Manipulation de l'arborescence d'un système Linux

1. L'arborescence des fichiers sous Linux.

Dans les systèmes de type GNU/Linux, toute l'information stockée dans vos médias de stockage (disques durs, clé USB, cartes SD, CD-ROM, etc.) est nécessairement accessible en suivant un chemin depuis un emplacement logique appelée la racine (notée /). (Norme FHS : Monde libre)

Répertoire	Contenu
/	Racine du système, hiérarchie primaire
/bin	Exécutables des commandes essentielles
/boot	Fichiers statiques du chargeur d'amorçage
/dev	Fichiers spéciaux des périphériques
/etc	Fichiers textes de configuration.
/home	Répertoires personnels des utilisateurs
/lib	Bibliothèques partagées essentielles et modules du noyau
/media	Contient les points de montages pour les médias amovibles
/mnt	Point de montage pour monter temporairement un système de fichiers
/proc	Répertoire virtuel pour les informations système (états du noyau et des processus système)
/root	Répertoire personnel du super-utilisateur
/sbin	Exécutables système essentiels
/srv	Données pour les services du système
/tmp	Fichiers temporaires
/usr	Hiérarchie secondaire, pour des données en lecture seule par les utilisateurs. Ce répertoire contient les applications usuelles des utilisateurs et leurs fichiers.
/var	Données variables et diverses.
/opt	Emplacement pour des applications installées hors gestionnaire de paquets

Tous ces répertoires sont appelés répertoires primaires du système : ils sont directement à la racine.
Tous les éléments stockés dans le système, qu'il s'agisse de fichiers ou de dossiers, sont appelées des noeuds.

2. Manipulation des noeuds.

Déplacement :

```
$cd <répertoire à atteindre>
```

```
$cd /       #accès au répertoire racine.
$cd /etc    #accès au répertoire primaire etc.
$cd /home   #accès au dossier contenant le répertoire personnel.
```

Répertoires spéciaux:

```
$cd ..      #accès au dossier parent du dossier courant
$cd ~       #accès direct au répertoire personnel de l'utilisateur
courant
```

Chemin relatif:

Permet d'accéder à un nœud en partant du répertoire courant.

```
$cd etc           #accès au sous dossier etc, s'il existe dans le
dossier courant.
$cd ../test       #accès au sous dossier test, situé dans le dossier
parent du dossier courant
```

Manipulation des répertoires :

Pour lister les fichiers et dossiers dans le dossier courant :

```
$ls         #Afficher les fichier et dossier du répertoire courant.
$ls -l      #Afficher les informations de manière détaillée.
$ls -a      #Afficher les fichiers cachés.
$ls -h      #Afficher la taille des fichiers de façon lisible.
$ls -la     #Afficher tous les fichiers y compris les fichiers cachés.
$ls -R      #Afficher tout le nœud du répertoire courant de façon
récursive.
```

Création	`$mkdir -p <nomDossierACréer>   #[-p] crée l'arborescence`
Renommage	`$mv <ancienNomDossier> <nouveauNomDossier>`
Suppression	`$rm -r <nomDossier>`
Déplacement	`$mv <ancienNomDossier> <nouveauNomDossier>`
Copie	`$cp -r <nomDossierOrigine> <nomDossierCopie>`

3. Manipulation des fichiers.

Les commandes de manipulation des fichiers s'exécutent toujours dans la limite des droits autorisés pour l'utilisateur courant.

Création	`#Utiliser un éditeur de texte, comme nano, vi, ou autre`
Renommage	`$mv <ancienNomFichier> <nouveauNomFichier>`
Suppression	`$rm <nomFichier>`
Déplacement	`$mv <ancienNomFichier> <nouveauNomFichier>`
Copie	`$cp <nomFichierOrigine> <nomFichierCopie>`

Affichage du contenue d'un fichier :

```
$cat <fichierÀAfficher>  #cat affiche la fin du fichier. (shift + ? et ? pour naviguer)
$more <fichierÀAfficher>      #q pour quitter
$less <fichierÀAfficher>      #↑ ↓ pour naviguer et q pour quitter
```

Afficher les premières lignes d'un fichier :

```
$head [-n <nbLignes>] <fichierÀAfficher>
```

L'option [-n <nbLignes>] permet de spécifier le nombre de lignes à afficher.
Par défaut, ce sont les 10 premières lignes du fichier qui sont affichées.

