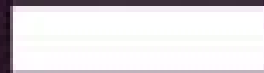


> Projet

# SHELL



Hervé BEZIAT

# SOMMAIRE

<b>JOB 01</b>	<b>4</b>
Afficher le manuel de la commande ls	4
Afficher les fichiers cachés du home de votre utilisateur	4
Afficher les fichiers cachés plus les informations sur les droits sous forme de liste	5
<b>JOB 02</b>	<b>6</b>
Lisez un fichier en utilisant une commande qui permet seulement de lire	6
Afficher les 10 premières lignes du fichier ".bashrc"	6
Afficher les 10 dernières lignes du fichier ".bashrc"	7
Afficher les 20 premières lignes du fichier ".bashrc"	7
Afficher les 20 dernières lignes du fichier ".bashrc"	8
<b>JOB 03</b>	<b>9</b>
Installer le paquet "cmatrix"	9
Lancer le paquet que vous venez d'installer	9
Mettre à jour son gestionnaire de paquets	9
Mettre à jour ses différents logiciels	10
Télécharger les internets : Google	10
Redémarrer votre machine	11
Éteindre votre machine	11
<b>JOB 04</b>	<b>12</b>
Créer un fichier users.txt qui contiendra "User1" et "User2" séparé par un retour à la ligne	12
Créer un groupe appelé "Plateformeurs"	13
Créer un utilisateur appelé "User1"	13
Créer un utilisateur appelé "User2"	13
Ajouter "User2" au groupe Plateformeurs	13
Copier votre "users.txt" dans un fichier "droits.txt"	13
Copier votre "users.txt" dans un fichier "groupes.txt"	14
Changer le propriétaire du fichier "droits.txt" pour mettre "User1"	14
Changer les droits du fichier "droits.txt" pour que "User2" ai accès seulement en lecture	14
Changer les droits du fichier "groupes.txt" pour que les utilisateurs puissent accéder au fichier en lecture uniquement	14
Changer les droits du fichier pour que le groupe "Plateformeurs" puissent y accéder en lecture/écriture.	15
<b>JOB 05</b>	<b>16</b>
Ajouter un alias qui permettra de lancer la commande "ls -la" en tapant "la"	17
Ajouter un alias qui permettra de lancer la commande "apt-get update" en tapant "update"	17

Ajouter un alias qui permettra de lancer la commande "apt-get upgrade" en tapant "upgrade"	18
Ajouter une variable d'environnement qui se nommera "USER" et qui sera égale à votre nom d'utilisateur	18
Mettre à jour les modifications de votre bashrc dans votre shell actuel	18
Afficher les variables d'environnement	18
Ajouter à votre Path le chemin "/home/'votre utilisateur'/Bureau"	18
<b>JOB 06</b>	<b>18</b>
<b>La commande tar permet d'archiver ou de désarchiver des répertoires et des fichiers de façon optimale.</b>	<b>19</b>
<b>JOB 07</b>	<b>19</b>
Créer un fichier "une_commande.txt" avec le texte suivant "Je suis votre fichier texte"	20
Compter le nombre de lignes présentes dans votre fichier de source apt et les enregistrer dans un fichier nommé "nb_lignes.txt"	20
Afficher le contenu du fichier source apt et l'enregistrer dans un autre fichier appelé "save_sources"	20
Faites une recherche des fichiers commençant par "." tout en cherchant le mot alias qui sera utilisé depuis un fichier	20
<b>POUR ALLER PLUS LOIN</b>	<b>22</b>
Installer la commande tree	22
Lancer la commande tree en arrière-plan qui aura pour but d'afficher toute l'arborescence de votre / en enregistrant le résultat dans un fichier "tree.save"	22
Lister les éléments présents dans le dossier courant et utiliser directement le résultat de votre première commande pour compter le nombre d'éléments trouvés	22
Lancer une commande pour update vos paquets, si l'update réussit alors, vous devrez lancer un upgrade de vos paquets. Si l'update échoue, votre upgrade ne se lancera pas	23

# JOB 01

- Afficher le manuel de la commande ls

```
herve@debian:~$ man ls
```

L'option longue **--help** donne le même résultat que la commande **man**

```
NOM
    ls - Afficher le contenu de répertoires

SYNOPSIS
    ls [OPTION]... [FICHIER]...

DESCRIPTION
    Afficher les informations des FICHIERS (du répertoire courant par défaut). Les entrées sont triées alphabétiquement si aucune des options -cftuvSUX ou --sort n'est indiquée.

    Les paramètres obligatoires pour les options de forme longue le sont aussi pour les options de forme courte.

    -a, --all
        inclure les entrées débutant par « . »

    -A, --almost-all
        omettre les fichiers « . » et « .. »

    --author
Manual page ls(1) line 1 (press h for help or q to quit)
```

- Afficher les fichiers cachés du home de votre utilisateur

L'option **-a** ou **--all** va permettre à la commande **ls** de nous afficher tout les dossiers et fichiers du répertoire en cours.

```
herve@debian:~$ ls -a
.      .bashrc  Documents  Modèles   Public
..     Bureau  .gnupg     .mozilla  .ssh
.bash_history  .cache    Images     Musique   Téléchargements
.bash_logout   .config   .local     .profile  Vidéos
herve@debian:~$
```

```
herve@debian:~$ ls --all
```

- **Afficher les fichiers cachés plus les informations sur les droits sous forme de liste**

Pour cette commande nous aurions besoin de 2 options. Une pour afficher tout les fichiers **-a** et une autre pour lister les droits de chacun des fichiers et dossiers **-l**.

```
herve@debian:~$ ls -a -l
total 25644
drwxr-xr-x 17 herve herve          4096 21 sept. 14:47 .
drwxr-xr-x  3 root  root          4096 14 sept. 14:15 ..
-rwxrwxrwx  1 herve herve           90 20 sept. 11:35 .bash_aliases
-rw-r----- 1 herve herve       11550 21 sept. 16:16 .bash_history
-rw-r--r--  1 herve herve         220 14 sept. 14:15 .bash_logout
-rw-r--r--  1 herve herve        3526 14 sept. 14:15 .bashrc

herve@debian:~$ ls --all -l
total 25644
drwxr-xr-x 17 herve herve          4096 21 sept. 14:47 .
drwxr-xr-x  3 root  root          4096 14 sept. 14:15 ..
```

Questions :

### Comment ajouter des options à une commande?

Certaines commandes prennent des paramètres. Les paramètres qui commencent par « - » ou « -- » sont appelés options et contrôlent le comportement de la commande.

