande de substitution sed :**

Applique une substitution au mots qui satisfait l’expression régulier. **sed**

Syntaxe : **sed [adresse]s/expr-régulière/remplacement/options**



**- sed "s/[Cc]omputer/COMPUTER/g" fichier**  remplace Computer ou computer par COMPUTER sur toutes les occurrences dans fichier.

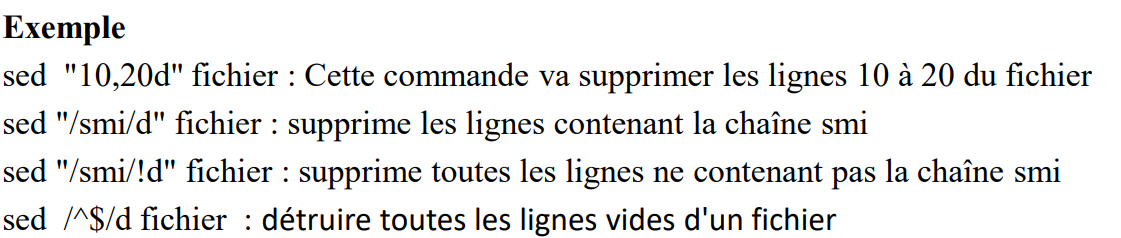
**- sed " s/moi/toi/g " fich.moi > fich.toi**  remplace moi par toi, le résultat redirigé vers un fichier toi.

**- sed -e "s/[Ff]raise/FRAISE/g" fichier** substitue toutes les chaînes Fraise ou fraise par FRAISE.

**La commande de suppression sed :**

La commande sed avec l’option d supprime les lignes comprises dans un intervalle donné.

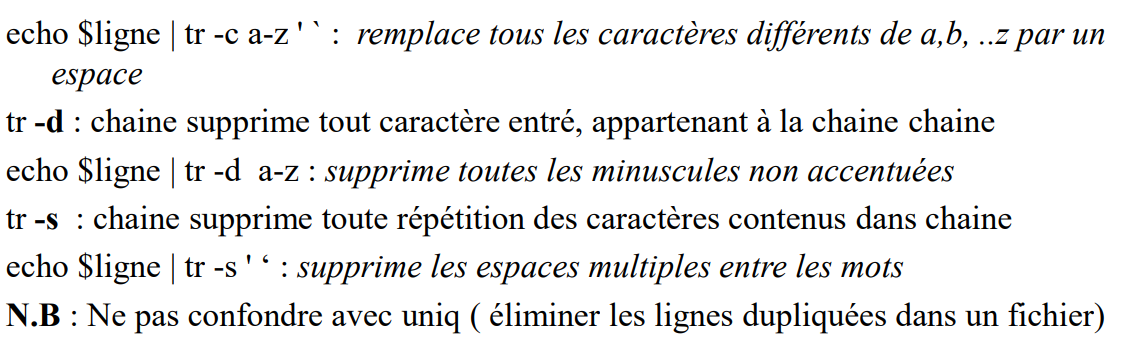
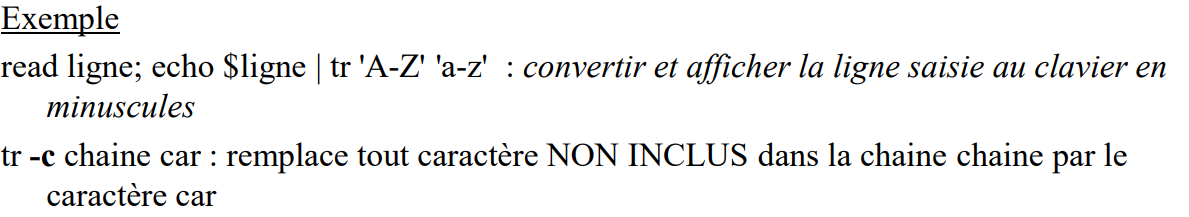
Syntaxe : **sed "n1,n2d" fichier**



* **sed "/smi/=" fichier :** afficher le numéro de la ligne contenant la chaîne smi
* **sed –n '10,20p' fichier** : sort sur stdout les lignes 10 à 20 du fichier
* **sed "/ˆ smi/w resultat" fichier** : Ecrire dans le fichier resultat toutes les lignes du fichier fichier commençant par la chaîne smi

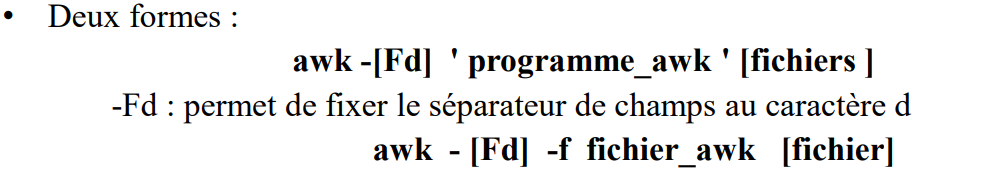
**La commande tr :**

Syntaxe : **tr [options] ch1 ch2 fich2**  Remplace toutes les occurrences de TOUS les caractères de ch1 par le caractère de ch2



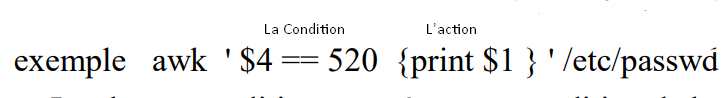
**La commande awk :**

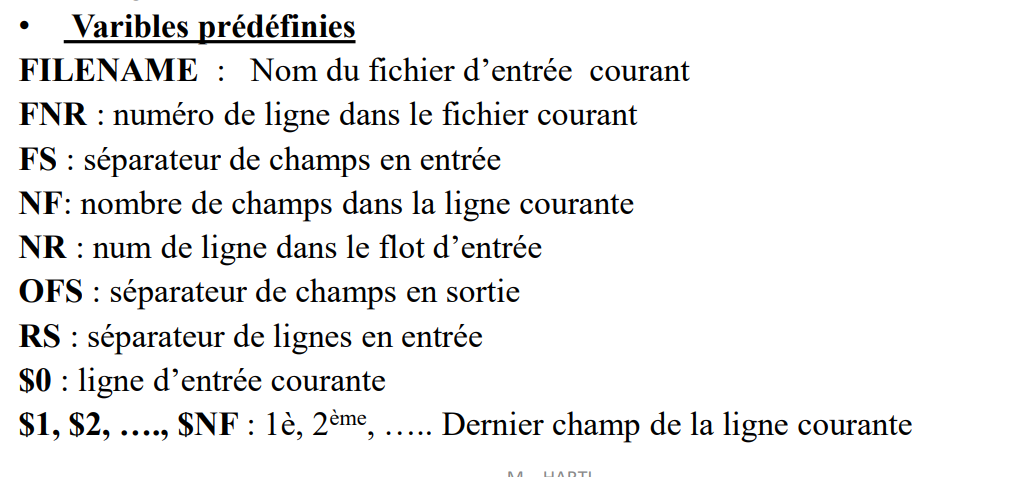
Un langage de programmation spécialisé dans la manipulation de texte.

