Système d'exploitation

open Source

Linux

I- Définition de système d'exploitation

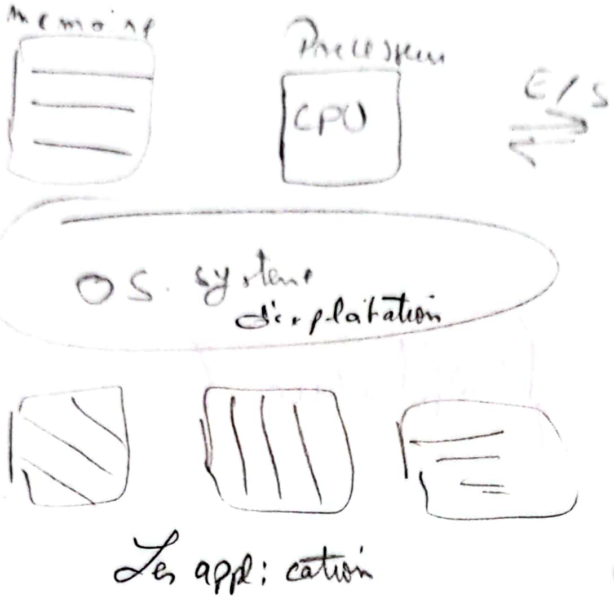
C'est un ensemble de programmes exécuté par le processeur Centrale d'une machine informatique, assurant la gestion de l'ordinateur et des périphériques.

* Il est l'interface entre le Matériel (Hardware) et le logiciel (software)

* Le système d'exploitation permet de gérer :
  - La mémoire
  - Les accès aux Périphérique
  - Le processeur
  - Contrôler Fiabilité (موثوقية) du système
  - La sécurité

Système d'exploitation.

Définition



CPU: Central Processing Unit

RAM: Random Access Memory (Binaire)

HDD: Hard Disk Drive (Disque Dur)

GPU: Graphics Processing Unit (Carte graphique)

Système Informatique

Matérielle

Les Périphériques

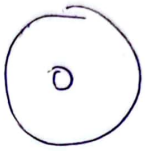
E/S

↑ ↑

Imprimante.

Hardware

Logicielle



Software

La Programmation Shell

pour exécuter un code :

touch script1.sh | => pour exécuté cette code
gedit script1.sh | ./script1.sh (si on a pas l'accès chmod 777 script1.sh)

mon premier code :

#/bin/sh (déclaration d'un script)

echo "bonjour, donnez votre age : " (printf)

read age. (scanf)

echo " votre age est : $age"

on modifie le code pour afficher l'age mais après 10 ans exp si on donne 23 on affiche 33 ans.

read age

a = $ age (valeur de age)

affectation

a = 'expr $age +10'

echo " votre age après 10 ans est : $a ans "

Les Arguments (paramétre)

./script1.sh Argument => votre age est : Argument.

echo "votre age est : $1" ← le 1ère argument

echo " le total est : $# " ← affiche combien d'argument qui donne l'utilisateur.

Les Opérations + Arguments

#/bin/sh

r1 = 'expr $1 + $2'
r2 = 'expr $1 - $2'

r4 = 'expr $1 \/ $2'
r5 = 'expr $1 \% $2'
echo "La somme est : $r1"
echo "La différence est $r2"

./ script.sh 2 3

Résultat

La somme est 5
la différence est -1
le produit est 6

:

On veut calculer la longueur d'une chaine de caractère

```
#/bin/sh
r='expr length $1'
echo $r
```

./script.sh Sirine → 6.

Les tests

if [exp.]

if [$a -eq $b]

[m -eq m] => vrai si m = m

[m -ne m] => vrai si m ≠ m

[m -ge m] => vrai si m ≥ m

[m -gt m] => vrai si m > m

[m -le m] => vrai si m ≤ m

(Les espaces sont obligatoires !!)

[m -lt m] => vrai si m < m

[-f fichier] => vrai si fichier est un fichier.

