



MI4 - 2017 / 2018

Informatique pour le calcul scientifique

Alexandre Pinlou
alexandre.pinlou@umontpellier.fr
Pôle Informatique Transversal

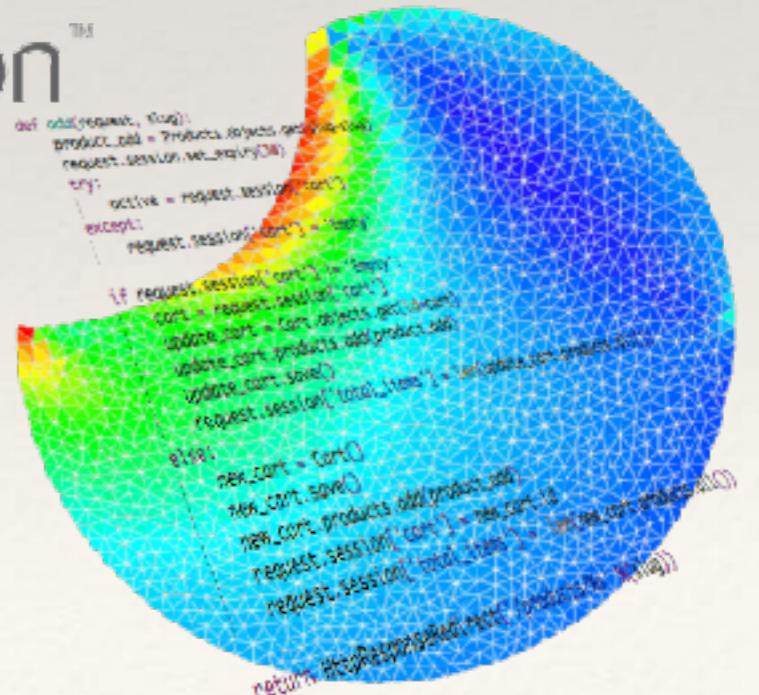


Organisation

- ❖ Séances :
 - ▶ 5 CM/TD (1h30)
 - ▶ 2 TP (3h chacun)

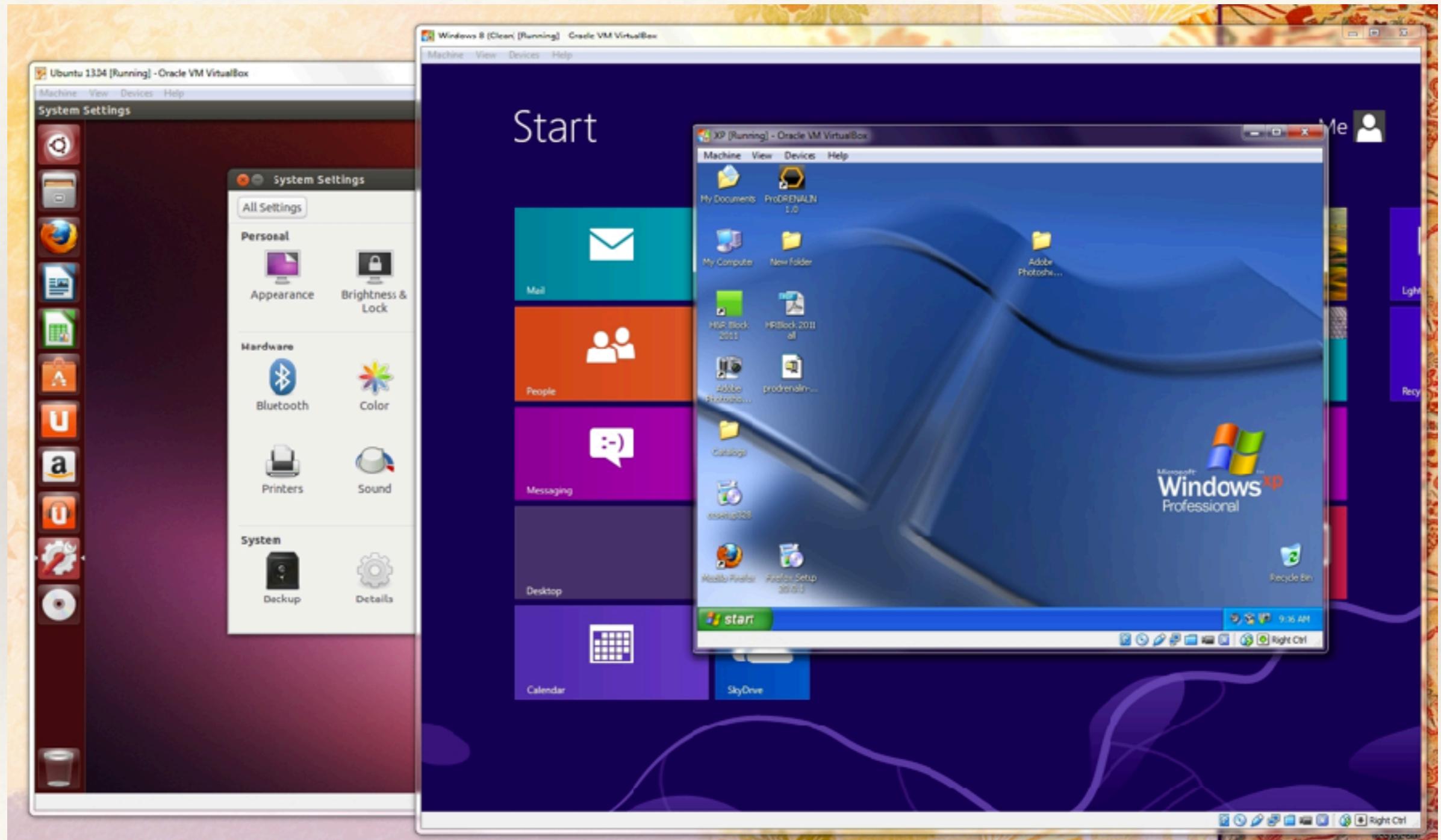
- ❖ Notions abordées :
 - ▶ Utilisation d'un système Linux  Linux
 - ▶ Programmation en Python  python

- ❖ Notation :
 - ▶ 1 projet : Optimisation de forme



Initiation aux systèmes Linux

Utilisation d'une VM Linux



Utilisation d'une machine virtuelle (VM)