### Quelles sont les deux syntaxes principales d'écriture des options pour une commande ?

Presque tous les programmes GNU/Linux obéissent à un ensemble de conventions concernant l'interprétation des arguments de la ligne de commande. Les arguments attendus par un programme sont classés en deux catégories : les options (ou drapeaux (NdT. flags en anglais)) et les autres arguments. Les options modifient le comportement du programme, alors que les autres arguments fournissent des entrées (par exemple, les noms des fichiers d'entrée).

Les options peuvent prendre deux formes:

- Les options courtes sont formées d'un seul tiret et d'un caractère isolé (habituellement une lettre en majuscule ou en minuscule). Elles sont plus rapides à saisir.
- Les options longues sont formées de deux tirets suivis d'un nom composé de lettres majuscules, minuscules et de tirets. Les options longues sont plus faciles à retenir et à lire (dans les scripts shell par exemple).

Généralement, un programme propose une version courte et une version longue pour la plupart des options qu'il prend en charge, la première pour la brièveté et la seconde pour la lisibilité. Par exemple, la plupart des programmes acceptent les options **-h** et **--help** et les traitent de façon identique. Normalement, lorsqu'un programme est invoqué depuis la ligne de commande, les options suivent immédiatement le nom du programme. Certaines options attendent un argument immédiatement à leur suite. Beaucoup de programmes, par exemple, acceptent l'option **--output foo** pour indiquer que les sorties du programme doivent être redirigées vers un fichier appelé **foo**. Après les options, il peut y avoir d'autres arguments de ligne de commande, typiquement les fichiers ou les données d'entrée.

## JOB 02

- Lisez un fichier en utilisant une commande qui permet seulement de lire

La commande **cat** permet de lire un fichier dans le terminal sans pouvoir le modifier.

```
CAT(1)                                Commandes de l'utilisateur                                CAT(1)

NOM
    cat - Concaténer des fichiers et les afficher sur la sortie standard

SYNOPSIS
    cat [OPTION] ... [FICHIER] ...

DESCRIPTION
    Concaténer le ou les FICHIER(s) sur la sortie standard.

    L'entrée standard est lue quand FICHIER est omis ou quand FICHIER vaut
    « - ».

```

```
herve@debian: ~
herve@debian:~$ cat .bashrc
# ~/.bashrc: executed by bash(1) for non-login shells.
# see /usr/share/doc/bash/examples/startup-files (in the package bash-doc)
# for examples

# If not running interactively, don't do anything
case $- in
    *i*) ;;
    *) return;;
esac

# don't put duplicate lines or lines starting with space in the history.
# See bash(1) for more options

```

- Afficher les 10 premières lignes du fichier “.bashrc”

La commande **head** va nous permettre d’afficher les 10 premières lignes d’un fichier.

```
HEAD(1)                                Commandes de l'utilisateur                                HEAD(1)

NOM
    head - Afficher le début des fichiers

SYNOPSIS
    head [OPTION]... [FICHIER]...

DESCRIPTION
    Afficher les 10 premières lignes de chaque FICHIER sur la sortie stan-
    dard. Avec plus d'un FICHIER, faire précéder chacun d'un en-tête don-
    nant le nom du fichier.

    L'entrée standard est lue quand FICHIER est omis ou quand FICHIER vaut

```

```

herve@debian: ~
herve@debian:~$ head .bashrc
# ~/.bashrc: executed by bash(1) for non-login shells.
# see /usr/share/doc/bash/examples/startup-files (in the package bash-doc)
# for examples

# If not running interactively, don't do anything
case $- in
    *i*) ;;
    *) return;;
esac

```

- **Afficher les 10 dernières lignes du fichier “.bashrc”**

La commande **tail** va nous permettre d'afficher les 10 dernières lignes d'un fichier.

```

TAIL(1)                                Commandes de l'utilisateur                                TAIL(1)

NOM

    tail - Afficher la dernière partie de fichiers

SYNOPSIS

    tail [OPTION] ... [FICHIER] ...

DESCRIPTION

    Afficher les 10 dernières lignes de chaque FICHIER sur la sortie stan-
    dard. Lorsqu'il y a plus d'un FICHIER, faire précéder chaque groupe de
    lignes d'un en-tête donnant le nom du fichier.

```

```

herve@debian:~$ tail .bashrc
# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_completion ]; then
    . /usr/share/bash-completion/bash_completion
  elif [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
  fi
fi

```

- **Afficher les 20 premières lignes du fichier “.bashrc”**

```

-n, --lines=[-]N
    afficher les N premières lignes au lieu des 10 premières ; avec
    le préfixe « - », afficher toutes les lignes sauf les N der-
    nières lignes de chaque fichier

```

**head -n 20 .bashrc**

**head -20 .bashrc**

**head --lines 20 .bashrc**

```

herve@debian: ~
herve@debian:~$ head -n 20 .bashrc
# ~/.bashrc: executed by bash(1) for non-login shells.
# see /usr/share/doc/bash/examples/startup-files (in the package bash-doc)
# for examples

# If not running interactively, don't do anything
case $- in
    *i*) ;;
    *) return;;
esac

# don't put duplicate lines or lines starting with space in the history.
# See bash(1) for more options
HISTCONTROL=ignoreboth

# append to the history file, don't overwrite it
shopt -s histappend

# for setting history length see HISTSIZE and HISTFILESIZE in bash(1)
HISTSIZE=1000
HISTFILESIZE=2000
herve@debian:~$

```

- **Afficher les 20 dernières lignes du fichier “.bashrc”**

La commande **tail** et son option courte **-n** fonctionnent de la même manière que pour **head**.