[-d fichier] => vrai si fich est un rép

[-r fich] => vrai si fich autorisé en lecture

[-w fich] => vrai en écriture

[-x fich] => vrai exécution

[-s fich] => vrai si fich de taille non nulle.

[-z chaine] => vrai si chaine est vide.

[-n chaine] => vrai chaine est non vide.

```
if [ $1 -ge $2 ] then
  echo "$1 > $2"
else
  echo "$2 > $1"
fi
```

```
#/bin/sh
if [ $1 -ge 0 ] then
  r='expr $1 \% 2'
  if [ $r -eq 0 ] then
    echo "le nombre est paire"
  else
    echo "le nombre impaire"
  fi
else
  echo "le nombre est négatif."
fi
```

L'instruction case.

```
case Valeur in
expression 1) commande
              autre commande ;;
expression 2) commande.
              autre commande ;;
    :
*) commande.
   :: autre commande ;;
esac
```

exple on veut créer un prog shell qui donner en parmetre 3 argument 1: nom de fich
2: u,g,0
3: le droit d'accès

et le prog modifie l'accès du fichier selon le besoin de l'utilisateur:

```
#/bin/sh
case $2 in
u) chmod u$3 $1 ;;
g) chmod g$3 $1 ;;
0) chmod 0$3 $1 ;;
*) echo "erreur" ;;
esac
```

→ ./script1.sh bbb u +rwx

Les Boucles

Boucle for.

(1)
```
for i in 1 2 3 4 5
do
 echo $i
done
```

(2)
```
for i in 'seq 1 5'
do
 touch fich.$i
done
```
1 → 5 intervalle

(3)
```
for i in $*
do
echo $i
done
```
$* les arguments

(4)
```
for i in 'ls'
do
 cp $i ~/Bureau
done
```
ajoute un fichier

(5)
```
for i in *.txt
do
 cp $i ~/Bureau
done
```
les éléments de rep
L: A7 ma Fich
chaine de caractère quelconque

exple (1)

```
for i in 1 2 3 4 5 6 7 8 9 10
do
echo $i
affiche 1
        :
        10
done
```

exple (3)

```
for i in $*
do
echo $i;
done.
```

(affiche la valeur de i)

→ ./sujet.sh 1 2 3 4 5

```
affiche
  1
  2
  3
  4
  5
```

→ ./sujet.sh A B C D

```
affiche
  A
  B
  C
  D
for i in $*
do
echo $i;
mkdir $i;
done.
```

./sujet1.sh A B C D

→ Affiche 4 dossier

A
B
C
D

et crée dossier correspondant. A et B et C et D

```
for i in 'seq 1 10'
do
  echo $i;
done.
```

affiche 1 : 10

Exple 4

```
for i in 'ls'
do
 echo $i;
 done.
```

→ ./sujet1.sh

Affiche tout les element de cette bureau

```
for i in 'ls'
do
chmod 000 $i;
done.
```

Exple 5

```
j=1
for i in $*
do
mkdir A/rep$j.$i
j='expr $j + 1'
done..
```

→ ./sujet1.sh ccccc Rtmp Rtmp

j s Rtmp

tree A (affiche l'arborescence de A)

```
for i in *.c
do
 chmod 700 A/$i
done
```

```
#/bin/sh
cd A
for i in *.c
do
 chmod 700 $i
 done.
```

```
cdA
for i in rep* (Rep ... ?)
do
 chmod u-r $i
 done.
```

```
cdA
for i in *.html
do
 rm -r $i
 done.
```

Boucle while

(1)

```
while read ligne.
do                  (parcourir ligne par ligne)
   echo $ligne
done < $fichier
```

(Ecrire le nom du fichier)

```
while [exp]
do
   inst
done
```

(2)

```
until [expression]
do
   inst
done
```

(3)

Exemple 1)

```
i=1
while [ $i -le 10 ]
do
 echo $i
 mkdir A/rep $i
 i=`expr $i + 1`
done.
```