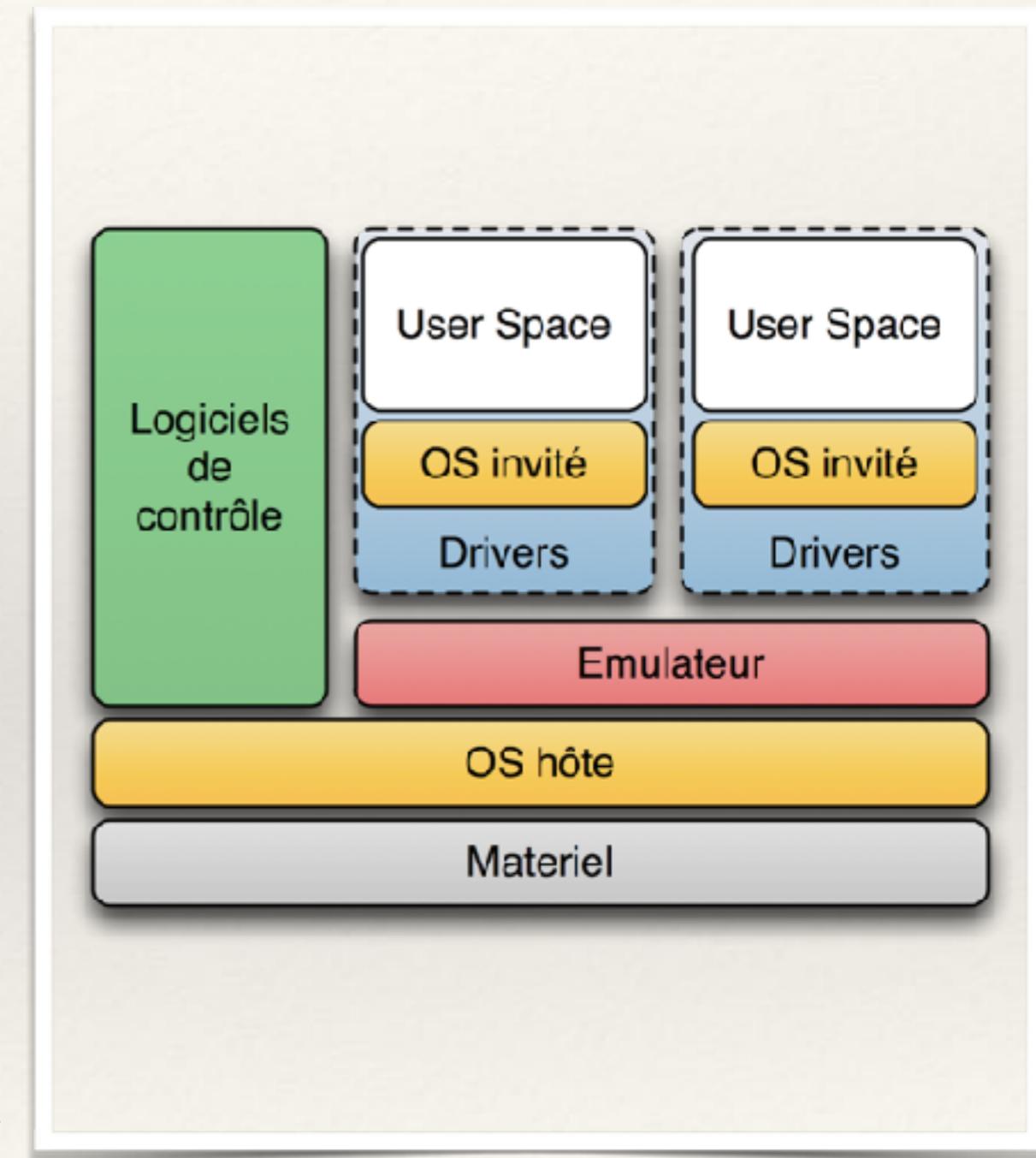
- ❖ Loguez-vous sur votre machine (système Windows)
- ❖ Aller sur le lecteur réseau **Y:**\ (publicens) et récupérer les 3 fichiers suivants :

Lubuntu.zip dans **C:\Linux**

LubuntuSTX.vbox dans **C:\Linux**

home.vdi dans **P:\Linux**

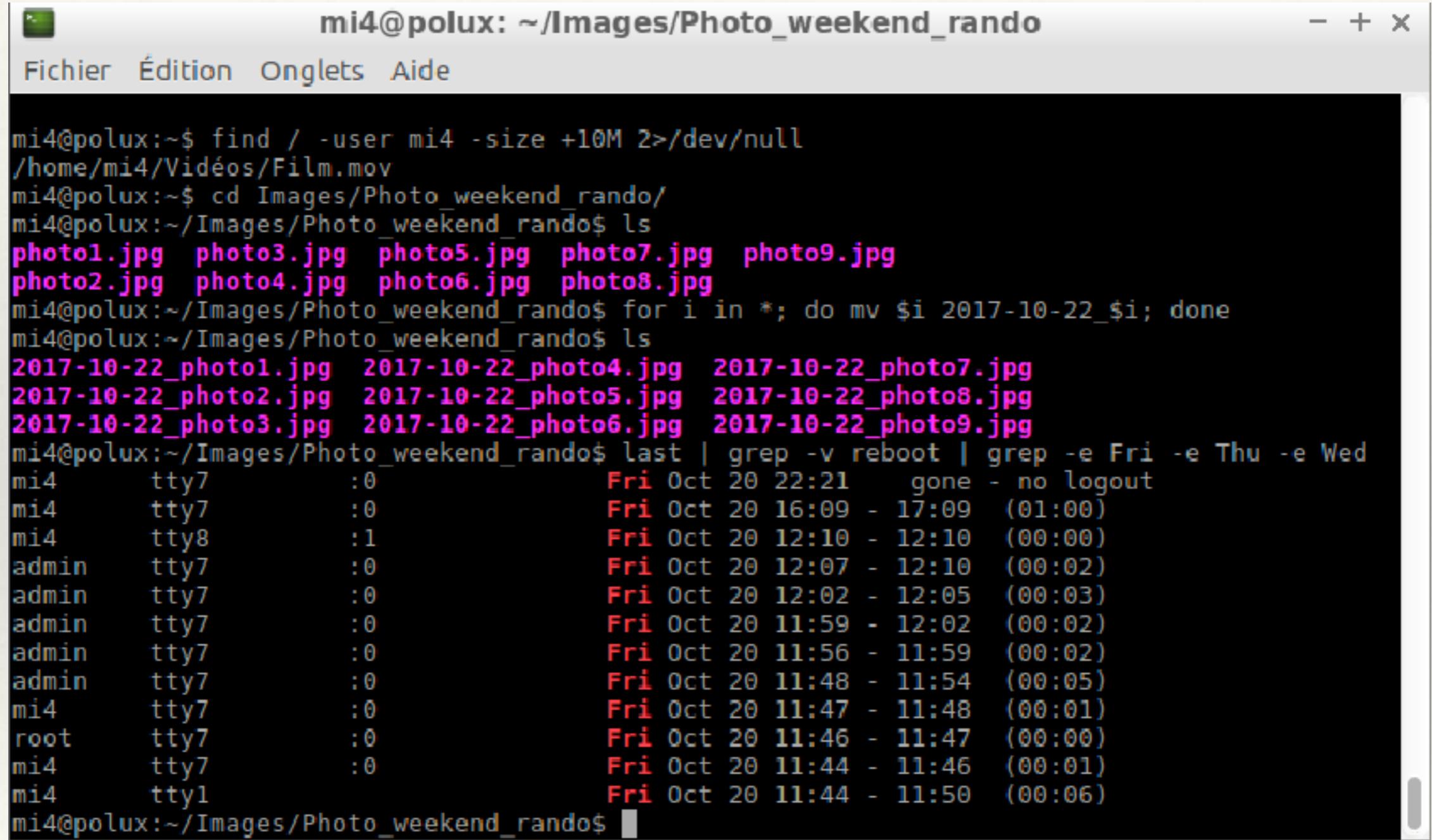
- ❖ Dézippez le fichier **.zip**
- ❖ Double-cliquez sur le fichier **.vbox** pour lancer le programme VirtualBox



Les systèmes d'exploitation

- ❖ **S.E. (O.S.)** : Programme de base qui assure la gestion du matériel (clavier, écran, disque dur, ...), du système de fichiers et des applications (logiciels) des utilisateurs
- ❖ **Les différents S.E. :**
 - ▶ *UNIX* : Linux, BSD, Ubuntu, Solaris, MacOS X, ...
 - ▶ *Windows* : NT, 2000, XP, Seven, 10, ...
 - ▶ *MacOS < 10*
- ❖ **Avantages d'UNIX**
 - ▶ Stable et sûr (peu de virus) car OpenSource
 - ▶ Langage de commandes
- ❖ **Pourquoi vous parler d'UNIX ?**
 - ▶ En entreprise, les logiciels de calcul scientifique sont souvent installés sur des serveurs UNIX.
 - ▶ Ces serveurs sont souvent uniquement accessible à distance en ligne de commande.