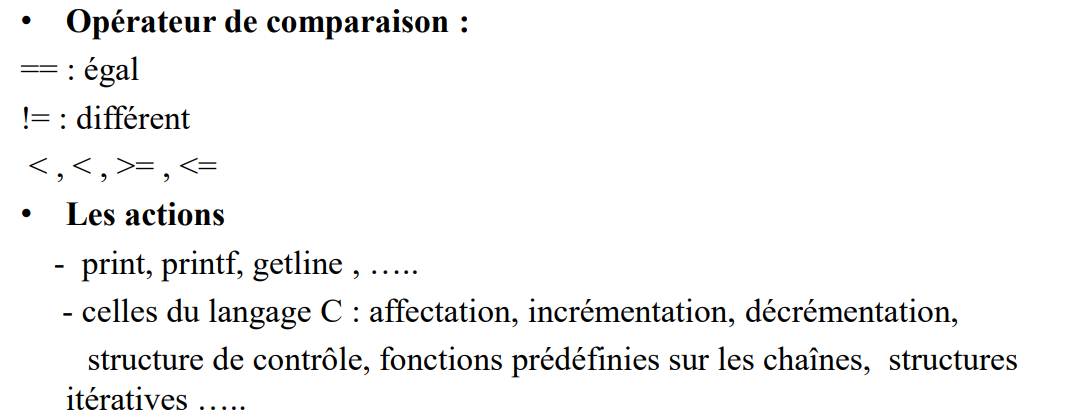


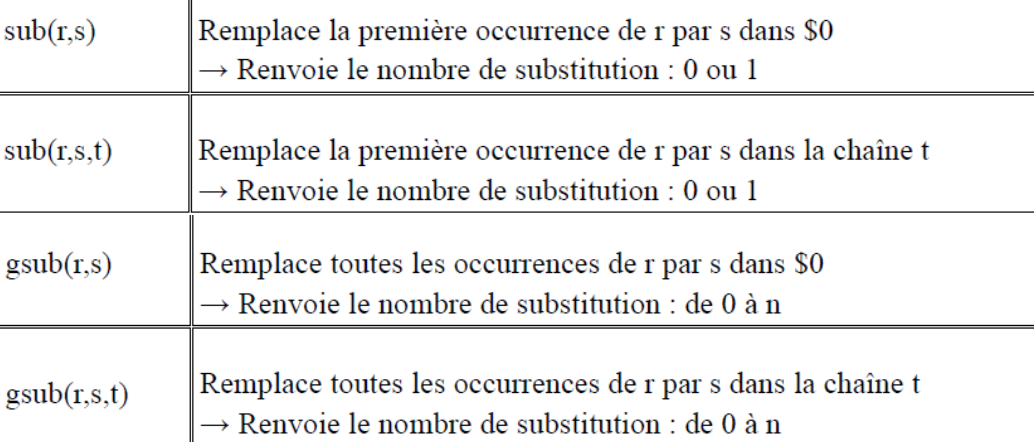
* **awk –F: '$5 == " " ' /etc/passwd** : sélectionne les lignes dont le cinquième champ est vide.
* **awk -F":" '{print "Le répertoire personnel de " $1 " est " $6}' /etc/passwd**  pour chaque ligne de /etc/passwd on affiche le champ 1 et le champ $6, avec le séparateur du champ ":" .

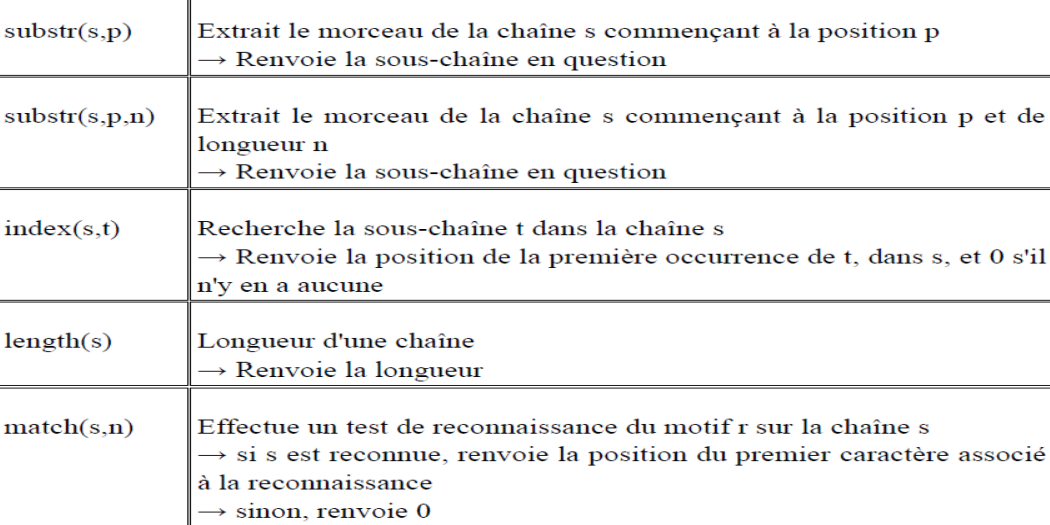
**Awk est une suite de ligne sous forme condition, action :**











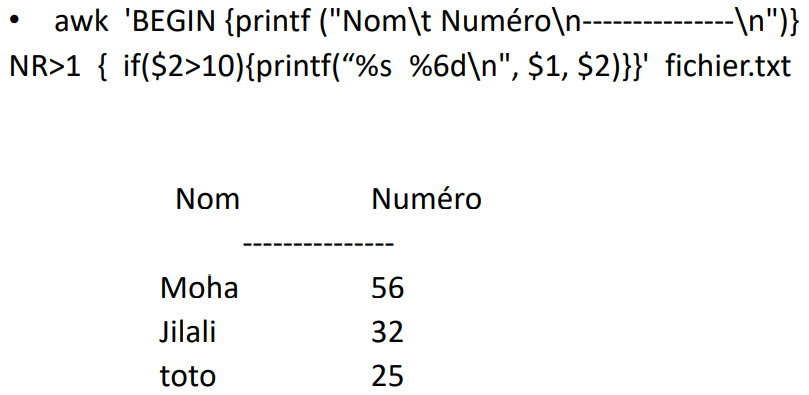
**• awk -F":" '{print "Le nombre de champs = " NF " et le dernier vaut " $NF }' /etc/passwd**  afficher pour chaque ligne de /etc/passwd le nombre des champs et le dernier champ.

