

TP — Manipuler fichiers et répertoires

Création, copie, déplacement, suppression, redirections

🎯 Objectifs de ce chapitre

- Créer, copier, déplacer, renommer et supprimer des fichiers et des répertoires en ligne de commande.
- Afficher un fichier et y chercher une information avec `cat`, `less`, `head` et `grep`.
- Utiliser les redirections `>` et `>>`.
- Comprendre que, sous Linux, « tout est fichier ».

Avant de commencer

Dans ce TP, vous allez créer et supprimer des fichiers : travaillez donc toujours dans un répertoire réservé à cet usage, pour ne rien abîmer d'autre. La méthode est toujours la même : **essaie, observe, déduis**. Avant de valider une commande, essaie de prévoir ce qu'elle va faire ; après, comparez avec ce que vous observez.

🛠 Préparation

1. Placez-vous dans votre répertoire personnel avec la commande `cd`.
2. Créez un répertoire `TP_OS` avec `mkdir TP_OS`, puis placez-vous dedans avec `cd TP_OS`.
3. Vérifiez avec `pwd` que vous y êtes bien. Chemin absolu de `TP_OS` :

Tous les exercices de ce TP se font dans `TP_OS`. Si tu ne sais plus où tu es, tape `pwd`.

Exercice 1 — Créer, copier, déplacer, supprimer

Objectif : maîtriser les commandes de base de gestion des fichiers et des répertoires.

Commande	Effet	Exemple
<code>mkdir nom</code>	Crée un répertoire (<i>MaKe DiRectory</i>).	<code>mkdir test</code>
<code>touch nom</code>	Crée un fichier vide.	<code>touch test.txt</code>
<code>cp source destination</code>	Copie un fichier.	<code>cp f1.txt f2.txt</code>
<code>cp -r source destination</code>	Copie un répertoire et tout son contenu.	<code>cp -r avant apres</code>
<code>mv source destination</code>	Déplace ou renomme un fichier ou un répertoire (<i>MoVe</i>).	<code>mv f1.txt f2.txt</code>
<code>rm fichier</code>	Supprime un fichier (<i>ReMove</i>).	<code>rm test.txt</code>
<code>rmdir répertoire</code>	Supprime un répertoire vide .	<code>rmdir test</code>
<code>rm -r répertoire</code>	Supprime un répertoire et tout son contenu.	<code>rm -r test</code>

⚠ Pas de corbeille

`rm` supprime **définitivement** et sans demander confirmation, et `rm -r` supprime tout un répertoire. Relisez votre commande avant d'appuyer sur Entrée, et vérifiez toujours avec `pwd` et `ls` où vous vous trouvez.

Manipulations

1. À l'aide de `mkdir` et `touch`, créez l'arborescence suivante dans `TP_OS` :

```
TP_OS/
|-- dir1/
|   |-- fichier1
|   |-- .fichier2    (fichier caché : son nom commence par un point)
|-- dir2/
|   |-- fichier3
```

Astuce : on peut créer un fichier dans un autre répertoire sans s'y déplacer, grâce à un chemin relatif : `touch dir1/fichier1`.

2. Placez-vous dans `dir1` et tapez `ls`. Que remarquez-vous ? Cherchez dans `man ls` l'option qui affiche aussi les fichiers cachés (q pour quitter le manuel).

3. Depuis `dir1`, écrivez le chemin relatif de `fichier3`. Vérifiez-le avec `ls`.

4. Tapez `cd ..`, puis écrivez à nouveau le chemin relatif de `fichier3`.

5. Copiez `fichier3` dans `dir2` sous le nom `fichier4`. Vérifiez avec `ls dir2`.

6. Avec `mv`, renommez `fichier4` en `copie`. Que contient `dir2` maintenant ?

7. Tapez `cp dir2 dir3`. Que se passe-t-il ? Corrigez la commande pour copier le répertoire et son contenu.

8. Tapez `rmdir dir1`. Que se passe-t-il ? Consultez `man rmdir` pour expliquer.

9. **Avant** de la taper, prédissez ce que va faire la commande `mv dir1/* dir2/` : quels fichiers vont être déplacés ? Tapez-la, puis observez avec `ls -a dir1` et `ls -a dir2`. Votre prédiction était-elle juste ? Déplacez ensuite ce qui reste dans `dir1` vers `dir2`.