Utilisation d'un terminal



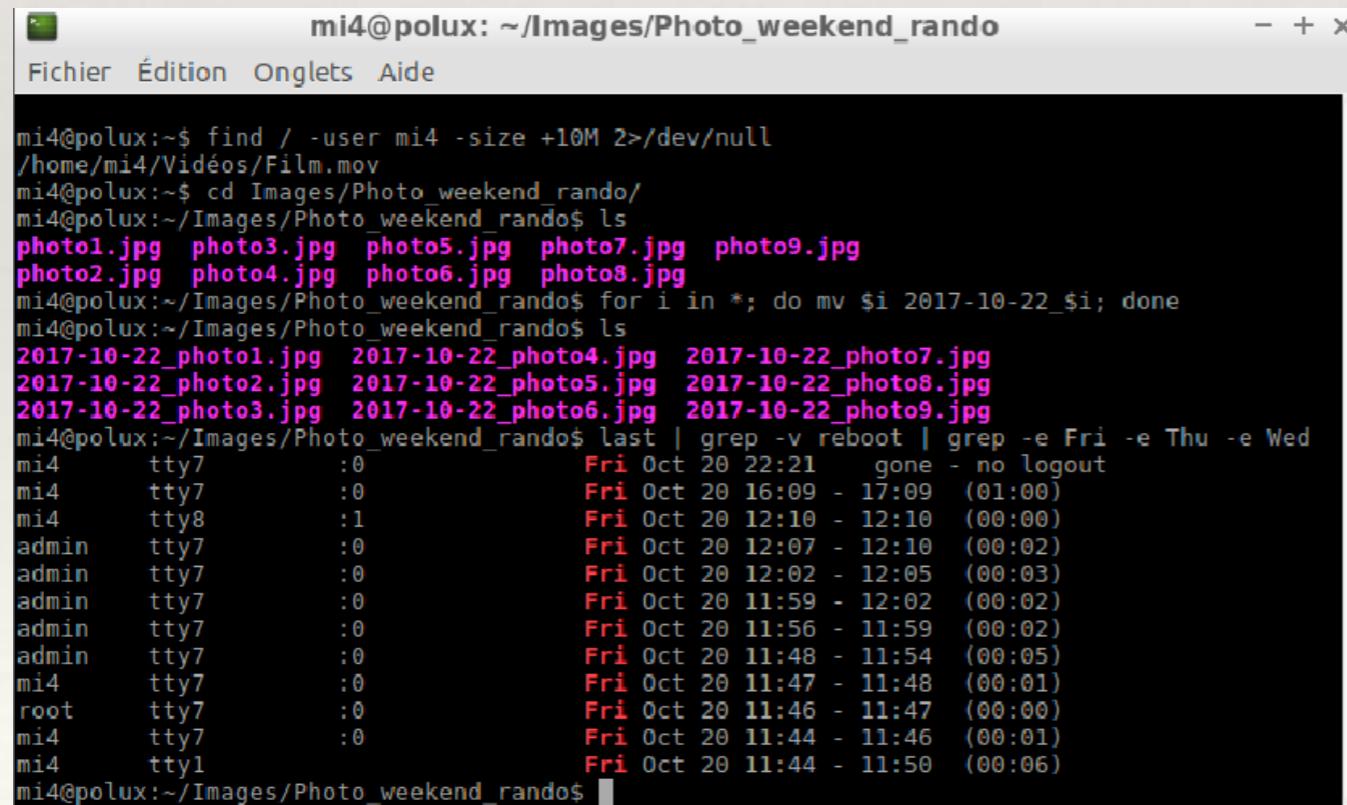
The screenshot shows a terminal window with the following details:

- Title Bar:** mi4@polux: ~/Images/Photo_weekend_rando
- Menu Bar:** Fichier Édition Onglets Aide
- Terminal Content:**

```
mi4@polux:~$ find / -user mi4 -size +10M 2>/dev/null
/home/mi4/Vidéos/Film.mov
mi4@polux:~$ cd Images/Photo_weekend_rando/
mi4@polux:~/Images/Photo_weekend_rando$ ls
photo1.jpg  photo3.jpg  photo5.jpg  photo7.jpg  photo9.jpg
photo2.jpg  photo4.jpg  photo6.jpg  photo8.jpg
mi4@polux:~/Images/Photo_weekend_rando$ for i in *; do mv $i 2017-10-22_$i; done
mi4@polux:~/Images/Photo_weekend_rando$ ls
2017-10-22_photo1.jpg  2017-10-22_photo4.jpg  2017-10-22_photo7.jpg
2017-10-22_photo2.jpg  2017-10-22_photo5.jpg  2017-10-22_photo8.jpg
2017-10-22_photo3.jpg  2017-10-22_photo6.jpg  2017-10-22_photo9.jpg
mi4@polux:~/Images/Photo_weekend_rando$ last | grep -v reboot | grep -e Fri -e Thu -e Wed
mi4    tty7      :0          Fri Oct 20 22:21      gone - no logout
mi4    tty7      :0          Fri Oct 20 16:09 - 17:09  (01:00)
mi4    tty8      :1          Fri Oct 20 12:10 - 12:10  (00:00)
admin  tty7      :0          Fri Oct 20 12:07 - 12:10  (00:02)
admin  tty7      :0          Fri Oct 20 12:02 - 12:05  (00:03)
admin  tty7      :0          Fri Oct 20 11:59 - 12:02  (00:02)
admin  tty7      :0          Fri Oct 20 11:56 - 11:59  (00:02)
admin  tty7      :0          Fri Oct 20 11:48 - 11:54  (00:05)
mi4    tty7      :0          Fri Oct 20 11:47 - 11:48  (00:01)
root   tty7      :0          Fri Oct 20 11:46 - 11:47  (00:00)
mi4    tty7      :0          Fri Oct 20 11:44 - 11:46  (00:01)
mi4    tty1      :0          Fri Oct 20 11:44 - 11:50  (00:06)
mi4@polux:~/Images/Photo_weekend_rando$
```

Le terminal de commandes

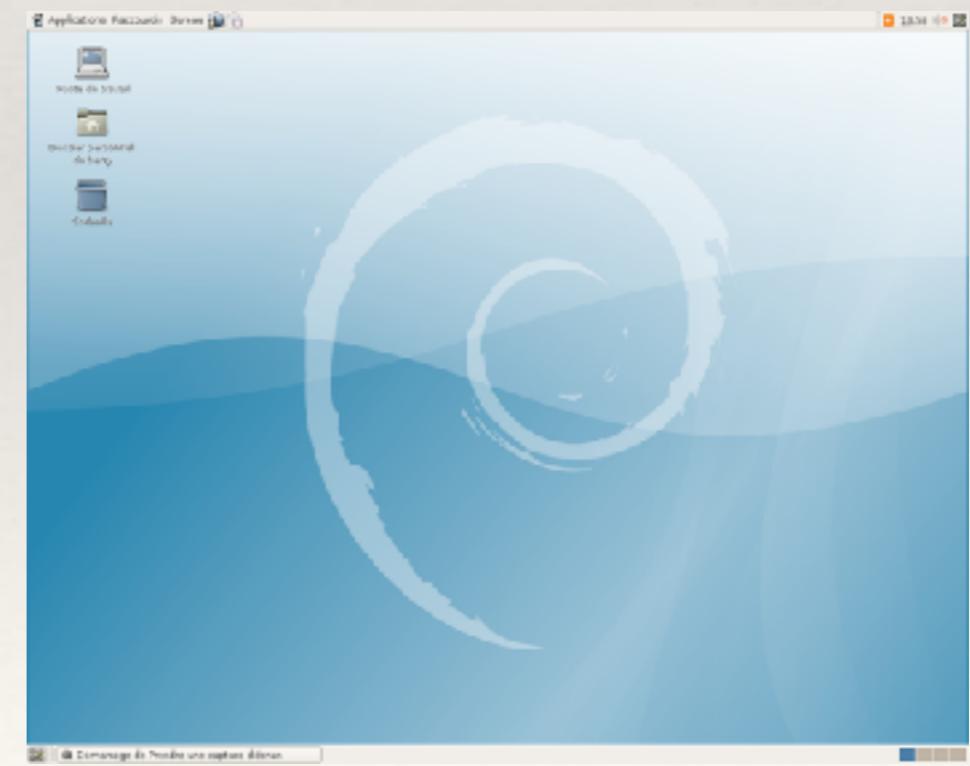
- ❖ Un **terminal de commande** est une application qui permet à l'utilisateur d'interagir avec le S.E. par le biais de lignes de commandes.
- ❖ Cette application permet de lui demander des **traitements sophistiqués** que l'on ne peut réaliser par le « cliquodrome » de l'interface graphique.