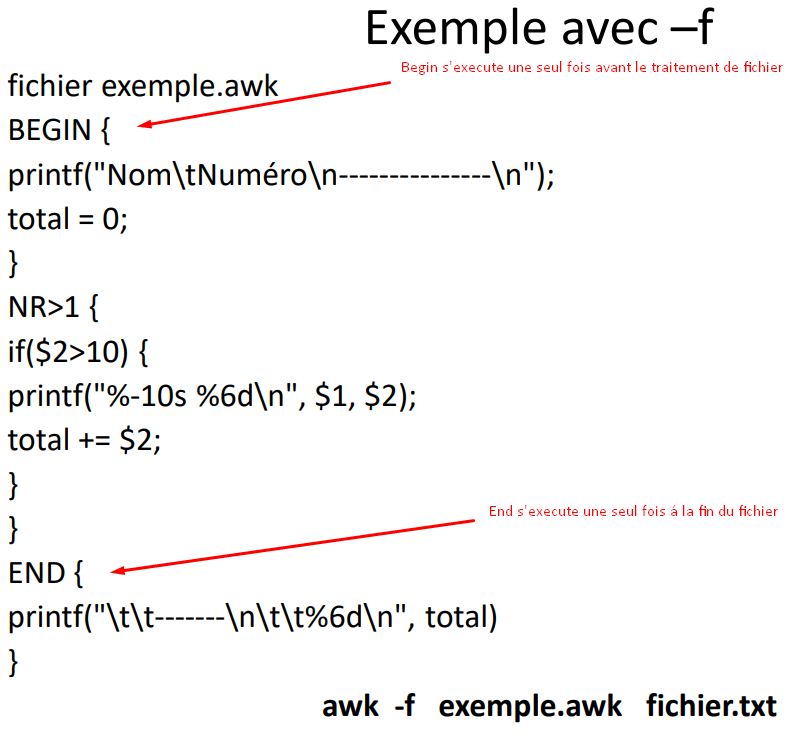
**• awk – F : ' $7 != " " {print $1 } ' /etc/passwd**  si le champ 7 différent n'est pas vide affiche le champ 1.

**• awk –F : ' $7 ~/sh/ {print $1,$2}' /etc/passwd**  si $7 contient la sous chaine sh affiche $1 et $2.

• **awk ' { mot=mot+NF} END { print NR, mot}' /etc/passwd**  calcule le nombre des mots, et afficher á la fin du traitement du fichier une seul fois, le nombre de ligne et le nombre des mots.

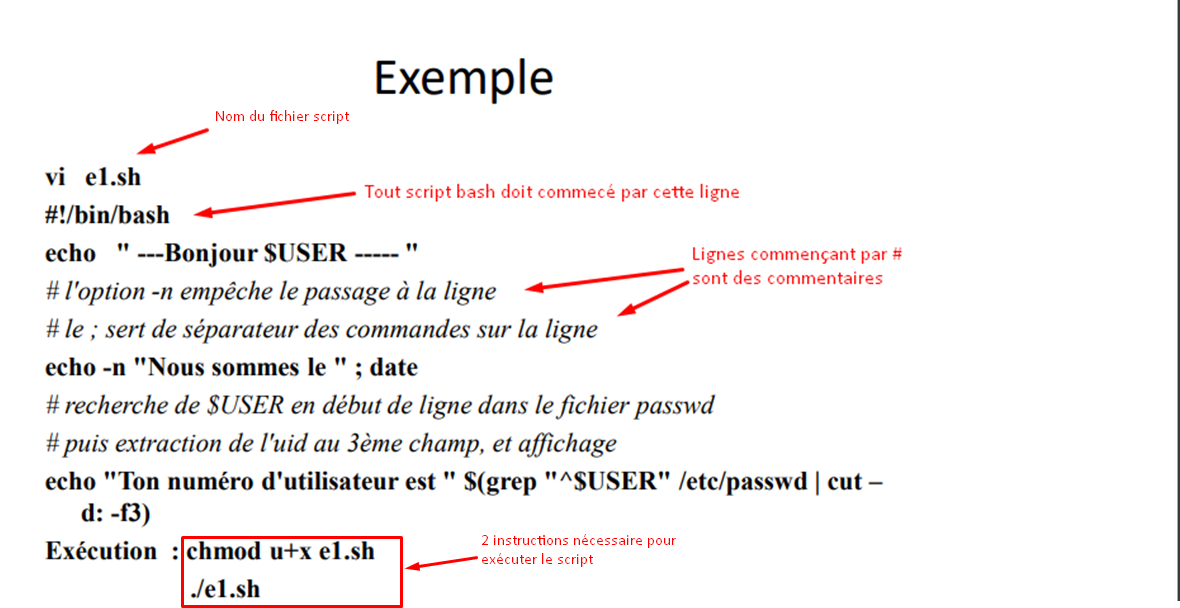
• **awk ' {som=0; for (i=1;i<=NF;i++) som=som+$i; if ( som < 100) print}' fichier**  sert a ajouté les calculer la somme des champs et de l'afficher si cette somme reste inferieur á 100.





**Script bash**

• Un script Bash est un fichier de type texte contenant une suite de commandes Shell, exécutable par l'interpréteur, comme une commande unique.



**Echo dans Script:**

Echo implique un retour á la ligne, l’option "-n" empêche la retour en ligne, l'option "-e" permet de prendre en charge les caractères spéciaux en milieu du texte tel que \n, \t … .

Exemple : echo -n "hello world" ; echo -e "hello \n, wolrd "

**Entrées : read dans Script:**

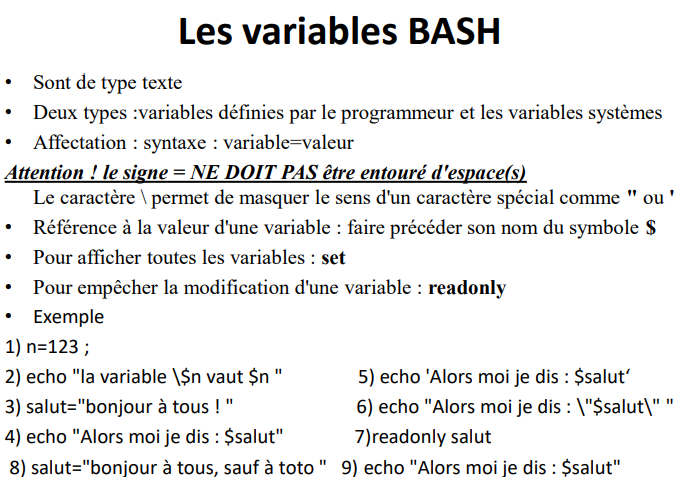
• permet l'affectation directe par lecture de la valeur, saisie sur l'entrée standard au clavier