**tail -n 20 .bashrc**

**tail -20 .bashrc**

**tail --lines 20 .bashrc**

```

herve@debian: ~
herve@debian:~$ tail -20 .bashrc

# Alias definitions.
# You may want to put all your additions into a separate file like
# ~/.bash_aliases, instead of adding them here directly.
# See /usr/share/doc/bash-doc/examples in the bash-doc package.

if [ -f ~/.bash_aliases ]; then
    . ~/.bash_aliases
fi

# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
    if [ -f /usr/share/bash-completion/bash_completion ]; then
        . /usr/share/bash-completion/bash_completion
    elif [ -f /etc/bash_completion ]; then
        . /etc/bash_completion
    fi
fi

```



# JOB 03

- **Installer le paquet "cmatrix"**

L'option **install** suivi du nom du paquet qui nous intéresse va nous permettre d'installer celui-ci.

```
herve@debian: ~  
root@debian:~# apt install cmatrix  
Lecture des listes de paquets... Fait  
Construction de l'arbre des dépendances... Fait  
Lecture des informations d'état... Fait  
Paquets suggérés :  
  cmatrix-xfont  
Les NOUVEAUX paquets suivants seront installés :  
  cmatrix  
0 mis à jour, 1 nouvellement installés, 0 à enlever et 0 non mis à jour.  
Il est nécessaire de prendre 17,5 ko dans les archives.  
Après cette opération, 53,2 ko d'espace disque supplémentaires seront utilisés.  
Réception de :1 http://deb.debian.org/debian bullseye/main amd64 cmatrix amd64 2  
.0-3 [17,5 kB]  
17,5 ko réceptionnés en 0s (89,8 ko/s)  
Sélection du paquet cmatrix précédemment désélectionné.  
(Lecture de la base de données... 140049 fichiers et répertoires déjà installés.)  
Préparation du dépaquetage de .../cmatrix_2.0-3_amd64.deb ...  
Dépaquetage de cmatrix (2.0-3) ...  
Paramétrage de cmatrix (2.0-3) ...  
Traitement des actions différées (« triggers ») pour mailcap (3.69) ...  
Traitement des actions différées (« triggers ») pour desktop-file-utils (0.26-1)  
...  
Traitement des actions différées (« triggers ») pour gnome-menus (3.36.0-1) ...
```

- **Lancer le paquet que vous venez d'installer**

Il faut simplement mettre le nom du programme en commande

```
herve@debian:~$ cmatrix
```

- **Mettre à jour son gestionnaire de paquets**

```
herve@debian:~$ sudo apt update  
Atteint :1 http://deb.debian.org/debian bullseye InRelease  
Atteint :2 http://security.debian.org/debian-security bullseye-security InRelease  
Atteint :3 http://deb.debian.org/debian bullseye-updates InRelease  
Atteint :4 https://dl.google.com/linux/chrome/deb stable InRelease  
Lecture des listes de paquets... Fait  
Construction de l'arbre des dépendances... Fait  
Lecture des informations d'état... Fait  
Tous les paquets sont à jour.
```

L'option **update** de la commande **apt** permet de mettre à jour la liste des paquets connus par le système.

- **Mettre à jour ses différents logiciels**

```
herve@debian:~$ sudo apt upgrade
```

L'option upgrade conserve les paquets dans leur version actuelle si la mise à jour nécessite l'installation de paquets supplémentaires pour satisfaire une nouvelle dépendance. La commande full-upgrade est moins conservatrice.

**sudo apt full-upgrade**

- **Télécharger les internets : Google**

On commence par télécharger le .deb de google chrome.

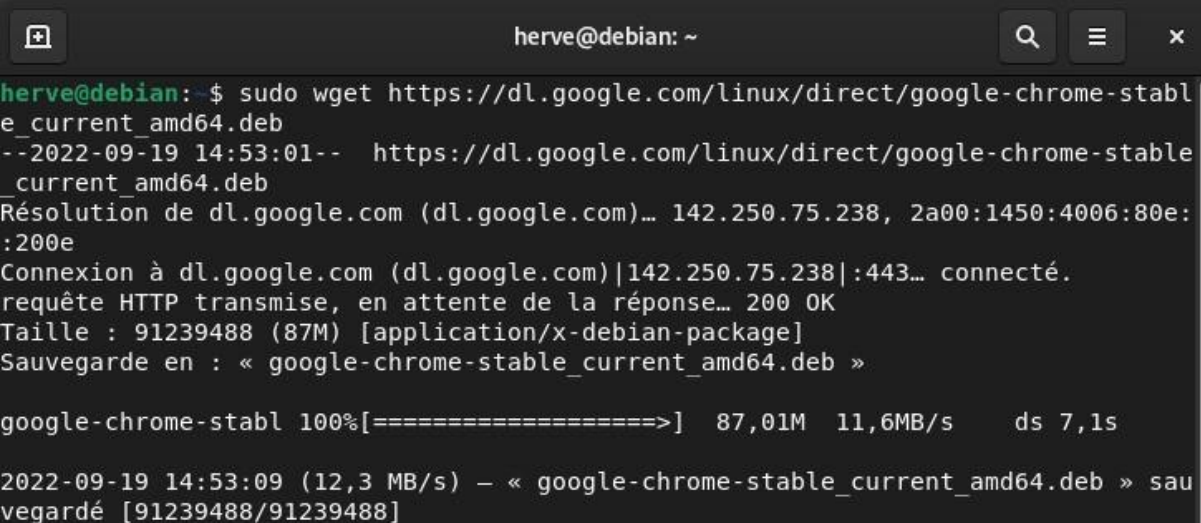
**wget** est un programme en ligne de commande non interactif de téléchargement de fichiers depuis le Web.

Il supporte les protocoles HTTP, HTTPS et FTP ainsi que le téléchargement au travers des proxys HTTP.

```
wget https://dl.google.com/linux/direct/google-chrome-
stable_current_amd64.deb
```

**sudo wget**

[https://dl.google.com/linux/direct/google-chrome-stable\\_current\\_amd64.deb](https://dl.google.com/linux/direct/google-chrome-stable_current_amd64.deb)



```
herve@debian: ~
herve@debian:~$ sudo wget https://dl.google.com/linux/direct/google-chrome-stable
e_current_amd64.deb
--2022-09-19 14:53:01-- https://dl.google.com/linux/direct/google-chrome-stable
_current_amd64.deb
Résolution de dl.google.com (dl.google.com)... 142.250.75.238, 2a00:1450:4006:80e:
:200e
Connexion à dl.google.com (dl.google.com)|142.250.75.238|:443... connecté.
requête HTTP transmise, en attente de la réponse... 200 OK
Taille : 91239488 (87M) [application/x-debian-package]
Sauvegarde en : « google-chrome-stable_current_amd64.deb »

google-chrome-stabl 100%[=====>] 87,01M 11,6MB/s ds 7,1s
2022-09-19 14:53:09 (12,3 MB/s) – « google-chrome-stable_current_amd64.deb » sau
vegardé [91239488/91239488]
```

Ensuite on rentre la commande pour installer Google Chrome de la même manière qu'avec **cmatrix**.

```
sudo apt install ./google-chrome-stable_current_amd64.deb
```

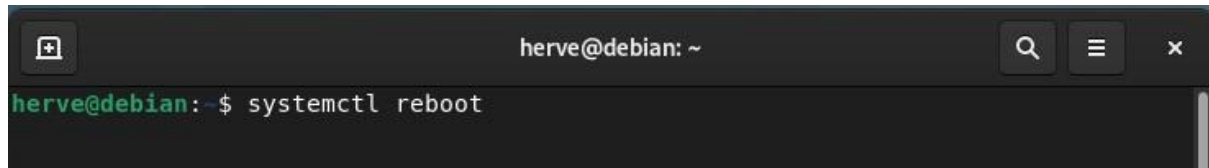
- **Redémarrer votre machine**

La commande **systemctl** fait partie du **démon systemd**.