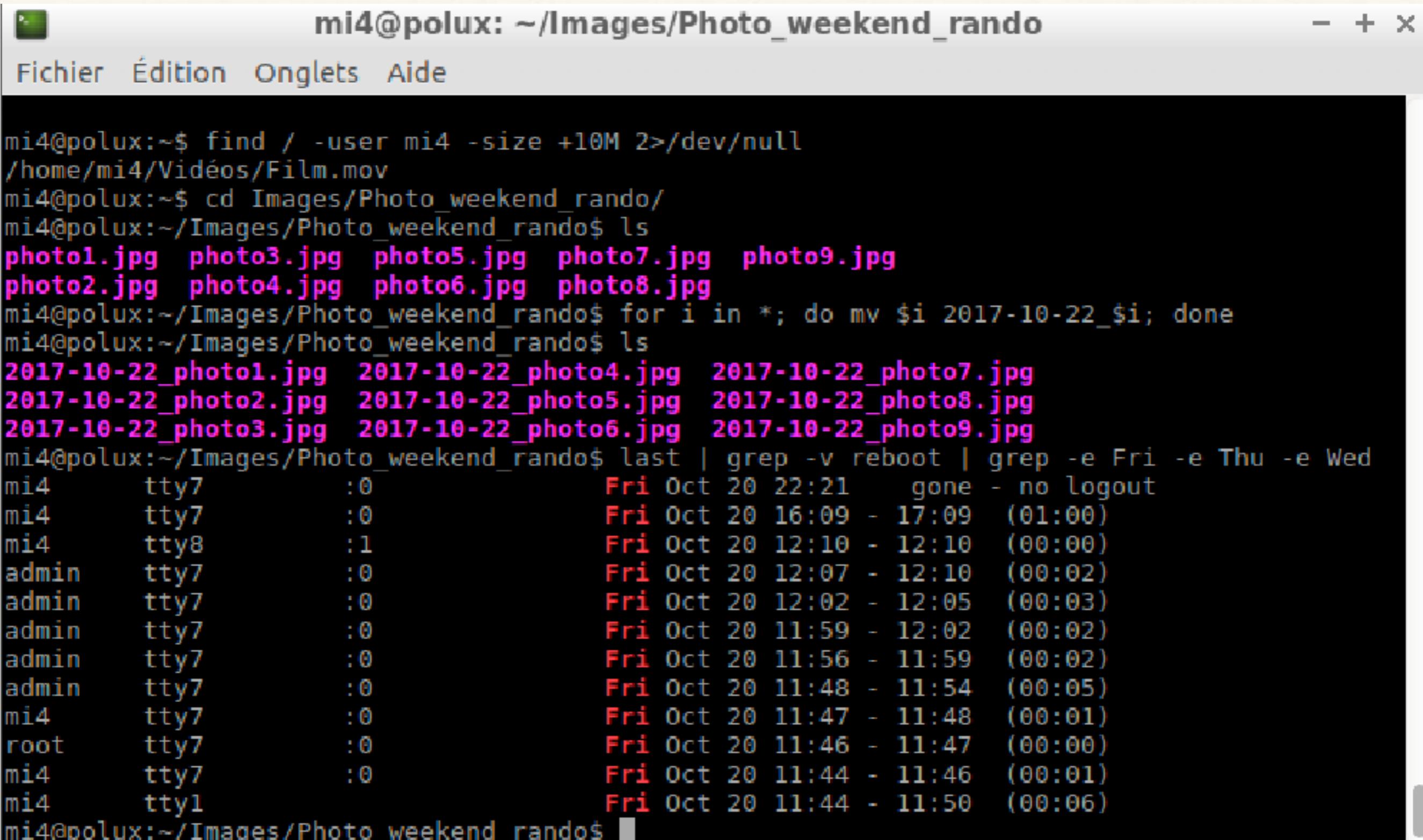
mi4@polux: ~/Images/Photo_weekend_rando

Fichier Édition Onglets Aide

```
mi4@polux:~$ find / -user mi4 -size +10M 2>/dev/null
/home/mi4/Vidéos/Film.mov
mi4@polux:~$ cd Images/Photo_weekend_rando/
mi4@polux:~/Images/Photo_weekend_rando$ ls
photo1.jpg photo3.jpg photo5.jpg photo7.jpg photo9.jpg
photo2.jpg photo4.jpg photo6.jpg photo8.jpg
mi4@polux:~/Images/Photo_weekend_rando$ for i in *; do mv $i 2017-10-22_$i; done
mi4@polux:~/Images/Photo_weekend_rando$ ls
2017-10-22_photo1.jpg 2017-10-22_photo4.jpg 2017-10-22_photo7.jpg
2017-10-22_photo2.jpg 2017-10-22_photo5.jpg 2017-10-22_photo8.jpg
2017-10-22_photo3.jpg 2017-10-22_photo6.jpg 2017-10-22_photo9.jpg
mi4@polux:~/Images/Photo_weekend_rando$ last | grep -v reboot | grep -e Fri -e Thu -e Wed
mi4    tty7      :0          Fri Oct 20 22:21  gone - no logout
mi4    tty7      :0          Fri Oct 20 16:09 - 17:09 (01:00)
mi4    tty8      :1          Fri Oct 20 12:10 - 12:10 (00:00)
admin  tty7      :0          Fri Oct 20 12:07 - 12:10 (00:02)
admin  tty7      :0          Fri Oct 20 12:02 - 12:05 (00:03)
admin  tty7      :0          Fri Oct 20 11:59 - 12:02 (00:02)
admin  tty7      :0          Fri Oct 20 11:56 - 11:59 (00:02)
admin  tty7      :0          Fri Oct 20 11:48 - 11:54 (00:05)
mi4    tty7      :0          Fri Oct 20 11:47 - 11:48 (00:01)
root   tty7      :0          Fri Oct 20 11:46 - 11:47 (00:00)
mi4    tty7      :0          Fri Oct 20 11:44 - 11:46 (00:01)
mi4    tty1      :0          Fri Oct 20 11:44 - 11:50 (00:06)
mi4@polux:~/Images/Photo_weekend_rando$
```



Le terminal de commandes



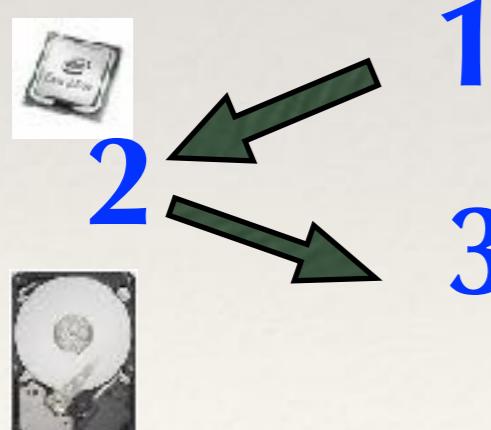
A screenshot of a terminal window titled "mi4@polux: ~/Images/Photo_weekend_rando". The window includes standard Linux window controls (minimize, maximize, close) at the top right. The menu bar contains "Fichier", "Édition", "Onglets", and "Aide". The terminal displays the following command-line session:

```
mi4@polux:~$ find / -user mi4 -size +10M 2>/dev/null  
/home/mi4/Vidéos/Film.mov  
mi4@polux:~$ cd Images/Photo_weekend_rando/  
mi4@polux:~/Images/Photo_weekend_rando$ ls  
photo1.jpg photo3.jpg photo5.jpg photo7.jpg photo9.jpg  
photo2.jpg photo4.jpg photo6.jpg photo8.jpg  
mi4@polux:~/Images/Photo_weekend_rando$ for i in *; do mv $i 2017-10-22_$i; done  
mi4@polux:~/Images/Photo_weekend_rando$ ls  
2017-10-22_photo1.jpg 2017-10-22_photo4.jpg 2017-10-22_photo7.jpg  
2017-10-22_photo2.jpg 2017-10-22_photo5.jpg 2017-10-22_photo8.jpg  
2017-10-22_photo3.jpg 2017-10-22_photo6.jpg 2017-10-22_photo9.jpg  
mi4@polux:~/Images/Photo_weekend_rando$ last | grep -v reboot | grep -e Fri -e Thu -e Wed  
mi4      tty7        :0          Fri Oct 20 22:21    gone - no logout  
mi4      tty7        :0          Fri Oct 20 16:09 - 17:09  (01:00)  
mi4      tty8        :1          Fri Oct 20 12:10 - 12:10  (00:00)  
admin    tty7        :0          Fri Oct 20 12:07 - 12:10  (00:02)  
admin    tty7        :0          Fri Oct 20 12:02 - 12:05  (00:03)  
admin    tty7        :0          Fri Oct 20 11:59 - 12:02  (00:02)  
admin    tty7        :0          Fri Oct 20 11:56 - 11:59  (00:02)  
admin    tty7        :0          Fri Oct 20 11:48 - 11:54  (00:05)  
mi4      tty7        :0          Fri Oct 20 11:47 - 11:48  (00:01)  
root    tty7        :0          Fri Oct 20 11:46 - 11:47  (00:00)  
mi4      tty7        :0          Fri Oct 20 11:44 - 11:46  (00:01)  
mi4      tty1        :0          Fri Oct 20 11:44 - 11:50  (00:06)  
mi4@polux:~/Images/Photo_weekend_rando$
```