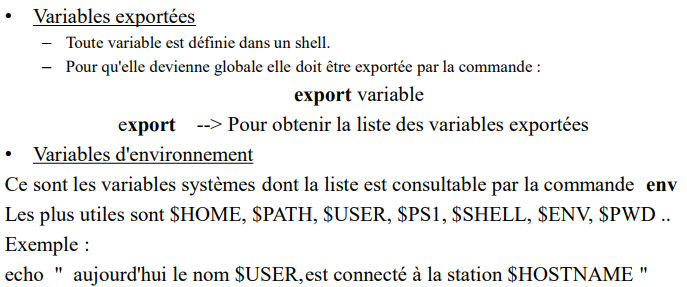
Exemple :

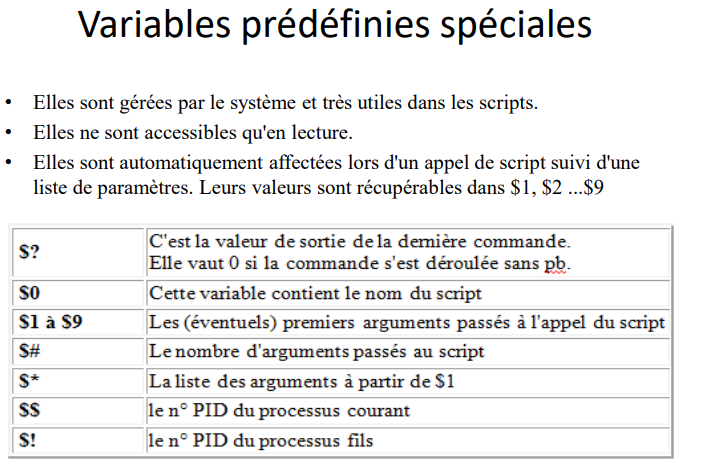
echo "Donnez votre prénom et votre nom "

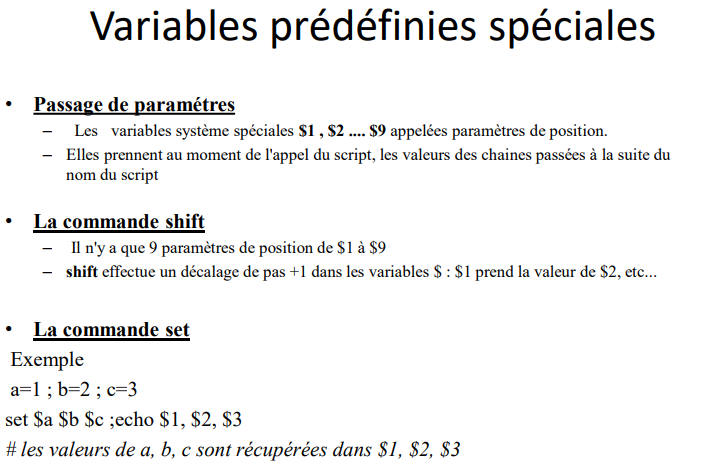
read prenom nom

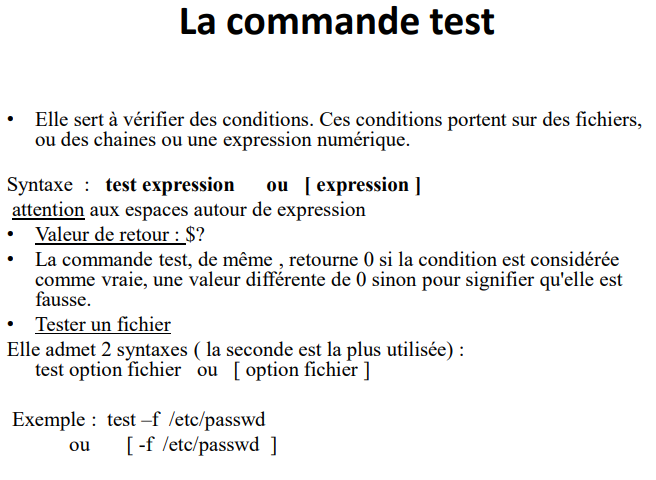
echo "Bonjour $prenom $nom"