**systemd** est couramment utilisé dans les nouvelles distributions de systèmes basées sur Linux pour gérer les services du système.

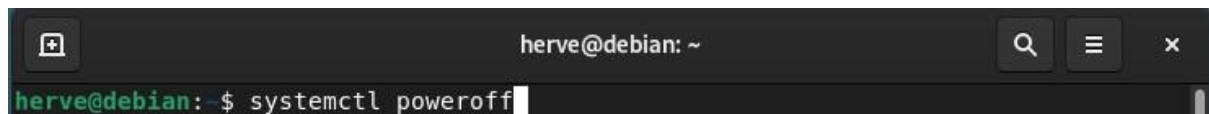
Un **démon** ou **daemon**, ou service du système, est un processus qui s'exécute en tâche de fond, généralement lancé pendant la séquence d'amorçage par Init. Les démons s'exécutent habituellement indépendamment des utilisateurs, à l'écoute des événements du système, fournissant des services en réponse.

Ici l'option **reboot** va nous permettre de redémarrer l'ordinateur.

A terminal window with a dark background. The title bar shows a window icon, the text 'herve@debian: ~', a search icon, a menu icon, and a close icon. The terminal content shows the prompt 'herve@debian: \$' followed by the command 'systemctl reboot'.

- **Éteindre votre machine**

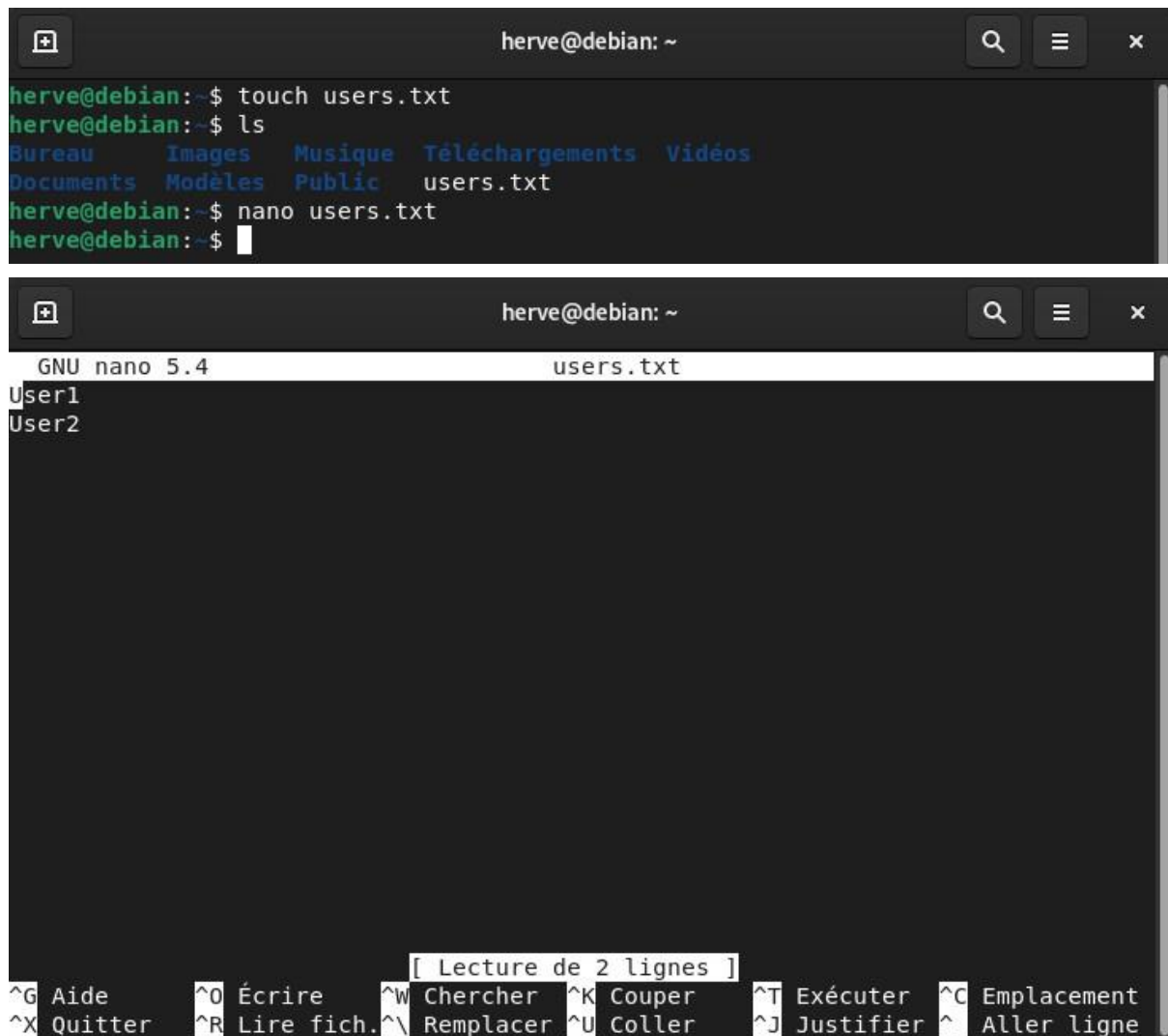
Ici l'option **poweroff** va nous permettre d'éteindre l'ordinateur.

A terminal window with a dark background. The title bar shows a window icon, the text 'herve@debian: ~', a search icon, a menu icon, and a close icon. The terminal content shows the prompt 'herve@debian: \$' followed by the command 'systemctl poweroff' and a cursor.

# JOB 04

- Créer un fichier `users.txt` qui contiendra "User1" et "User2" séparé par un retour à la ligne

Nous allons ici utiliser 2 commandes. La première **touch** pour créer un fichier texte nommé "users.txt". Et la 2eme commande va nous permettre d'éditer le fichier pour ajouter User1 et User2 en texte.