Le terminal de commandes

- ❖ Un terminal de commande exécute une boucle infinie composée de trois actions dans l'ordre :
 1. lire une commande de l'utilisateur
 2. exécuter la commande
 3. indiquer le résultat de la commande

invite de commandes →



```
mi4@polux: ~/Python
Fichier Édition Onglets Aide
mi4@polux:~$ pwd
/home/mi4
mi4@polux:~$ ls -l
total 36
drwxr-xr-x 2 mi4 mi4 4096 oct. 17 14:18 Bureau
drwxr-xr-x 3 mi4 mi4 4096 oct. 17 13:52 Documents
drwxr-xr-x 2 mi4 mi4 4096 oct. 17 13:34 Images
drwxr-xr-x 2 mi4 mi4 4096 oct. 17 13:34 Modèles
drwxr-xr-x 2 mi4 mi4 4096 oct. 17 13:34 Musique
drwxr-xr-x 2 mi4 mi4 4096 oct. 17 13:34 Public
drwxrwxr-x 2 mi4 mi4 4096 oct. 17 14:14 Python
drwxr-xr-x 2 mi4 mi4 4096 oct. 17 13:34 Téléchargements
drwxr-xr-x 2 mi4 mi4 4096 oct. 17 13:34 Vidéos
mi4@polux:~$ cd Python/
mi4@polux:~/Python$
```

Le terminal de commandes

- ❖ Format des commandes :

nomCde [-option(s)] [argument(s)]



*espaces
entre chaque option,
entre chaque argument !*

- ❖ Exemples :

date

whoami

affiche le nom de l'utilisateur connecté

pwd

affiche le nom du répertoire courant

ls -l

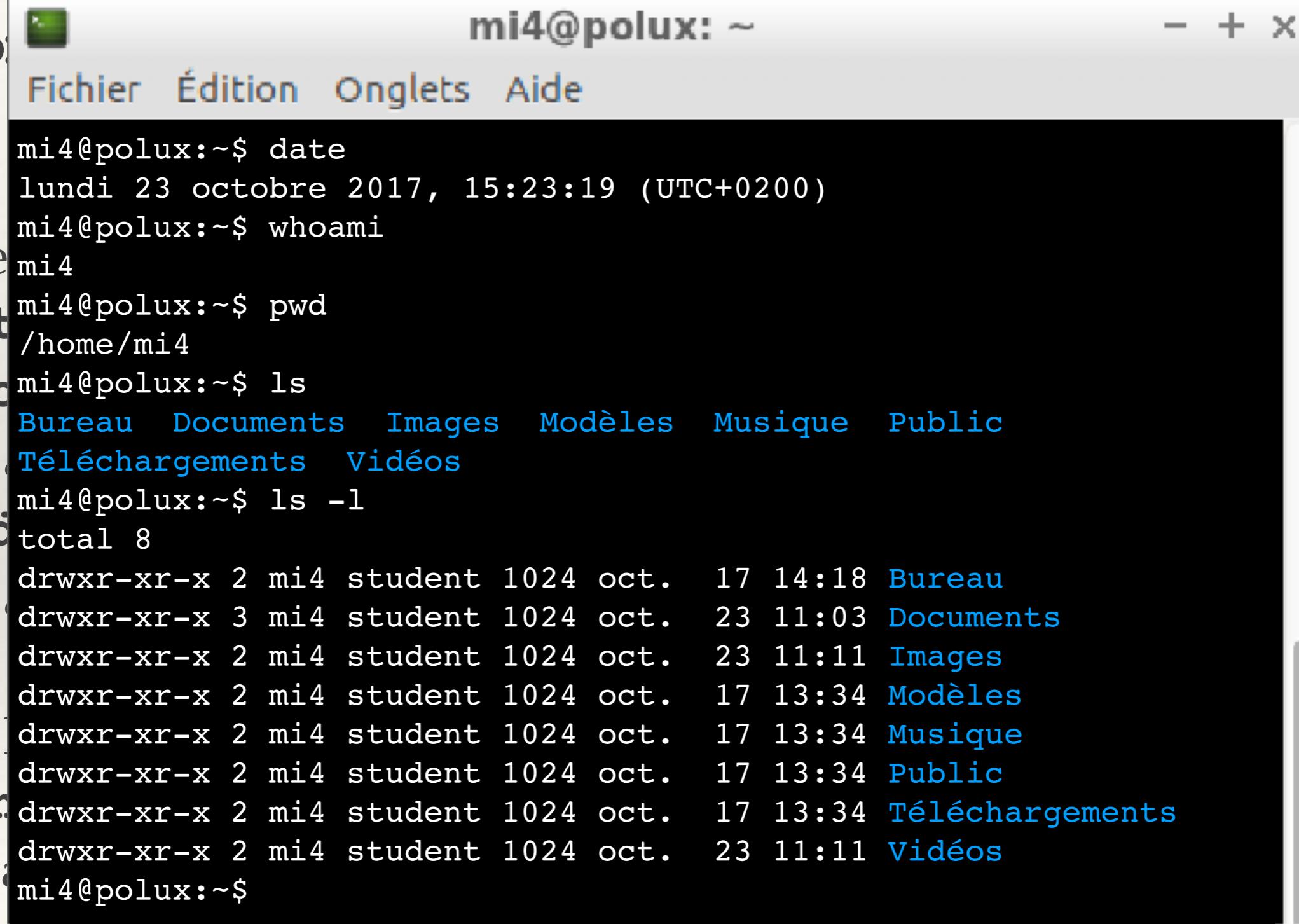
liste le contenu d'un répertoire **de façon détaillée**

man <cmd>

accès au manuel de <cmd>

Le terminal de commandes

❖ Format des commandes :



The screenshot shows a terminal window with the following command history:

```
mi4@polux:~$ date
lundi 23 octobre 2017, 15:23:19 (UTC+0200)
mi4@polux:~$ whoami
mi4
mi4@polux:~$ pwd
/home/mi4
mi4@polux:~$ ls
Bureau Documents Images Modèles Musique Public
Téléchargements Vidéos
mi4@polux:~$ ls -l
total 8
drwxr-xr-x 2 mi4 student 1024 oct. 17 14:18 Bureau
drwxr-xr-x 3 mi4 student 1024 oct. 23 11:03 Documents
drwxr-xr-x 2 mi4 student 1024 oct. 23 11:11 Images
drwxr-xr-x 2 mi4 student 1024 oct. 17 13:34 Modèles
drwxr-xr-x 2 mi4 student 1024 oct. 17 13:34 Musique
drwxr-xr-x 2 mi4 student 1024 oct. 17 13:34 Public
drwxr-xr-x 2 mi4 student 1024 oct. 17 13:34 Téléchargements
drwxr-xr-x 2 mi4 student 1024 oct. 23 11:11 Vidéos
mi4@polux:~$
```

A red annotation box is positioned in the top right corner of the terminal window. It contains the text "espaces entre chaque ent." in red, which corresponds to the spaces between each argument in a command line. A larger red arrow points from this text towards the command line "ls -l". Another red arrow points from the text "option, argument!" towards the command line "ls -l".