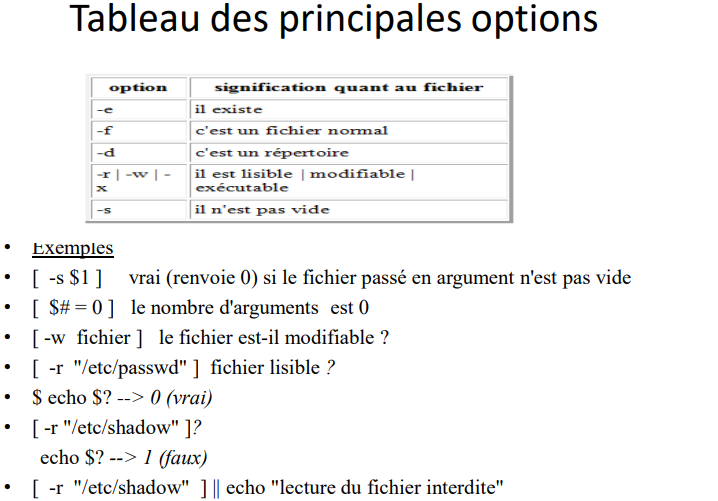


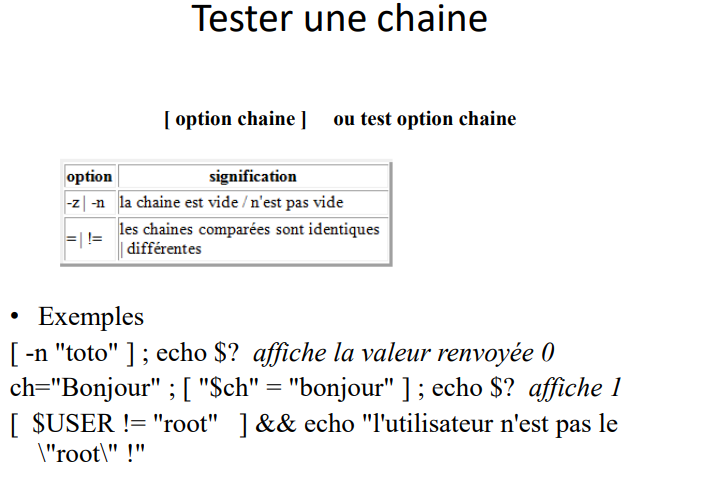


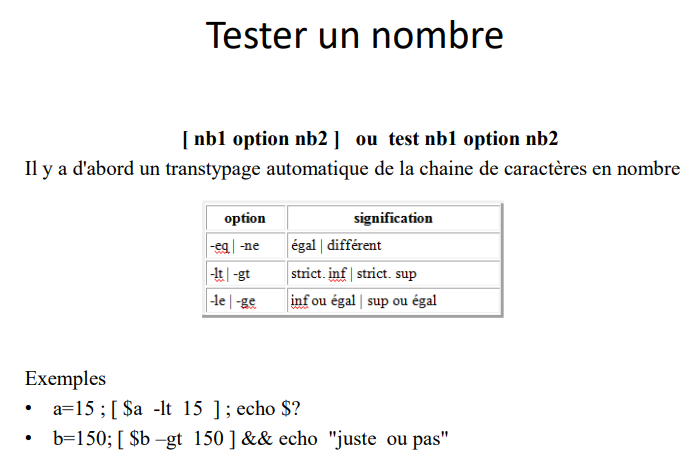


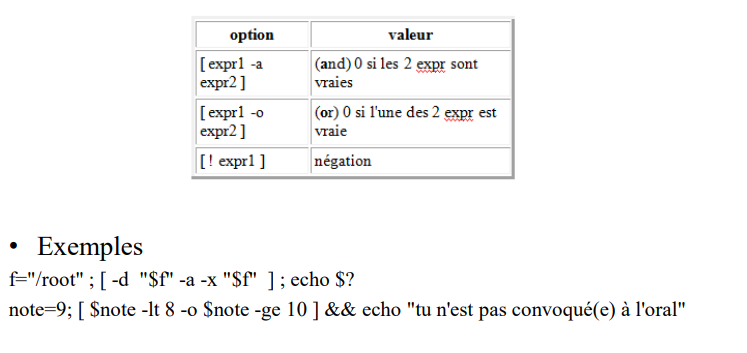


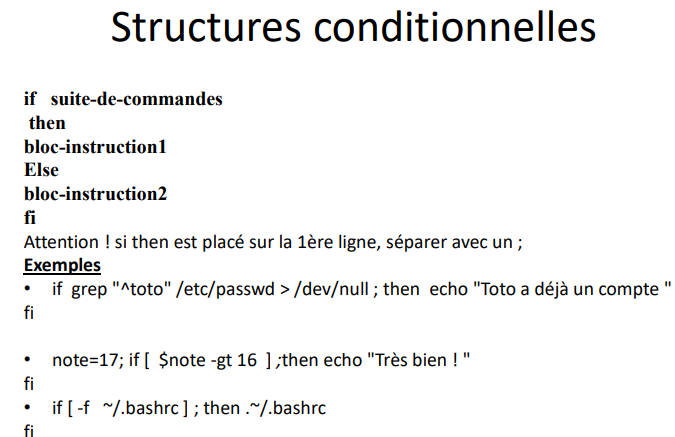


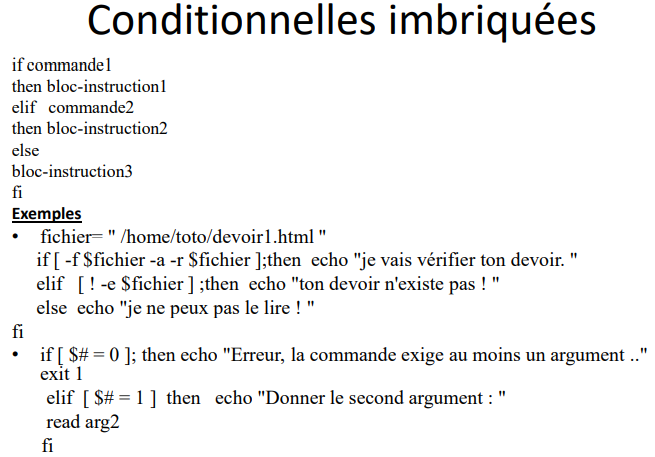


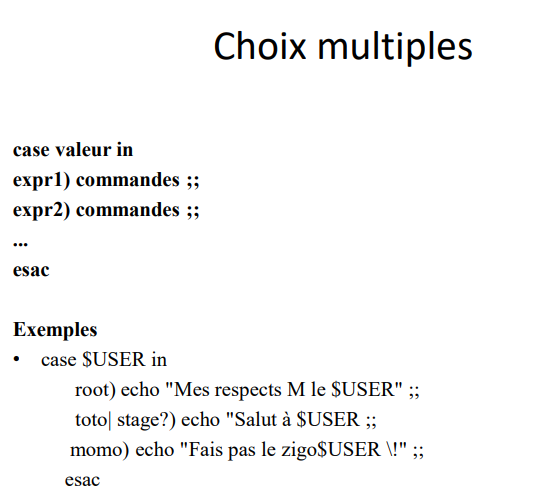


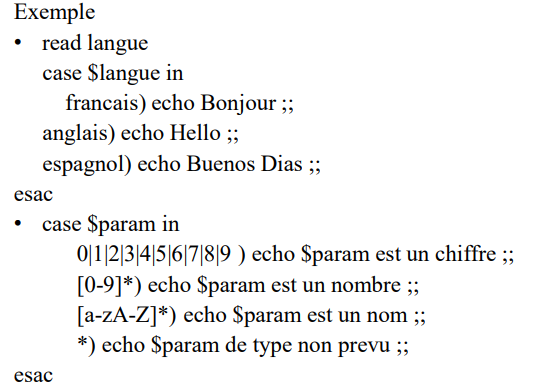