Affiche 1 : 10

calculer le factorielle

```
#!/bin/sh
f=1
i=1
while [ $i -le $1 ]
do
 f=`expr $f \* $i`
 i=`expr $i + 1`
done.
echo "Le factorielle de $1 est : $f"
```

until ≠ while le ≠ gt

```
#!/bin/sh
i=1
until [ $i -gt 10 ]
do
echo $i
i='expr $i + 1'
done.
```

affiche
1
:
10

exple 1 — Affiche les ligne qui contient le nombre de caract sup à 3

```
#!/bin/sh.
while read ligne.
do
v='echo $ligne | wc -c'
3 car de saut de ligne est un car ici
if [ $v -gt 3 ] then
echo $ligne
fi
done < $1.
```

parcourir le fichier pour chercher dans chaque lignes: "abdo" dans la ligne afficher. "qui ligne" si on on affichera "Non ligne"

```
#!/bin/sh
while read ligne
do
v='echo ligne | grep abdo'
case $v in
$ligne)
echo "qui ligne" ;;
*)
echo "non ligne" ;;
esac
done < $1
```

Les fonctions

```
ma fonction ()
{
i=1
f=1
while [ $i -le $1 ]
do
f='expr $f \* $i'
i='expr $i + 1'
done
echo 'la factoriel de $1 est: $f'
}
ma fonction $1
```

sed: Ajouter une ligne au debut de la liste

>>: Ajouter à la fin de la liste

```
echo "texte" >>
cat
```

Les Commandes

CD : Change Directory

Permet de naviguer dans les repertoires

cd : permet de revenir au repertoire personnel (exple : /home/users)

cd.. => Permet de remonter au répertoire parent

cd /usr/lib => on se place dans le repertoire /usr/lib.

cd image => on se place dans le dossier image.

« Pwd »

Permet d'afficher le chemin absolu du repertoire.

Mkdir : Make directory

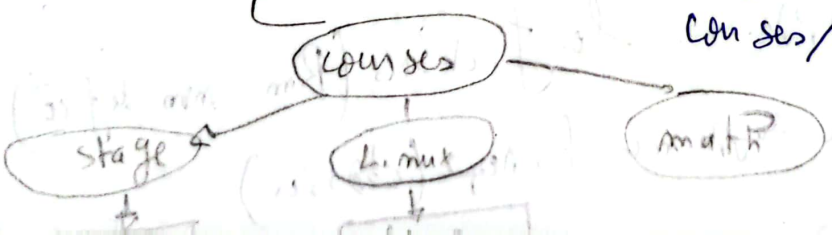
Création d'un répertoire et des dossiers

exple : Mkdir Music photo... (plusieurs dossiers)

P { cd music
  { Mkdir jazz (creer jaz dans Music)

* Mkdir -p => Permet de creer des arborescences

Exple mkdir -p courses/stage/docs courses/linux/devers courses/math



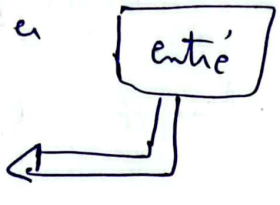
Les Commandes de Création des fichiers.

1 - La Commande « touch » : Permet de Créer un fichier vide.

expl touch fichier.
touch fichier1 fichier2

2) La Commande « cat␣> » : permet de Créer un fichier et écrire

exple cat␣> fichier



bonjour.

Ctrl + D = pour fermer et enregistrer le fichier.

La Commande « LS » LISTER.

ls : Permet d'afficher le contenu du répertoire.

Ex : ls le nom du rép.

ls␣-l : permet d'afficher le détails du contenue de rép.

ls -a : Affiche les dossiers caché

La Commande est aussi : Remove = Supprimer.

rm : Permet de supprimer les fichiers (Rm nom de fic)

rm␣-r : Permet de supprimer les rép (dossier)

- rm -rf : Permet de supprimer les rép. et les sous rép.