Téléchargements Vidéos

```
mi4@polux:~$ ls -l
total 8
drwxr-xr-x 2 mi4 student 1024 oct. 17 14:18 Bureau
drwxr-xr-x 3 mi4 student 1024 oct. 23 11:03 Documents
drwxr-xr-x 2 mi4 student 1024 oct. 23 11:11 Images
drwxr-xr-x 2 mi4 student 1024 oct. 17 13:34 Modèles
drwxr-xr-x 2 mi4 student 1024 oct. 17 13:34 Musique
drwxr-xr-x 2 mi4 student 1024 oct. 17 13:34 Public
drwxr-xr-x 2 mi4 student 1024 oct. 17 13:34 Téléchargements
drwxr-xr-x 2 mi4 student 1024 oct. 23 11:11 Vidéos
```

```
mi4@polux:~$ mkdir TP_Unix
```

```
mi4@polux:~$ ls -l
```

```
total 8
```

```
drwxr-xr-x 2 mi4 student 1024 oct. 17 14:18 Bureau
drwxr-xr-x 3 mi4 student 1024 oct. 23 11:03 Documents
drwxr-xr-x 2 mi4 student 1024 oct. 23 11:11 Images
drwxr-xr-x 2 mi4 student 1024 oct. 17 13:34 Modèles
drwxr-xr-x 2 mi4 student 1024 oct. 17 13:34 Musique
drwxr-xr-x 2 mi4 student 1024 oct. 17 13:34 Public
drwxr-xr-x 2 mi4 student 1024 oct. 17 13:34 Téléchargements
drwxr-xr-x 2 mi4 student 1024 nov. 08 13:15 TP_Unix
drwxr-xr-x 2 mi4 student 1024 oct. 23 11:11 Vidéos
```

```
mi4@polux:~$ cd TP_Unix
```

```
mi4@polux:~/TP_Unix$ ls -l
```

```
mi4@polux:~/TP_Unix$ cd /home/mi4
```

```
mi4@polux:~$ ls
```

```
Bureau Documents Images Modèles Musique Public
```

```
Téléchargements TP_Unix Vidéos
```

YUE option,
argument !

❖ Fonctions

❖ Exercices

dat

who

pwd

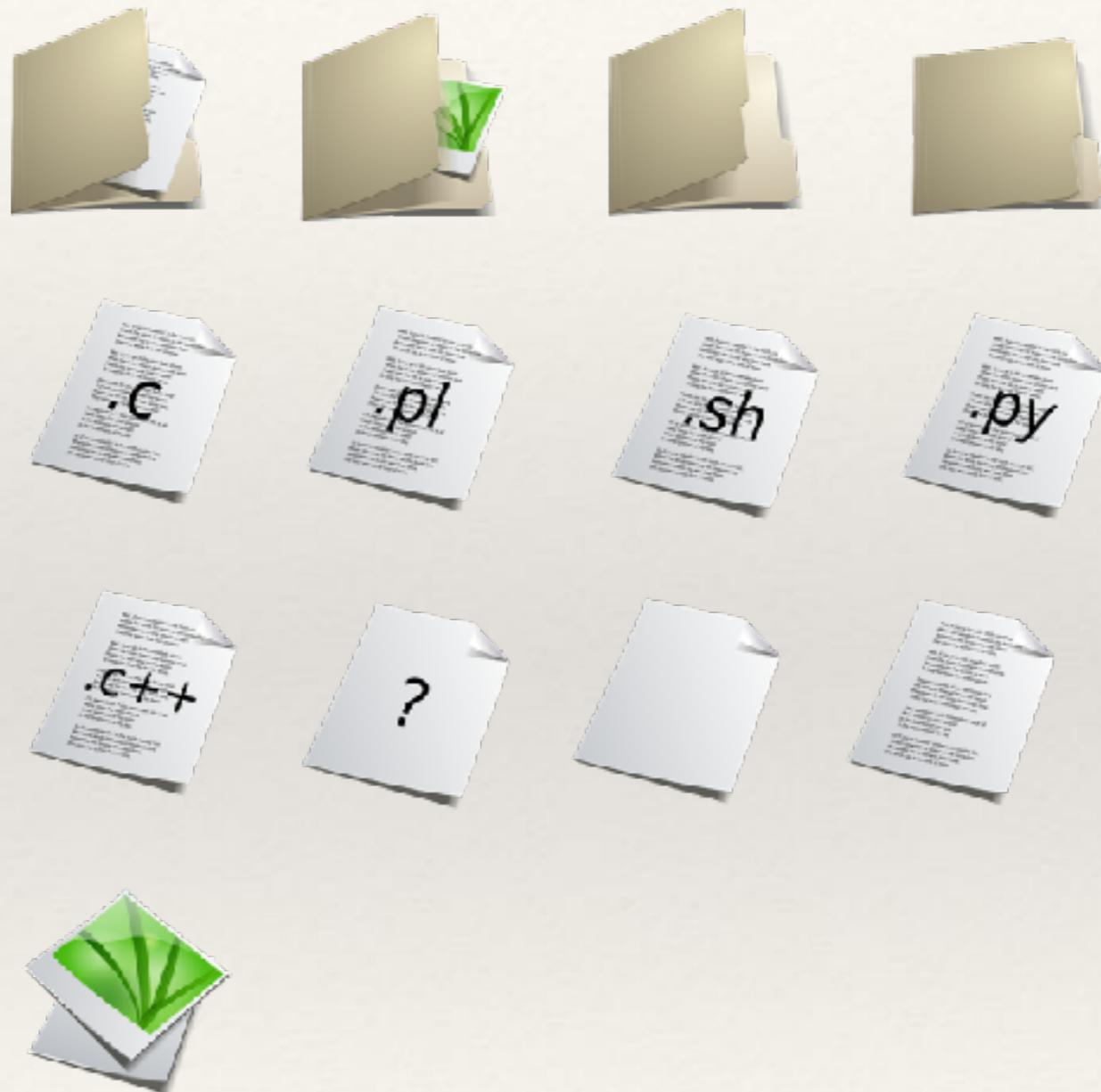
ls

man

?

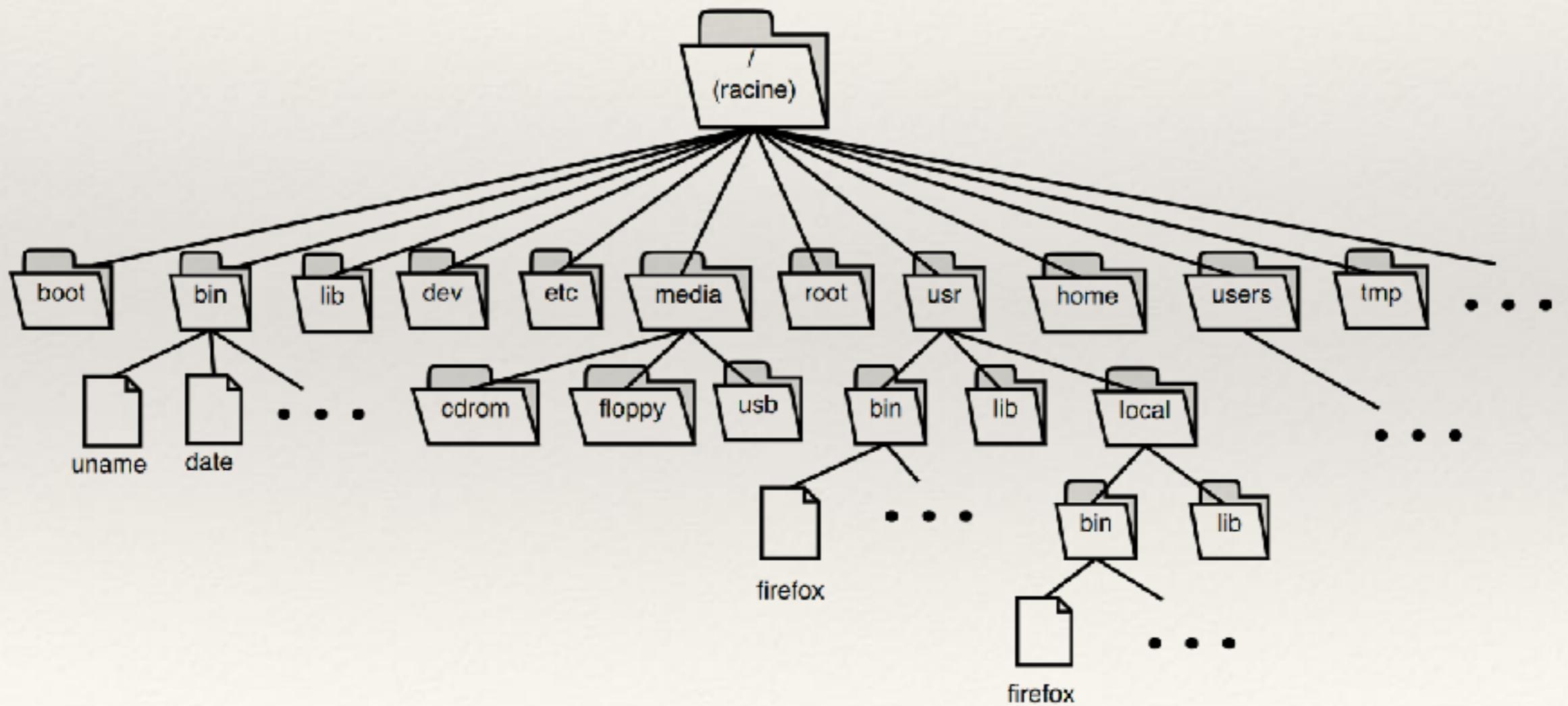
TP : Exos 1 et 2
(15 minutes)