10. Supprimez `dir1` (devenu vide) avec `rmdir`, puis `dir3` et son contenu avec `rm -r`. Vérifiez avec `ls`.

Ce que vous avez découvert

- `cp` ne copie pas un répertoire sans l'option `-r`.
- `rmdir` ne supprime qu'un répertoire **vide** ; `rm -r` supprime un répertoire et tout son contenu.
- `mv` sert à déplacer **et** à renommer.
- Le joker `*` remplace n'importe quelle suite de caractères, **sauf** au début d'un nom commençant par un point : les fichiers cachés ne sont pas concernés.

Exercice 2 — Les redirections

Objectif : comprendre où va le texte affiché par une commande, et la différence entre `>` et `>>`.

Commande	Effet
<code>echo texte</code>	Affiche le texte à l'écran.
<code>cat fichier</code>	Affiche le contenu d'un fichier.
<code>commande > fichier</code>	Envoie le résultat de la commande dans le fichier, au lieu de l'écran.

 Essayez, observez

1. Tapez `echo 'Longtemps, je me suis '`. Que se passe-t-il ?
2. Tapez `echo 'Longtemps, je me suis ' > debut.txt`. Que se passe-t-il à l'écran ? Où est passé le texte ?
3. Affichez le contenu du fichier avec `cat debut.txt`.
4. Tapez `echo 'couché de bonne heure' >> debut.txt`, puis `cat debut.txt`. Qu'est devenu le fichier ?
5. **Prédisez** ce que contiendra `debut.txt` après `echo 'autre chose' > debut.txt`, puis vérifiez avec `cat`.

 À retenir

- `>` **écrase** le contenu du fichier (ou le crée s'il n'existe pas).
- `>>` **ajoute** à la fin du fichier, sans effacer ce qu'il contient.

Exercice 3 — Tout est fichier

Objectif : découvrir que, sous Linux, même les périphériques sont représentés par des fichiers.

Chaque terminal ouvert correspond à un **fichier spécial**, situé dans le répertoire `/dev/pts/` (par exemple `/dev/pts/0`). Ce n'est pas un fichier texte classique : **écrire dedans revient à envoyer du texte sur le terminal correspondant**.

 Essaie, observe

1. Ouvrez **deux terminaux** : `term1` et `term2`.
2. Dans `term1`, tape `tty`. Notez le chemin affiché (par exemple `/dev/pts/0`) :
3. Dans `term2`, tape `echo 'Bonjour' > chemin_noté_à_1_étape_2`. Que se passe-t-il dans `term1` ?
4. Quelle commande faut-il taper dans `term1` pour afficher « Bien et vous » dans `term2` ? (Indice : tapez d'abord `tty` dans `term2`.)
5. *Pour aller plus loin* : tape `head -n 5 /proc/cpuinfo` puis `head -n 3 /proc/meminfo`. Ces fichiers n'existent pas sur le disque : le noyau les fabrique à la demande. Que contiennent-ils ?

Vous venez d'utiliser une redirection vers un terminal, exactement comme vers un fichier sur le disque : c'est ce que veut dire « tout est fichier ».

Exercice 4 — Lire un fichier et y chercher une information

Objectif : afficher un fichier et retrouver rapidement ce qui nous intéresse avec `grep`.

Nous utiliserons `/etc/passwd`, un fichier lisible par tous qui liste les comptes utilisateurs du système, à raison d'une ligne par compte. Chaque ligne est composée de champs séparés par `:` : le premier est le nom du compte, les deux derniers sont le répertoire personnel et le programme lancé à la connexion (le shell).

Commande	Effet
<code>cat fichier</code>	Affiche tout le contenu du fichier.
<code>less fichier</code>	Affiche le fichier page par page (flèches pour défiler, <code>q</code> pour quitter).
<code>head -n 5 fichier</code>	Affiche les 5 premières lignes.
<code>wc -l fichier</code>	Compte les lignes du fichier (<i>Word Count</i>).

Commande	Effet
<code>grep motif fichier</code>	Affiche les lignes qui contiennent le motif.