The first screenshot shows a terminal window with the prompt `herve@debian: ~`. The user enters `touch users.txt` and `ls`, which shows `users.txt` in the file list. Then the user enters `nano users.txt`. The second screenshot shows the `nano` editor interface for `users.txt`. The file contains two lines: `User1` and `User2`. The status bar at the bottom indicates "[ Lecture de 2 lignes ]" and lists various keyboard shortcuts.

```

herve@debian:~$ touch users.txt
herve@debian:~$ ls
Bureau  Images  Musique  Téléchargements  Vidéos
Documents  Modèles  Public  users.txt
herve@debian:~$ nano users.txt
herve@debian:~$
  
```

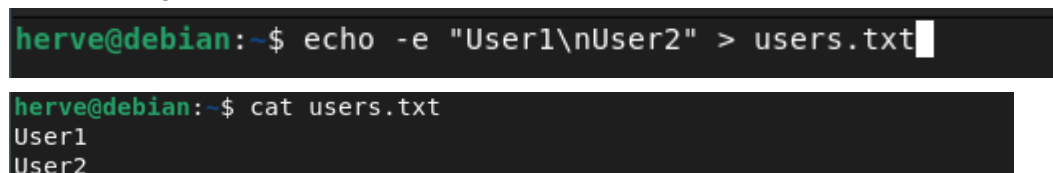
```

GNU nano 5.4 users.txt
User1
User2
  
```

[ Lecture de 2 lignes ]

^G Aide    ^O Écrire    ^W Chercher    ^K Couper    ^T Exécuter    ^C Emplacement  
^X Quitter    ^R Lire fich.    ^\ Remplacer    ^U Coller    ^J Justifier    ^\_ Aller ligne

Une 2ème méthode impliquant une seule ligne de code existe. La commande **echo** dans un terminal, affiche le texte en argument. Ici nous allons la coupler avec un chevron et le nom du fichier texte dans lequel nous voulons écrire User1 et User2 avec un retour à la ligne, ce retour à la ligne est caractérisé par `\n`. Ce qui nous donne :



The first screenshot shows the command `echo -e "User1\nUser2" > users.txt` being entered. The second screenshot shows the command `cat users.txt` being entered, which outputs `User1` and `User2` on separate lines.

```

herve@debian:~$ echo -e "User1\nUser2" > users.txt
herve@debian:~$ cat users.txt
User1
User2
  
```

- **Créer un groupe appelé "Plateformeurs"**

La commande **groupadd** permet d'ajouter de nouveaux groupes au système.

```
herve@debian: ~
herve@debian:~$ sudo groupadd Plateformeurs
```

- **Créer un utilisateur appelé "User1"**

La commande **useradd** permet de créer un nouvel utilisateur.

Pour utiliser cette commande, les droits superutilisateur sont nécessaires.

```
herve@debian: ~
herve@debian:~$ sudo useradd User1
```

- **Créer un utilisateur appelé "User2"**

```
herve@debian: ~
herve@debian:~$ sudo useradd User1
herve@debian:~$ sudo useradd User2
herve@debian:~$
```

- **Ajouter "User2" au groupe Plateformeurs**

Si **adduser** est appelé avec deux arguments sans option, il va ajouter un utilisateur existant dans un groupe existant.

```
herve@debian:~$ sudo adduser User2 Plateformeurs
Ajout de l'utilisateur « User2 » au groupe « Plateformeurs »...
Adding user User2 to group Plateformeurs
Fait.
```

- **Copier votre "users.txt" dans un fichier "droits.txt"**

La commande **cp** (copy paste) permet la copie de fichiers.

```
herve@debian: ~
herve@debian:~$ man cp
herve@debian:~$ cp users.txt droits.txt
herve@debian:~$ ls
Bureau      droits.txt  Modèles    Public      users.txt
Documents  Images     Musique    Téléchargements  Vidéos
herve@debian:~$ cat droits.txt
User1
User2
herve@debian:~$
```



- Copier votre "users.txt" dans un fichier "groupes.txt"

La commande **cp** (copy paste) permet la copie de fichiers.

```
herve@debian:~$ cp users.txt groupes.txt
herve@debian:~$ ls
Bureau    droits.txt  Images      Musique     Téléchargements  Vidéos
Documents groupes.txt Modèles     Public      users.txt
herve@debian:~$ cat groupes.txt
User1
User2
herve@debian:~$
```

- Changer le propriétaire du fichier "droits.txt" pour mettre "User1"

La commande **chown** permet de modifier le propriétaire et le groupe d'un fichier.

Seul l'utilisateur root peut changer le propriétaire d'un fichier.

```
herve@debian:~$ sudo chown User1 droits.txt
herve@debian:~$ ls -l
total 44
drwxr-xr-x 2 herve herve 4096 14 sept. 14:19 Bureau
drwxr-xr-x 2 herve herve 4096 19 sept. 11:03 Documents
-rw-r--r-- 1 User1 herve  12 19 sept. 23:16 droits.txt
```

- Changer les droits du fichier "droits.txt" pour que "User2" ai accès seulement en lecture

La commande **chmod** permet de modifier les permissions aux différents types d'accès (rwx) des fichiers (et répertoire) indépendamment pour le propriétaire, le groupe ou les autres utilisateurs.

```
herve@debian:~$ sudo chmod 744 droits.txt
[sudo] Mot de passe de herve :
herve@debian:~$ ls -l
total 44
drwxr-xr-x 2 herve herve 4096 14 sept. 14:19 Bureau
drwxr-xr-x 2 herve herve 4096 19 sept. 11:03 Documents
-rwxr--r-- 1 User1 herve  12 19 sept. 23:16 droits.txt
```

- Changer les droits du fichier "groupes.txt" pour que les utilisateurs puissent accéder au fichier en lecture uniquement

```
herve@debian:~$ sudo chmod 744 groupes.txt
[sudo] Mot de passe de herve :
herve@debian:~$ ls -l
total 44
drwxr-xr-x 2 herve herve 4096 14 sept. 14:19 Bureau
drwxr-xr-x 2 herve herve 4096 19 sept. 11:03 Documents
-rwxr--r-- 1 User1 herve  12 19 sept. 23:16 droits.txt
-rwxr--r-- 1 herve herve  12 19 sept. 23:19 groupes.txt
```

- **Changer les droits du fichier pour que le groupe "Plateformeurs" puissent y accéder en lecture/écriture.**