Manipulation de fichiers et dossiers



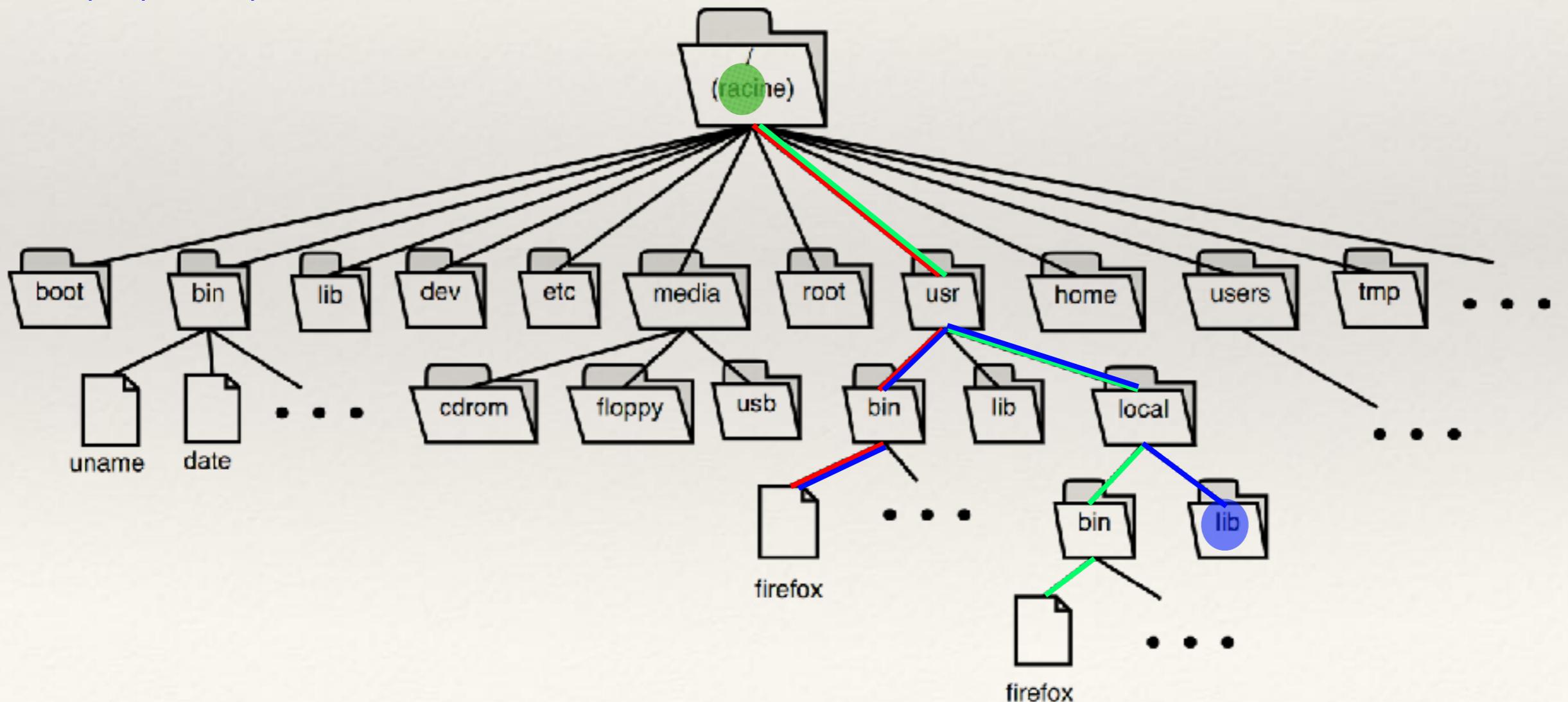
Organisation hiérarchique

- ❖ Les répertoires & fichiers sont organisés sous la forme d'une arborescence :



Désignation d'un fichier/dossier

- ❖ Chemin **absolu** depuis la racine (notée « / »)
`/usr/bin/firefox` `/usr/local/bin/firefox`
- ❖ Chemin **relatif** depuis le répertoire courant :
`../../bin/firefox`



Commandes sur les répertoires

Raccourcis :

le répertoire d'accueil : ~

le répertoire courant : .

le répertoire supérieur : ..

Commandes :