La Commande « Mv » : Move.

Mv : Permet de déplacer des fichiers du répertoire, et il permet aussi de renommer un fichier.

Exple : mv linux Cours ⇒ déplacer linux en Cours

La Commande "Cp" COPIER.

Cp : Permet de copier les fichiers.

cp -r : Permet de copier le répertoire et le sous rép.

cp -a : Permet de copier et conserver la date, les droits, le propriétaire.

La Commande « Wc »

Wc : Permet de calculer le nombre des ligne, des mots, et de taille.

wc -l nom de fichier ⇒ affiche le nombre des lignes.

Wc -w nom de fichier ⇒ affiche le nombre des mots

Wc -c nomb de fich ⇒ affiche la taille du fich. (ko)

Les Commandes « Head » et « tail » et « grep »

Head -n : affiche les n premier ligne d'un fichier

tail -n : affiche les n dernier ligne d'un fichier

Grep : Permet de Chercher dans un fichier.

Grep -c : Compte le nombre de ligne contenant la chaine.

Grep -i : ne fait pas la différence entre Maj et Min.

Grep -n : Permet de numéroté les lignes. (affiche les ligne contenant la chaine)

Grep -v : affiche les lignes ne contenant pas la chaine.

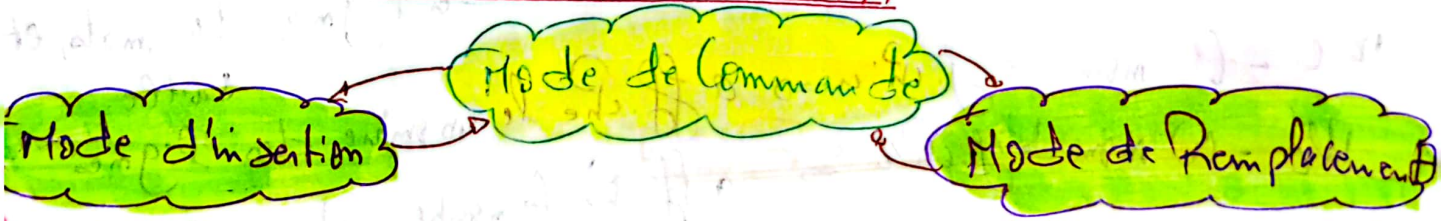
(exple grep␣-c␣ chaine(branche)␣Nom de fichier(Mis:c))

La Commande « VI » EDITEUR.

Vi : permet d'éditer et creer les fichiers

Ex Vi fichier.

Les mode de « Vi »



I : insérer à gauche de la position courant

I : insérer au début de la ligne courant

i insérer à droit de la position courant.

A insérer à droit de dernier caractère de ligne courant.

R : pour Remplacé un mot ou plus à un autre Mot(s)

r : pour remplacer le caract pointé par le curseur

- insère une nouvelle ligne au dessous de la ligne courante.

Mode Commande

Recherche
- /
- ?

Sauvegarde ou quitter

:w => sauvegarde le fichier.

:q => sortie sans sauvegarde la modification depuis la dernière sauvegarde.

:wq => sauvegarde les modification et sortie.

NB :

! => forcer la commande.

exple : q! quitter la modification sans afficher le message de vérification

La Commande « LN » LINK (fichier dis nécessaire)

Ln : Créé un lien physique vers un fichier.

Ln -s : Créé un lien symbolique (similaire au raccourci du monde windows)

Exple ln -s rep1/fichier fichier_racc

La Commande « Find »

find : Permet de chercher des fichier et répertoire.

- -name : Recherche d'un fichier par son nom.
- -type : par type
- -atime : Par date de dernier accès (3)

- mtime : par time de dernière modification
- link : par nombre de lien de fichier.
- user : Recherche de fichier appartenant à l'utilisateur donné
- group : Recherche de fichier appartenant au groupe donné
- newer X : (les fichiers modifiés plus récemment que X)

La Commande « Cat »

Cat : affiche le contenue d'un fichier.

setkmap ω fr

Cat -n : affiche les numéros de ligne.