La commande **chgrp** permet de changer le groupe propriétaire du fichier. Nous allons donc commencer par attribuer le groupe au fichier qui nous intéresse.

```
herve@debian:~$ sudo chgrp Plateformeurs groupes.txt
herve@debian:~$ ls -l
total 44
drwxr-xr-x 2 herve herve          4096 14 sept. 14:19 Bureau
drwxr-xr-x 2 herve herve          4096 19 sept. 11:03 Documents
-rwxr--r-- 1 User1 herve           12 19 sept. 23:16 droits.txt
-rwxr--r-- 1 herve Plateformeurs   12 19 sept. 23:19 groupes.txt
```

Une fois le groupe attribué au fichier, nous allons pouvoir en modifier les droits. Etant donné que nous souhaitons ajouter les droit en écriture nous allons faire :

```
herve@debian:~$ sudo chmod g+w groupes.txt
herve@debian:~$ ls -l
total 44
drwxr-xr-x 2 herve herve          4096 14 sept. 14:19 Bureau
drwxr-xr-x 2 herve herve          4096 19 sept. 11:03 Documents
-rwxr--r-- 1 User1 herve           12 19 sept. 23:16 droits.txt
-rwxrw-r-- 1 herve Plateformeurs   12 19 sept. 23:19 groupes.txt
```

Deux méthodes sont viables pour changer les droits d'un fichier :

- La 1ere méthode implique d'ajouter une option définissant si nous souhaitons modifier les droits pour le propriétaire, le groupe ou les utilisateurs (u,g,o ou all pour tous). Suivi d'un + (pour ajouter) ou d'un - (pour retirer des droits) et défini par r,w,x pour les différents droits en écriture, lecture ou exécution. Cette méthode permet de définir les droits pour une catégorie à la fois ou toutes en même temps.

Exemple:

```
herve@debian:~$ sudo chmod g+w groupes.txt
```

Ici nous avons souhaité ajouter au groupe propriétaire du fichier les droits en écriture sans modifier les autres droits.

- La 2eme méthode consiste à convertir les droits (r,w,x) en leur valeur octale pour simplifier la commande. Cependant cela implique de modifier les droits pour toutes les catégories d'un seul coup.

Position Binaire	Valeur octale	Droits	Signification
000	0	---	Aucun droit
001	1	--x	Exécutable
010	2	-w-	Ecriture
011	3	-wx	Ecrire et exécuter
100	4	r--	Lire
101	5	r-x	Lire et exécuter
110	6	rw-	Lire et écrire
111	7	rw x	Lire écrire et exécuter

Exemple :

```
herve@debian:~$ sudo chmod 744 groupes.txt
```

Ici nous avons modifié les droits pour toutes les catégories d'utilisateurs. Le 744 signifie que nous avons donné les droits (r,w,x) au propriétaire puis lecture au groupe et aux utilisateurs.



# JOB 05

- Ajouter un alias qui permettra de lancer la commande **"ls -la"** en tapant **"la"**

Création du fichier `~/.bash_aliases` pour éditer les alias sans modifier le fichier `~/bashrc`

```
herve@debian:~$ sudo nano ~/.bash_aliases
```



```

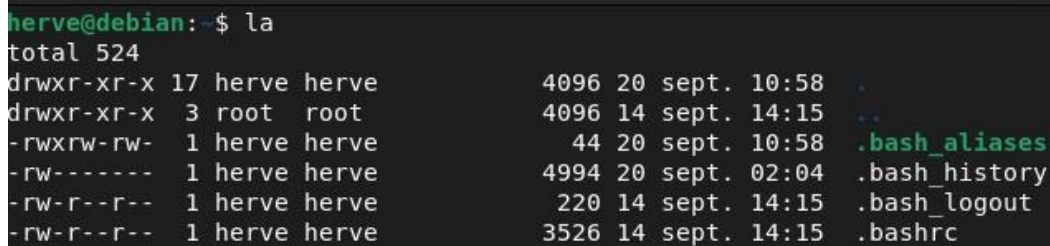
herve@debian: ~
GNU nano 5.4 /home/herve/.bash_aliases *
alias la="ls -la"

```

Pour profiter de l'alias nouvellement créé, il suffit de redémarrer le terminal ou alors d'exécuter le fichier.

```
herve@debian:~$ source ~/.bash_aliases
```

L'alias est maintenant disponible dans les commandes.



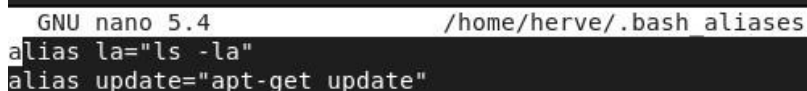
```

herve@debian:~$ la
total 524
drwxr-xr-x 17 herve herve      4096 20 sept. 10:58 .
drwxr-xr-x  3 root  root      4096 14 sept. 14:15 ..
-rwxr--r--  1 herve herve       44 20 sept. 10:58 .bash_aliases
-rw-----  1 herve herve    4994 20 sept. 02:04 .bash_history
-rw-r--r--  1 herve herve     220 14 sept. 14:15 .bash_logout
-rw-r--r--  1 herve herve    3526 14 sept. 14:15 .bashrc

```

- Ajouter un alias qui permettra de lancer la commande **"apt-get update"** en tapant **"update"**

```
herve@debian:~$ sudo nano ~/.bash_aliases
```

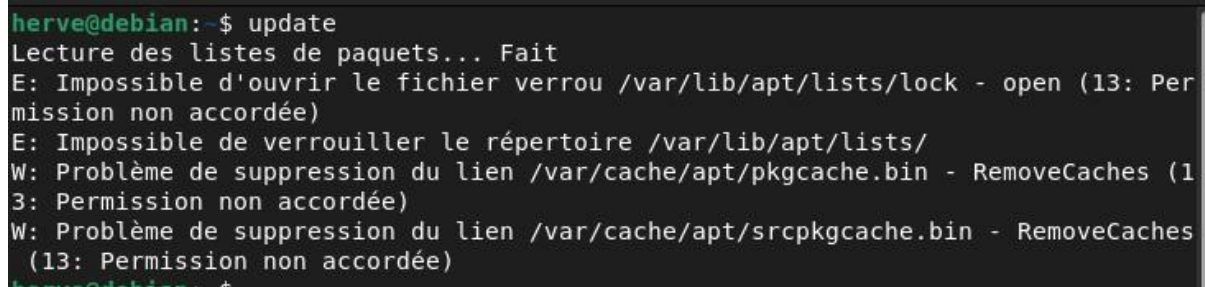


```

GNU nano 5.4 /home/herve/.bash_aliases
alias la="ls -la"
alias update="apt-get update"