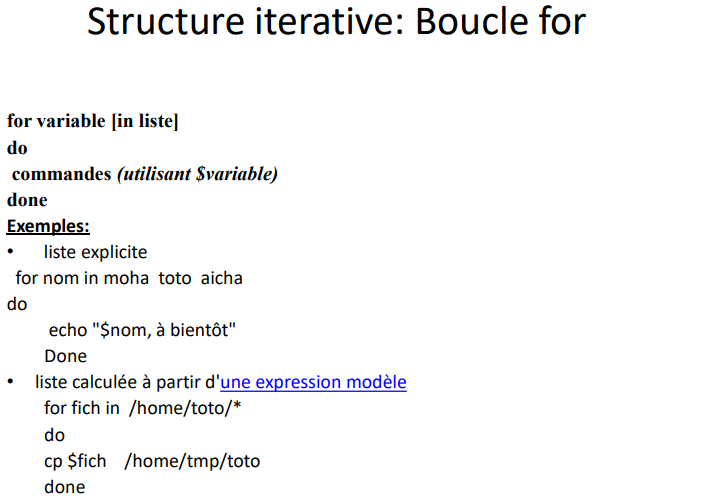


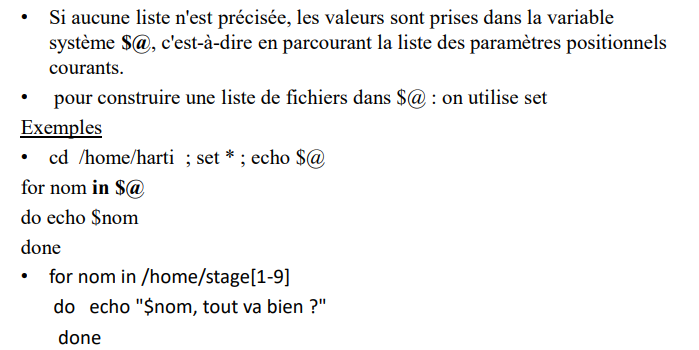




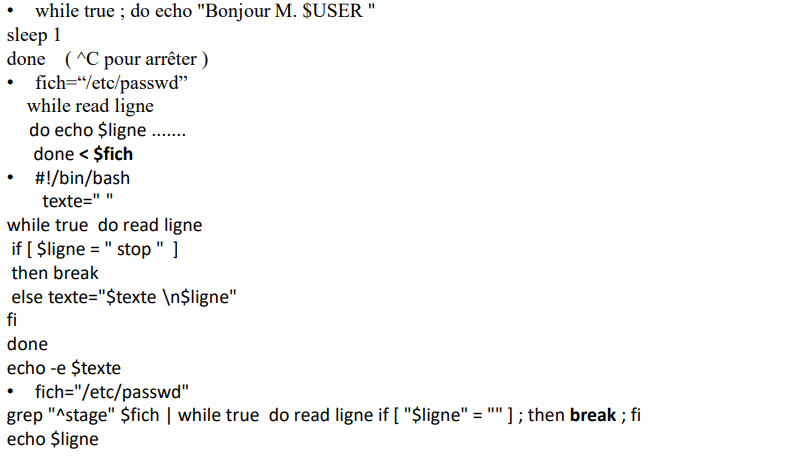




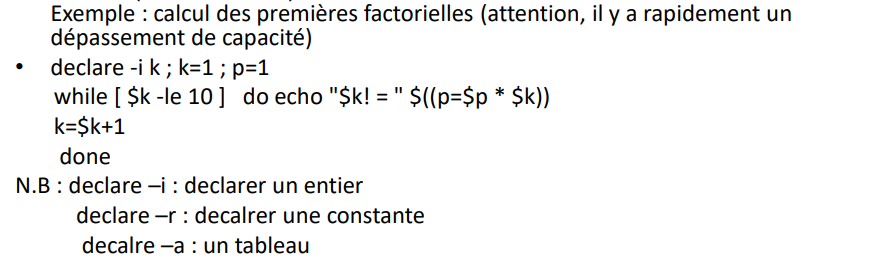


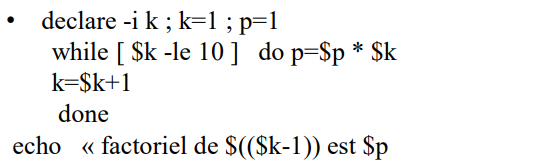


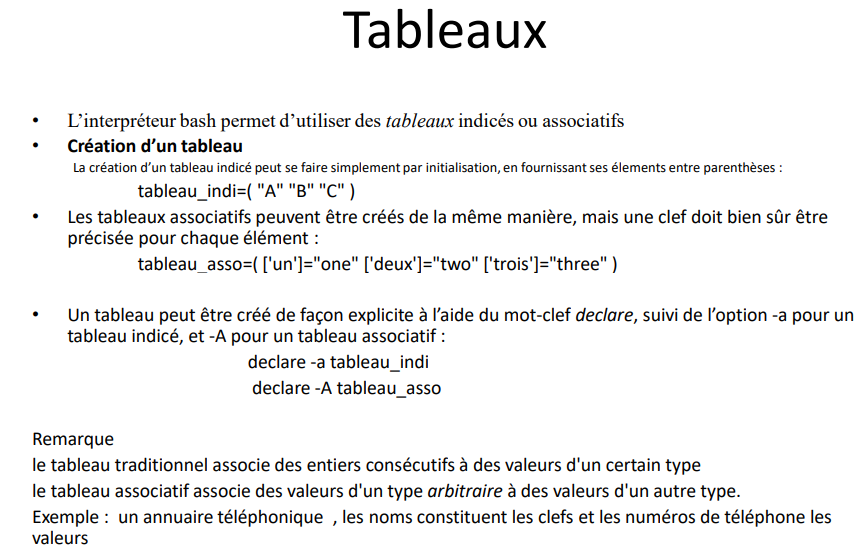
**La boucle while:**

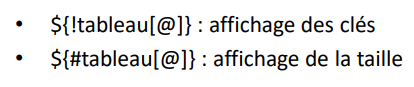
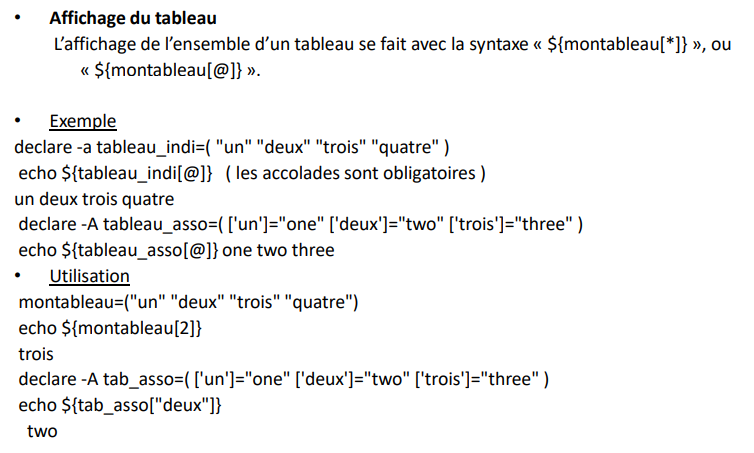