Exple : cat -n file ⇒ affiche file en numérotant les lignes à partir de 1.

La Commande « More »

More : Affiche un fichier page par page.

Exple : more fichier ⇒ Affiche le fichier page par page

cat /etc/passwd >> file10

cat file10 | more

La Commande « Paste »

Permet de concaténé deux fichiers.

La Commande « tac »

Affiche le contenue d'un fichier dans l'ordre inverse

La Commande « mand »

permet de modifier dans un fichier (exp : ajouté un groupe)

La Commande « uniq »

uniq -c : nombre d'instance de chaque valeur

Les caractères génériques = (Spéciaux)

'*' : 0 ou plusieurs

exple : ls f* : lister sauf que les rep et les fichiers commençant avec le caract f

'?' : un caractère.

exple ls fich? = lister sauf que les fich qui nomment par ex (fich1, fich2 ...)

'[...]' : il représente un intervalle de caract indique entre les crochets

expl ls [a-e]* = lister tout les rep et les fichiers commençant avec le caract (a, b, c, d, e)

'[^...]' : il représente tout les caractères sauf que les caract indiqué entre les crochets.

Exple : ls [^b]* = lister tout les rep et les fich sauf qu'ils contient <<b>>

'(^)' permet de l'afficher tout les ligne qui débutant au caractère indiq

expl : grep ^d fich1 = afficher tout les ligne débutant au <

'($)' : permet d'afficher tout les ligne finissant au caract indique.

exple grep h$ fich1 = afficher tout les ligne finissant par

'(> : supérieur)' rédirection de sortie standard

cat > fich1

(< : infichier) Redirection de l'entré standard

exple : Cat < fichier1. (prend fichiers et met dans cat)
(affiche le contenue de fichier1)

(>> : ecrase).

exple : Cat fich1 >> Docs1 => écrasé fich1 dans Docs1.

(| : pipe) : Exple : Head -4 fich1 | Wc -w => (execute les deux commande)

Compte le nombre des mots dans les 4 premières lignes du fichier (fich1)

Head -5 file1 | tail -1 (affiche la derinier ligne dans les 5 premier ligne)

Uniq -d : repeated

Uniq -D : print all duplicated lines.

Uniq -i : ignore case

Uniq -u : unique (only print unique line

tr " " ":" Remplacer les espaces par :

> File : enregistrer dans File

La Commande << Cut >>

permet de selectionné un champ dans une fichier.

Cut -d " " -f 1,2 : cut -c : c'est nombre de caractère a selec, cut -3 - fin caractère jusqu'à la a partir de 3 caractère

délimiteur — champ 1 et 2

La Commande << tee >>

Enregistrer.

La Commande tr

tr ',' '.' : remplace les virgule par les point virgule

tr -s : supp un caractère qui se répetent

Les Permissions (Droits) d'accès

Lecture = r , Ecriture = w , Exécution = x

La commande « Chmod »

Chmod : modifie les permission d'accès d'un fichier ou répertoire.

* la manière d'indique les perm de fichier ou rép.

1. Mode numérique.

NB : Les perm sont des nbre.

Perm	La v. num
r	4
w	2
x	1

Exple donner les droits suivant : au fichier 1 : - Utilisateur : lecture + écriture (4+2) ; - groupe : lecture ; - autre : lecture.

Chmod 644 fichier 1.

NB : La valeur max d'un fichier 666
La valeur max d'un rép 777

2 - Mode Symbolique

Qui ? , Opération , Permission.

a) Qui ?

* u (user) le propriétaire.
* g (group) les membres du group.
* o (others)
* a (all) : le propriétaire, Groupe, autres

b) Opérateur :

* (+) ajouter
* (-) Supprimer.
* (=) fixer

c) Permission.