```

```
herve@debian:~$ source ~/.bash_aliases
```



```

herve@debian:~$ update
Lecture des listes de paquets... Fait
E: Impossible d'ouvrir le fichier verrou /var/lib/apt/lists/lock - open (13: Permission non accordée)
E: Impossible de verrouiller le répertoire /var/lib/apt/lists/
W: Problème de suppression du lien /var/cache/apt/pkgcache.bin - RemoveCaches (13: Permission non accordée)
W: Problème de suppression du lien /var/cache/apt/srcpkgcache.bin - RemoveCaches (13: Permission non accordée)
herve@debian:~$

```

- Ajouter un alias qui permettra de lancer la commande **"apt-get upgrade"** en tapant **"upgrade"**

```
herve@debian:~$ sudo nano ~/.bash_aliases
```

```
GNU nano 5.4 /home/herve/.bash_aliases *
alias la="ls -la"
alias update="sudo apt-get update"
alias upgrade="sudo apt-get upgrade"
```

```
herve@debian:~$ source ~/.bash_aliases
```

```
herve@debian:~$ upgrade
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Calcul de la mise à jour... Fait
0 mis à jour, 0 nouvellement installés, 0 à enlever et 0 non mis à jour.
```

- Ajouter une variable d'environnement qui se nommera **"USER"** et qui sera égale à votre nom d'utilisateur

```
herve@debian:~$ export USER=herve
```

- Mettre à jour les modifications de votre **bashrc** dans votre shell actuel

La commande **source** peut être utilisée pour recharger/rafraîchir un fichier de fonctions dans le script shell courant ou d'une invite de commande.

Une fois le fichier **/home/user/.bashrc** modifié, pour le faire prendre en compte par la session courante, il faut lancer la commande **source** avec en paramètre le fichier.

```
herve@debian:~$ source ~/.bashrc
```

- Afficher les variables d'environnement

```
herve@debian:~$ printenv
```

```
herve@debian:~$ env
```

- Ajouter à votre Path le chemin **"/home/votre utilisateur/Bureau"**

```
herve@debian:~$ export PATH=$PATH:/home/herve/Bureau
```

## JOB 06

Nous constatons que le lien fourni ne se finit pas en **tar.gz** par conséquent le lien ne donne pas une archive si on utilise la commande **wget** mais nous aurons un lien web.

Nous devons ouvrir le lien fourni dans le sujet avec **open** et télécharger l'archive à partir du navigateur et de google drive.

```
herve@debian:~$ open https://drive.google.com/file/d/1ldSelXQuH4tih6zesbv-60MEpr-sT77X/view?usp=sharing
```



Une fois l'archive téléchargée nous allons pouvoir décompresser le fichier correspondant et ensuite l'ouvrir grâce au terminal :

```
herve@debian:~$ tar -xvf "Copie de Ghost in the Shell.tar.gz"
Ghost in the Shell.pdf
herve@debian:~$ open "Ghost in the Shell.pdf"
herve@debian:~$
```

La commande **tar** permet d'archiver ou de désarchiver des répertoires et des fichiers de façon optimale.

Plusieurs types d'archives **tar** sont possibles comme le **.gz** ou le **.bz2**. Les options changent légèrement suivant que nous voulons archiver/décompresser un type d'archive. Pour la décompression plusieurs options sont possibles.

Pour les archives de types **.gz**

```
x: permet d'extraire certains fichiers d'une archive
z: décompacte l'archive avec l'utilitaire gzip
f: extrait un fichier donné (ici le fichier est nom_du_fichier.tar.gz)
```

Et pour les archives de types **.bz2**

```
x: permet d'extraire certains fichiers d'une archive
j: décompacte l'archive avec l'utilitaire bzip2
f: extrait un fichier donné (ici le fichier est nom_du_fichier.tar.bz2)
```

Pour la compression voici les options disponibles

```
c: indique à tar de créer une archive
z: indique à tar de compacter une archive avec l'utilitaire gzip (-j pour bzip2)
v: est le mode "verbose", qui affiche les noms des fichiers tel qu'ils ont été archivés à l'origine
t: affiche la liste du contenu de l'archive sans l'extraire (à la place de -x ou -c)
p: préserver les permissions des fichiers
```

# JOB 07

- Créer un fichier "une\_commande.txt" avec le texte suivant "Je suis votre fichier texte"

1ere méthode

```
herve@debian:~$ echo "Je suis votre fichier texte" > une_commande.txt
herve@debian:~$ cat une_commande.txt
Je suis votre fichier texte
herve@debian:~$ ls
Bureau      groupes.txt  Téléchargements
'Copie de Ghost in the Shell.tar.gz'  Images      une_commande.txt
```

2eme méthode

```
herve@debian:~$ cat > une_commande.txt
Je suis votre fichier texte
^C
herve@debian:~$ cat une_commande.txt
Je suis votre fichier texte
herve@debian:~$ ls
Bureau      groupes.txt  Téléchargements
'Copie de Ghost in the Shell.tar.gz'  Images      une_commande.txt
```

- Compter le nombre de lignes présentes dans votre fichier de source apt et les enregistrer dans un fichier nommé "nb\_lignes.txt"

```
herve@debian:~$ wc -l /etc/apt/sources.list|cat > nb_lignes.txt
herve@debian:~$ cat nb_lignes.txt
20 /etc/apt/sources.list
```

***wc -l /etc/apt/sources.list | cat > nb\_lignes.txt***

Et si nous ne voulons pas afficher le fichier correspondant on peut faire :

```
herve@debian:~$ wc -l < /etc/apt/sources.list|cat > nb_lignes.txt
herve@debian:~$ cat nb_lignes.txt
20
```

***wc -l < /etc/apt/sources.list | cat > nb\_lignes.txt***

- Afficher le contenu du fichier source apt et l'enregistrer dans un autre fichier appelé "save\_sources"

```
herve@debian:~$ cat /etc/apt/sources.list|cat > save_sources|cat /etc/apt/sources.list
# deb cdrom:[Debian GNU/Linux 11.5.0 _Bullseye_ - Official amd64 NETINST 20220910-10:38]/ b
ullseye main
```

***cat /etc/apt/sources.list|cat > save\_sources|cat /etc/apt/sources.list***

- Faites une recherche des fichiers commençant par "." tout en cherchant le mot alias qui sera utilisé depuis un fichier

```
herve@debian:~$ grep -r alias .*
```

### Qu'est ce qu'un chevron ?

Le chevron ">" est une commande de redirection.

Par redirection, on entend la possibilité de rediriger l'affichage de l'écran vers un fichier, une imprimante ou tout autre périphérique, les messages d'erreurs vers un autre fichier, remplacer la saisie clavier par le contenu d'un fichier.