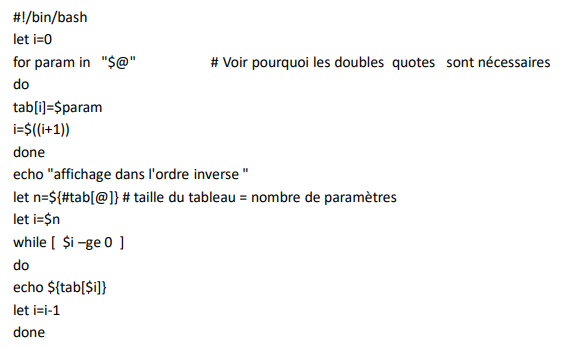
**Commandes diverses**

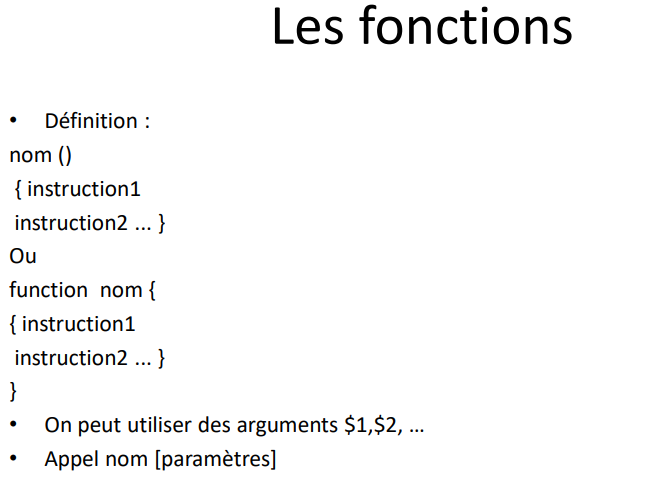


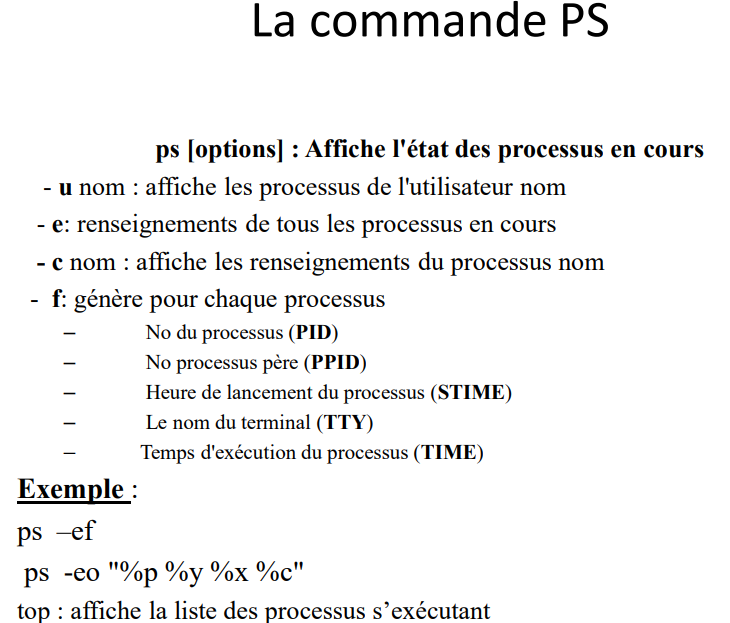


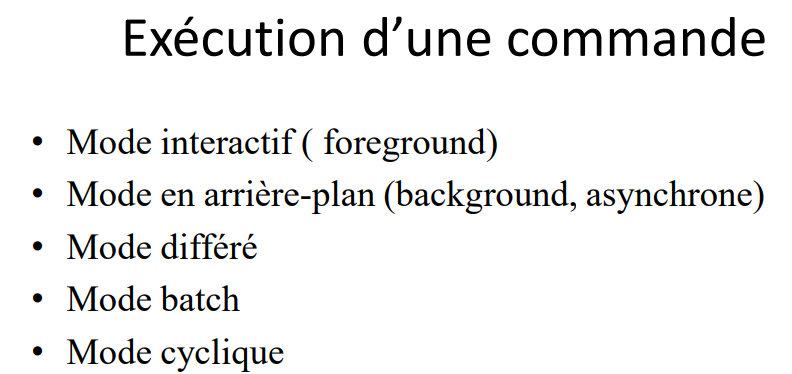


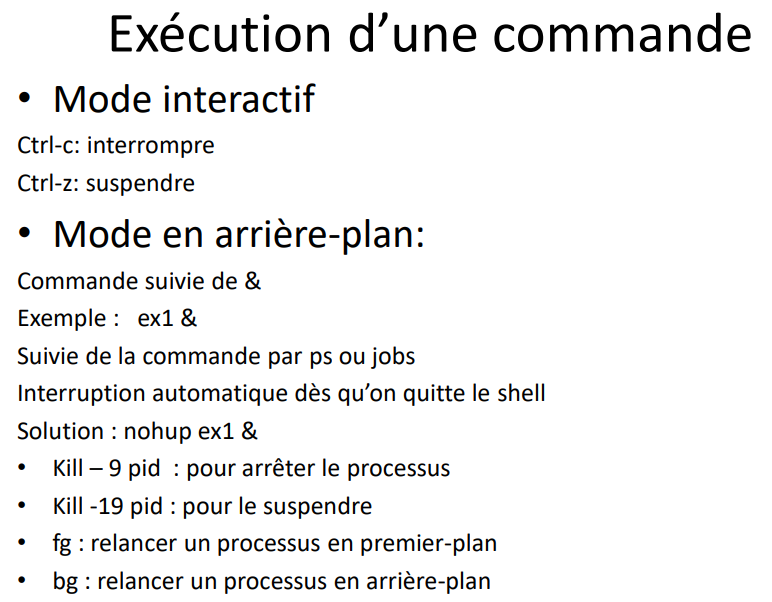


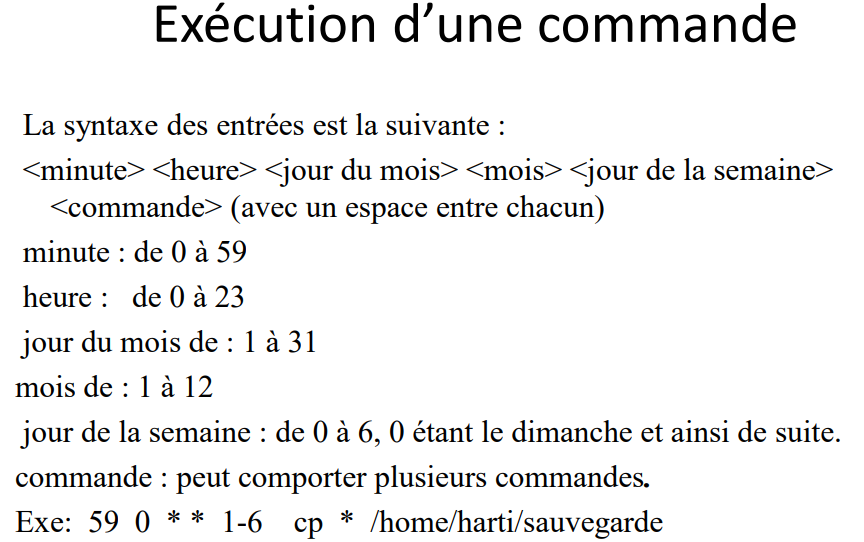


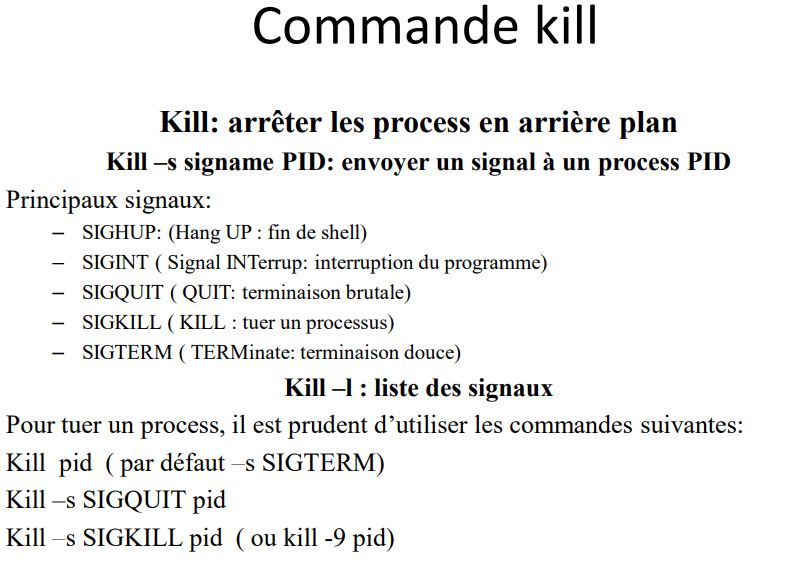


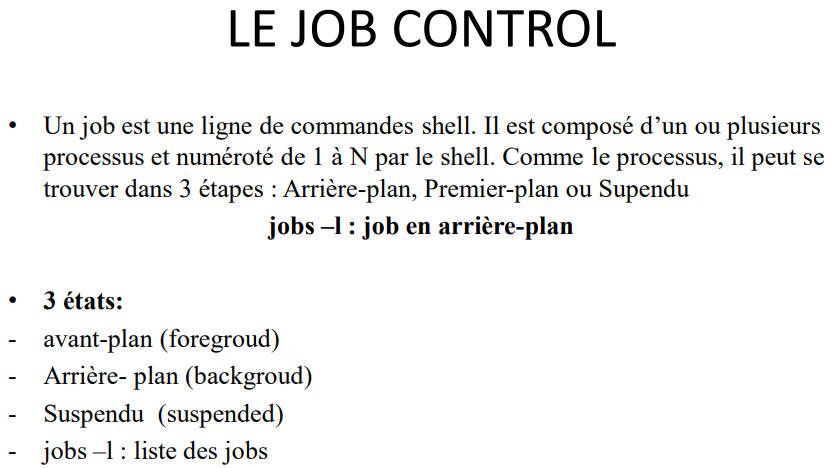


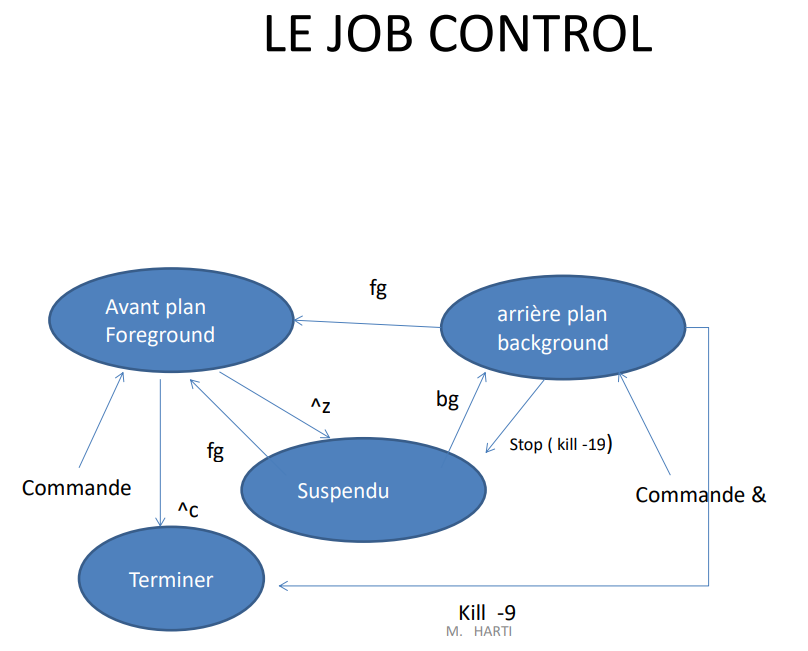












Queleque remarque :

* /dev / null → est le fichier poubelle.
* $USER est une variable qui contient le nom de l’utilisateur courant.

***Quelque commandes :***

**$ find / -type f -perm 666 -user $USER 2> /dev/null | wc -l**

→ savoir les fichiers qui appartient à l’utilisateur courant dont le droit d’acces = 666.

**$ find /etc -not -name "\*[0-9]\*"**

→ affiche que les fichiers qui comporte aucun chiffre.

**$ find /etc -name "??" -o -name "?" 2> /dev/null**

→ affiche les fichiers qui contient 2 caractères au plus, (-o → or) (-a → and).

**$ cut -d":" -f1,3 /etc/passwd | sort -t ":" -k 2 -n**

→ affiche 2 et 3 champ toute en triant numériquement selon champ deux, en utilisant ":" comme séparateur.