Afficher les dernières lignes d'un fichier :

```
$tail [-n <nbLignes>] [-f] <fichierÀAfficher>   #[-f] permet un affichage dynamique
```

4. Effectuer des recherches dans le système de fichier.

find est un outil de recherche qui permet de retrouver un fichier partout dans le système d'après son nom. Il peut aussi filtrer les résultats d'après la date de création ou les droits sur ces fichiers.

```
$find <dossierDeRecherche> [option]
```

```
$find <dossierDeRecherche> -name <chaineRecherche>
#recherche la chaîne dont le nom est précisé.

$find <dossierDeRecherche> -iname <chaineRecherche>
#recherche la chaîne dont le nom est précisé sans tenir compte de la
casse.
```

*A noter que le caractère générique * est utilisé pour remplacer n'importe quelle chaîne.*

Rechercher des fichiers contenant un texte précis :

Un autre outil de recherche, grep, permet de filtrer la recherche non sur le nom du fichier, mais sur ce qu'il contient.

```
$grep [options] '<dossierDeRecherche>' -e '<expressionARechercher>'
```

[options]

-r indique une recherche récursive, qui cherche dans tous les sous-dossiers jusqu'à avoir tout examiné.

-n affiche le numéro de la ligne correspondant à la recherche.

-l retourne uniquement le nom du fichier dans lequel figure la ligne trouvée.

-i rend la commande insensible à la casse.

--include='<chaineRecherche>' limite la recherche aux fichiers correspondant au critère

--exclude='<chaineRecherche>' limite la recherche aux fichiers ne correspondant pas au critère

```
#grep -rni '/etc' -e 'port 22'
```

permet de rechercher les fichiers de configuration qui contiennent la chaîne port 22, quelle que soit sa casse.

5. Manipuler les fichiers depuis le shell.

La commande touch :

```
$touch <nomFichier>     #créé le fichier si inexistant et modifie sa date de création.
```

Edition des fichiers depuis le shell :
La commande echo écrit dans le prompteur ou dans un fichier en précisant un redirecteur de flux.

```
$echo "<texte>"   #écrit à l'écran
```

```
$echo "<texte>" > <nomFichier>
#L'opérateur > crée le fichier indiqué. S'il existe, il est effacé.

$echo "<texte>" >> <nomFichier>
#L'opérateur >> crée le fichier indiqué. S'il existe, il est complété à la fin.
```

6. Transfert de fichier par ssh.

La commande scp :
Copie d'un fichier du poste local vers poste distant :

```
$scp <nomFichierLocal> <user>@<IPMachineDistante>:<dossierCible>
```

On ajoute l'option -r pour un répertoire.

Copie d'un fichier du poste local vers poste distant :

```
$scp <user>@<IPMachineDistante>:<dossier/fichierCible> <nomFichierLocal>
```

7. Téléchargement de fichiers depuis des serveurs web.

La commande wget :

```
$wget <adresse URL complète du fichier>
```

8. Gestion des archives.

La commande zip:

Création d'archive :

```
$zip -r <nomArchive> <DossierÀArchiver>
```

Décompression:

```
$unzip <nomArchive> [-d <DossierDestinationArchive>]
```

La commande tar:

Création d'archive :

```
$tar cvzf <nomArchive> <DossierÀArchiver>
```

description des option:

c	compress
v	verbose
z	zippe l'archive avec le programme gzip (le nom du fichier devra se terminer par .tar.gz)
f	force l'inclusion de fichiers en cas de doublons.
x	extract

Décompression:

```
$tar xvzf <nomArchive> [-C <DossierDestinationArchive>]
```