> permet aussi de vider le contenu d'un fichier sans le supprimer. L'intérêt de vider un fichier de cette manière, par rapport à un **rm** et un **touch**, c'est que l'on conserve le numéro d'inode (numéro d'identification du fichier dans le système).

Les commandes qui attendent des données ou des paramètres depuis le clavier peuvent aussi en recevoir depuis un fichier, à l'aide du caractère inverse "<".

Unix utilise des canaux d'entrées/sorties pour lire et écrire ses données.

Par défaut le canal d'entrée est le clavier, et le canal de sortie, l'écran.

Un troisième canal, le canal d'erreur, est aussi redirigé vers l'écran.

Pour ne rien effacer et/ou ajouter des données à la suite du fichier resultat.txt on utilise la double redirection représenté par ">>".

### Qu'est ce qu'un pipe ?

Le caractère pipe est "|".

Les redirections d'entrée/sortie permettent de rediriger les résultats vers un fichier. Ce fichier peut ensuite être réinjecté dans un filtre pour en extraire d'autres résultats.

Cela oblige à taper deux lignes :

- une pour la redirection vers un fichier,
- l'autre pour rediriger ce fichier vers le filtre.

Les tubes ou pipes permettent de rediriger directement le canal de sortie → d'une commande vers → le canal d'entrée d'autre.

Ainsi les 2 redirections suivantes

```
herve@debian:~$ ls -l > resultat.txt
```

```
herve@debian:~$ wc < resultat.txt
20 181 1337
```

Deviennent cette commande unique

```
herve@debian:~$ ls -l | wc
```

ou si l'on souhaite aussi écrire le résultat dans un fichier

```
herve@debian:~$ ls -l | wc > resultat.txt
```

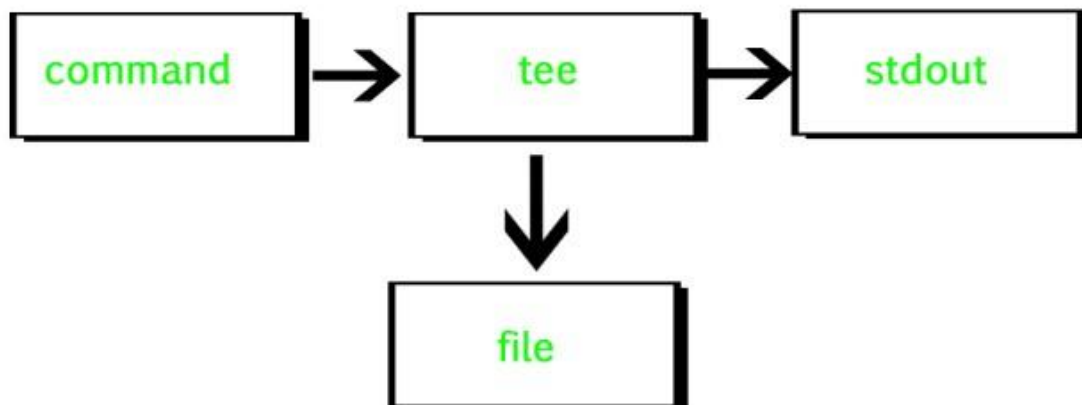
# POUR ALLER PLUS LOIN

- Installer la commande **tree**

```
herve@debian:~$ sudo apt install tree
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Les NOUVEAUX paquets suivants seront installés :
 tree
```

- Lancer la commande **tree** en arrière-plan qui aura pour but d'afficher toute l'arborescence de votre / en enregistrant le résultat dans un fichier "tree.save"

Ici la commande **tee** lit l'entrée standard et l'écrit à la fois dans la sortie standard et dans un ou plusieurs fichiers. La commande porte le nom du séparateur en T utilisé en plomberie. Il casse essentiellement la sortie d'un programme afin qu'il puisse être à la fois affiché et enregistré dans un fichier. Il effectue les 2 tâches simultanément, copie le résultat dans les fichiers ou variables spécifiés et affiche également le résultat.



L'utilisation de **&** permet de lancer une commande en arrière plan.

```
herve@debian:~$ tree / | tee tree.save &
```

```
tree / | tee tree.save &
```

- Lister les éléments présents dans le dossier courant et utiliser directement le résultat de votre première commande pour compter le nombre d'éléments trouvés

Ici l'utilisation d'un **pipe** est essentiel pour rediriger le canal de sortie de la commande **ls** vers le canal d'entrée de la commande **wc** et ainsi interpréter les résultats demandés.

```
herve@debian:/$ ls -a | wc -l
29
```

```
ls -a | wc -l
```



- **Lancer une commande pour update vos paquets, si l'update réussit alors, vous devrez lancer un upgrade de vos paquets. Si l'update échoue, votre upgrade ne se lancera pas**

L'exécution de cette commande implique une condition. Si la 1ère commande réussit, alors la 2ème commande est exécutée.

```
command1 && command2
```

command2 s'exécutera si command1 s'est exécuté avec succès. Cet opérateur nous permet de vérifier l'état de sortie de command1.

**&&** est à ne pas confondre avec l'opérateur point-virgule ";". Celui-ci est utilisé pour exécuter plusieurs commandes en une seule fois. L'exécution de la commande qui succède à cet opérateur s'exécutera toujours après l'exécution de la commande qui le précède, quel que soit l'état de sortie de la commande précédente.

```
command1 ; command2
```

L'exécution de la deuxième commande est indépendante de l'état de sortie de la première commande. Si la première commande n'est pas exécutée avec succès, la deuxième commande sera également exécutée.

Opérateur && (ET logique)	; Opérateur (point-virgule)
L'exécution de la deuxième commande dépend de l'exécution de la première commande	L'exécution de la deuxième commande est indépendante de l'état d'exécution de la première commande.
Si l'état de sortie de la commande précédente est différent de zéro, la commande suivante ne sera pas exécutée.	Même si l'état de sortie de la première commande est différent de zéro, la deuxième commande sera exécutée.
Autorise l'exécution conditionnelle	N'autorise pas l'exécution conditionnelle.
Bash a court-circuité l'évaluation du ET logique.	Aucune évaluation de court-circuit n'est nécessaire.
Le ET logique a une priorité plus élevée.	Il a moins de priorité que le ET logique.

Par conséquent nous utiliserons **&&** pour exécuter la commande **apt upgrade** si **apt update** réussit.

```
herve@debian:~$ sudo apt update && sudo apt upgrade
```