**$ tail -n5 test**

→ affiche les 5 derniers lignes.

**SED**

**$ sed '5d;7d' passwd**

→ afin de supprimer les lignes 5 et 7 dans un fichier.

**$ sed '5,7d' passw**d

→ supprime les lignes 5,6 et 7.

**$ sed '/^1/d' passwd**

→ supprime les lignes qui commence par 1.

**$ sed '/root/,/nobody/d' passwd**

→ supprime les lignes entre les chaines [root, nobody]

**AWK :**

**$ awk -F: 'length($0) >= 80' /etc/passwd**

→ affiche les lignes de plus de 80 caractères.

**$ awk -F: 'NF == 2{print $0}' /etc/passwd**

→ affiche que les lignes dont le nombre de champs exactement égal á 2.

**$ awk -F: '$0 ~/home/ {cpt++}END{print "nombre de home = "cpt}' /etc/passwd**

→permet de compter les ligne qui contient la sous chaine ‘home’.

**$ awk -F: '{for(i = 1;i < NF;i++) t[$i]++}END{for(i in t) print i" = "t[i] }' /etc/passwd** → compter le nombre d’occurence d’un champ, grace aux tableaux associatifs.

**GREP**

**$grep "^$USER" /etc/passwd**

→ affiche la ligne qui commence par le nom de l’utilisateur courant.

**$ grep -c "bin" /etc/passwd**

→ affiche le nombre des ligne contenant ‘bin’.