* r
* w
* x

exple donnez les droits suivants au fich1 : User : r-x, group : r--, others ---

=> chmod u=rx, g=r fich1.

chmod u=rwx, g-w, o-x Conses.

Gestion des Utilisateurs

Le fichier contenant les informations des Utilisateurs c'est : /etc/passwd

Ce fichier comporte 7 champs dans chaque ligne séparé par un ":"

nom user : mot de passe : uid : gid : information : dossier personnel : shell (par défaut bash)

Les programme d'un démarrage.

NB : uid (0-99) réservé aux comptes propres à la machine

Les commandes de gestion

* Useradd / adduser : Permet de créer un utilisateur

useradd (-u) (-g) (-c) (-d) (-s) nom d'utilisateur

* Userdel : Permet de supprimer l'utilisateur
* Userdel -r : Permet de supprimer l'util et le rep personnel
* Usermod : Permet de modifier les infos d'un utilis
* Passwd : Permet de sécuriser un utilis.

Exple :

useradd -u 1200 -g 500 -c ELyazid: -d /home/hamza HAMZA

→ Vi /etc/passwd

Affichage => HAMZA : x : 1200 : 500 : ELyazid : /home/hamza : /bin/bash /-s par défaut/

NB : Utilisable par root (admin)

pour accéder en mode root on écrit su et le mot de passe

exit pour retour $

Exple 2 (pr modifier)

Usermod -u 1000 HAMZA

V: /etc/passwd

(Pour supp HAMZA)

Userdel -r HAMZA

passwd HAMZA

Gestion Des Groupes

Le fichier contient les information des groupes c'est le fichier: /etc/group

Chaque ligne identifie un groupe Crée, et contient 4 champ séparé par "::"

nom de groupe : mot de passe : gid : Les membres (membre 1, membre 2 ...)

* V: /etc/group.
* Les commandes de Gestion

NB: utilisable par root

Groupadd : Creer un groupe.

Groupadd -g (gid) nom de groupe

Groupdel : Supprime un groupe.

Groupmod : modifier les infos d'un groupe.

(Groupmod -g modifie id groupe

Groupmod -p modifie le mot de passe.

Groupmod -n ancien nom nouveau nom modifie le nom d'un groupe.

Exple Groupmod -g 660 TRI. | Groupmod -p hi123 ITHC

Groupmod -n TRI ITHC

6- Commandes « Chown » et « CH GRP »

(utilisable par mode Root)

Chown : permet de modifier le propriétaire et le groupe d'un fichier ou d'un répertoire.

Chown (nouveau propriétaire) (nom de fichier) : Changer le propriétaire d'un fichier ou répertoire

Ch grp : permet de modifier le groupe d'un fichier ou d'un rép.

Chgrp (nouveau groupe) (nom de fich) : Changer le groupe d'un fich ou d'un rép.

Chown (nouveau propriétaire) : (nouveau groupe) (nom de fich ou rép)

NB ls -l ⇔ ll

0 , 1 , 2

entrée sortie erreur

- print 2 <

Les Commandes de Système

Who : Permet d'afficher qui est connecté

Date : permet d'afficher la date

Id : permet d'afficher les infos liées à l'utilisateur ou d'un groupe

Reboot : permet de redémarrer le système d'exp.

Halt : permet l'arrêt du système.

Uname : permet d'afficher les information sur le système (Nom, Process...)

Cal : affiche le Calendrier.

Su : permet de changer l'utilisateur Normal à l'administrateur

NB pour afficher les options d'une commandes :

Nom Commande --help

clear.

Commande Sort

Sort -r : tri b

inclus ↓ non inclus ↓

Sort + debut n -fin n Nom fichier ;

exple : 0 1 2

	Ali	32 kg	1m 91
	Med	62 kg	1m 65
	Sonia	90 kg	1m 54

tri selon poids

sort +1n.0 -1n.2

commence par le caractère → se termine par ce

tri selon Prénom

sort +2n.2 -3n.0