Option de grep	Effet
<code>-i</code>	Ignore la différence majuscules / minuscules.
<code>-n</code>	Affiche le numéro de chaque ligne trouvée.
<code>-c</code>	Compte le nombre de lignes trouvées, au lieu de les afficher.

Essaie, observe

1. Tapez `less /etc/passwd`, parcourez le fichier avec les flèches, puis quittez avec `q`.
2. Tapez `head -n 3 /etc/passwd`. Quel est le premier compte ?
3. Tapez `wc -l /etc/passwd`. Combien y a-t-il de lignes, donc de comptes ?
4. Tapez `grep "root" /etc/passwd`. Quelles lignes sont affichées ?
5. Comparez `grep "ROOT" /etc/passwd` et `grep -i "ROOT" /etc/passwd`. Que remarquez-vous ?
6. Tapez `grep -n "bash" /etc/passwd`, puis `grep -c "bash" /etc/passwd`. Quels comptes utilisent le shell Bash ? Combien sont-ils ?
7. Tapez `grep "$USER" /etc/passwd` pour retrouver ta propre ligne. Votre répertoire personnel est-il celui que vous connaissez ?

Si votre ligne n'apparaît pas, c'est que votre compte est géré par le réseau : demandez à votre professeur de vous l'expliquer.

Pour aller plus loin — le tube

Le **tube** | envoie le résultat d'une commande à la commande suivante. Essayez et expliquez ce que fait chacune de ces lignes :

```
grep "bash" /etc/passwd | wc -l
ls /etc | wc -l
ls /etc | sort -r
```

Bonus — expressions régulières

Pour les élèves à l'aise. Le motif de `grep` peut contenir des symboles spéciaux :

`grep "^root" /etc/passwd` : le symbole `^` désigne le **début** d'une ligne.

`grep "bash$" /etc/passwd` : le symbole `$` désigne la **fin** d'une ligne.

`grep -E "root|daemon" /etc/passwd` : avec l'option `-E`, le symbole `|` signifie « ou » (ce n'est pas un tube, car il est entre guillemets).

Question : quelle commande affiche les lignes qui se terminent par `nologin` ?

Exercice 5 — Mini-défi

Organiser un projet

Dans TP_OS, sans que les commandes vous soient données :

1. Créez un répertoire projet qui contient deux sous-répertoires : src et docs.
2. Dans docs, créez un fichier notes.txt de trois lignes, en utilisant seulement echo, > et >>.
3. Copiez notes.txt dans src sous le nom notes_copie.txt.
4. Renommez notes.txt en lisezmoi.txt.
5. Affichez d'un seul coup tout le contenu de projet, y compris ses sous-répertoires (cherche l'option R de ls).
6. Dessinez l'arborescence obtenue sur ton cahier, puis supprime projet et tout son contenu.

Pour s'amuser (facultatif)

Quelques commandes à tester si tu as terminé. Ctrl+C interrompt une commande qui ne s'arrête pas toute seule.

Commande	Effet
cal	Affiche un calendrier.
date	Affiche la date et l'heure.
echo "4+6" bc -l	Calcule une expression mathématique.
echo "scale=10; 4*a(1)" bc -l	Calcule π avec 10 décimales.
ls sort -r	Liste les fichiers dans l'ordre inverse.
echo 'Un palindrome' rev	Inverse une chaîne de caractères.
echo \$((\$RANDOM % 6 + 1))	Lance un dé : nombre aléatoire entre 1 et 6.
factor 12	Décompose un nombre en facteurs premiers.
yes	Écrit « y » sans fin (arrête avec Ctrl+C).
while true; do echo "\$(date '+%D %T')"; sleep 1; done	Affiche l'heure en direct (arrête avec Ctrl+C). C'est déjà un petit programme en Bash !
curl wttr.in/paris	Affiche la météo de Paris (nécessite un accès au réseau).
firefox &	Lance Firefox en arrière-plan : le terminal reste utilisable. On y reviendra avec les processus.
telnet towel.blinkenlights.nl	Démonstration du professeur : Star Wars en ASCII (pour quitter : Ctrl+] puis quit). Le service est parfois indisponible.