connaître le rép. courant : **pwd**

changer de répertoire : **cd <dir>**

lister le contenu d'un rép. : **ls**

lister avec détails : **ls -l**

lister les fichiers cachés : **ls -a**

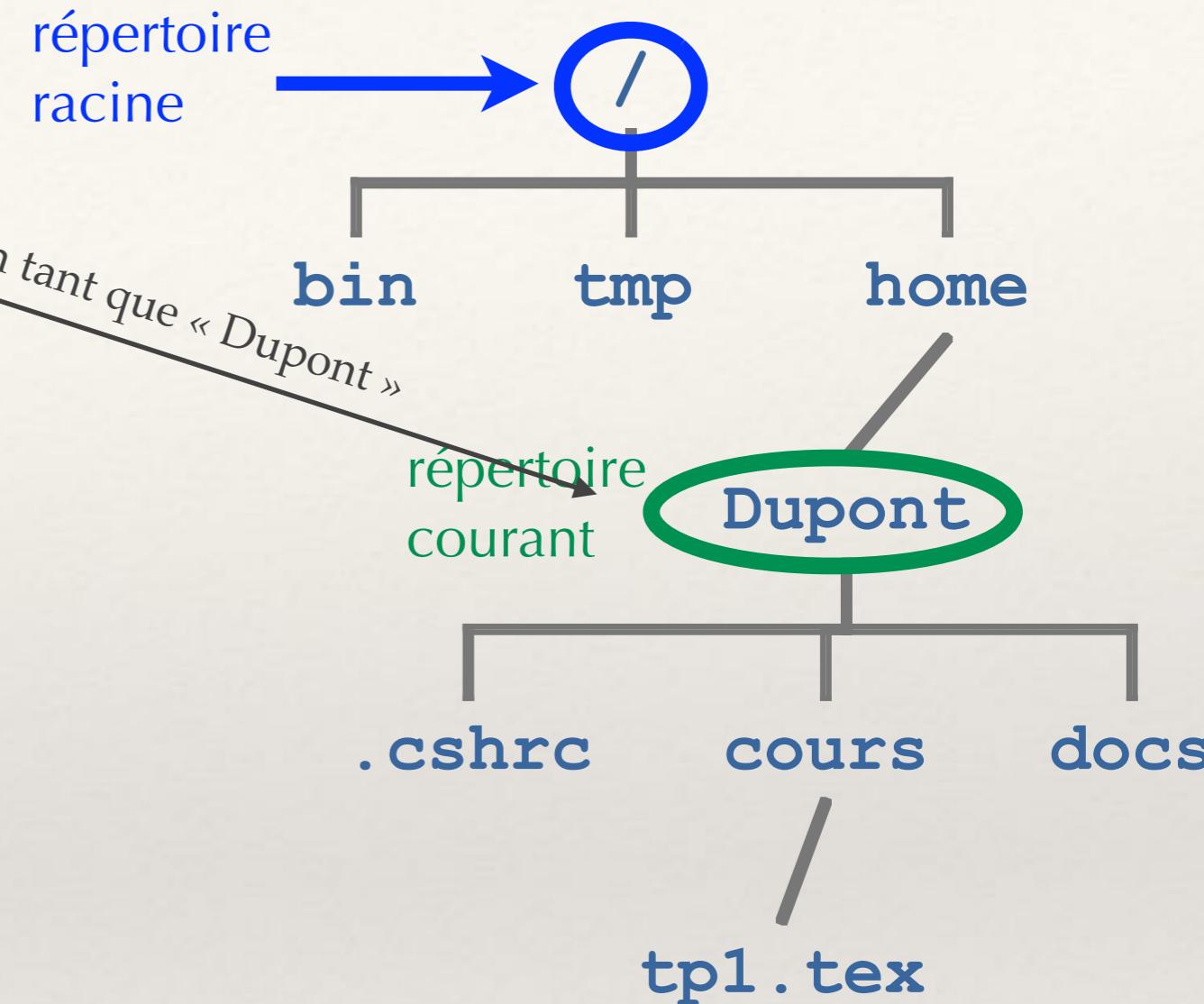
lister par date de modif. : **ls -t**

création d'un rép. : **mkdir**

Chemin d'accès au fichier **tp1.tex**:

relatif au rép. courant : **./cours/tp1.tex**

absolu : **/home/Dupont/cours/tp1.tex**



- Quelles commandes exécuter pour :
1. vous placer à la racine,
 2. descendre dans /usr/bin en 2 étapes,
 3. remonter à la racine.

Commandes sur les fichiers/dossiers

Copier un fichier	cp <code>fic1 fic2</code>
Copier un répertoire avec son contenu	cp <code>-r rep1 rep2</code>
Déplacer	mv <code>fic1 ../Image/fic1</code>
Renommer	mv <code>fic1 fic2</code>
Déplacer et renommer	mv <code>fic1 ../Image/fic2</code>
Effacer un fichier	rm <code>fic</code>
Effacer un dossier	rm <code>-r dir</code>
Afficher le contenu d'un fichier	more <code>fic</code> less <code>fic</code>
Concatener plusieurs fichiers	cat <code>fic1 fic2 ...</code>
Modifier l'heure de dernière modif. du fichier (crée le fichier s'il n'existe pas)	touch <code>fic</code>

Méta-caractères du shell

Ce sont des caractères qui ont un sens spécial :

@ + ~ ! ^ * ? [] () \ ; & < > | " ' . \$ espace

► L'**espace** sert à séparer le **nom de commande** de ses **options** et **arguments**

► L'astérisque ou étoile: *

► interprété comme toute suite de caractères alphanumériques

► utiliser avec **précaution** (commande rm par ex...)

```
mi4@polux:~$ rm img/*.jpg
```

► Le point d'interrogation: ?

► remplace un seul caractère alphanumérique

```
mi4@polux:~$ mv photo?.jpg img/
```

► Le point-virgule: ;

► Séparateur de commandes

```
mi4@polux:~$ cd img ; ls
```

► L'anti-slash: \

► Inhibe la signification du méta-caractère qui suit

```
mi4@polux:~$ rm Metallica\Sonate\ 9\ si\ majeur.mp3
```

\ suivi de **espace**

► Texte entre ' ' (simples **quotes**) : le texte n'est pas interprété par le shell
(peut contenir des caractères spéciaux), considéré comme **un seul mot**

Méta-caractères du shell

Ce sont des caractères qui ont un sens spécial :

@ + ~ ! ^ * ? [] () \ ; & < > | " ' . \$ espace

► L'**espace** sert à séparer le **nom de commande** de ses **options** et **arguments**

► L'astérisque ou étoile: *

- interprété comme toute suite de caractères alphanumériques
- utiliser avec **précaution** (commande rm par ex...)

```
mi4@polux:~$ rm img/*.jpg
```

► Le point d'interrogation: ?

- remplace un seul caractère alphanumérique

```
mi4@polux:~$ mv photo?.jpg img/
```

► Le point-virgule: ;

- Séparateur de commandes

```
mi4@polux:~$ cd img ; ls
```

► L'anti-slash: \

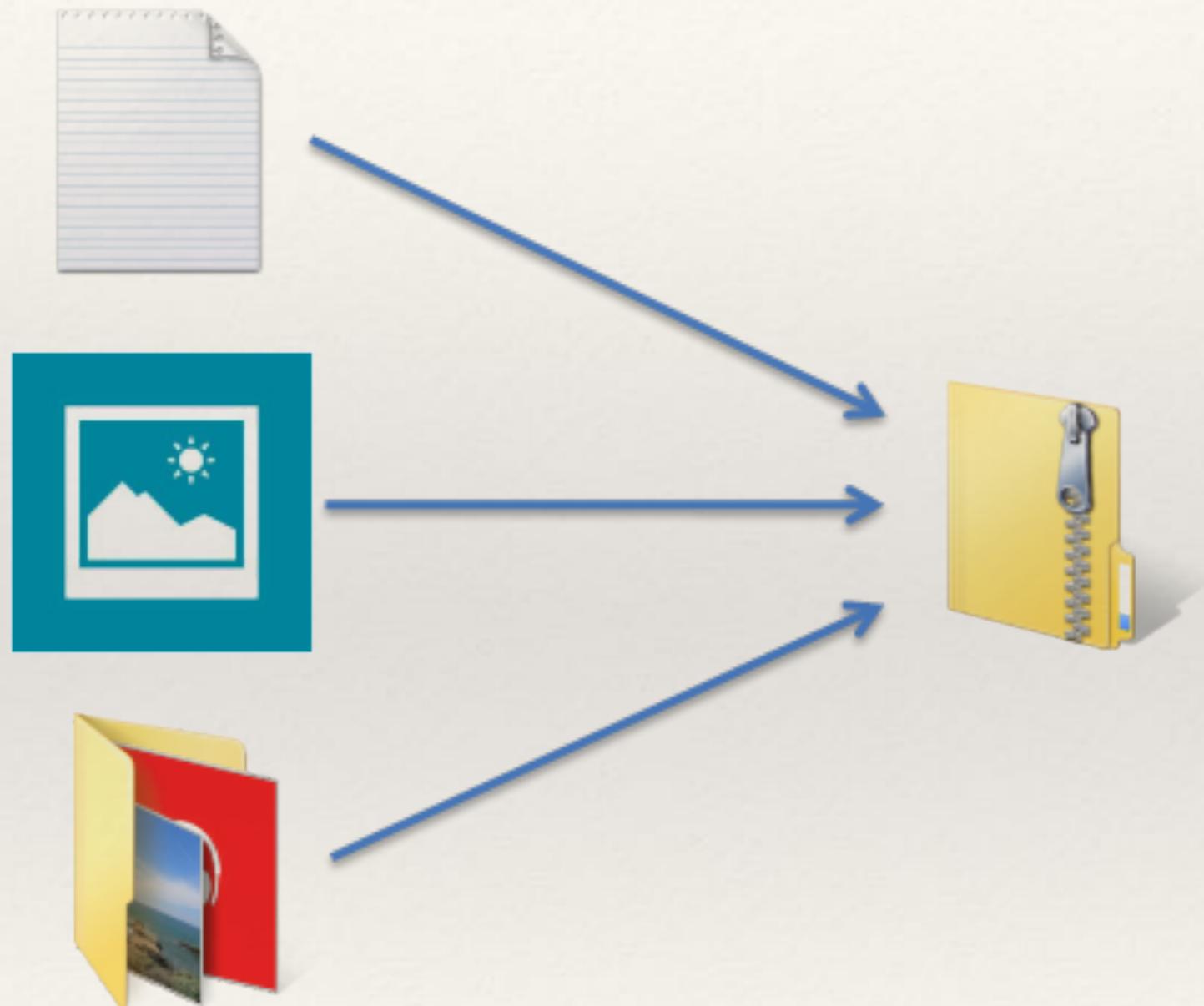
- Inhibe la signification du méta-caractère qui suit

```
mi4@polux:~$ rm 'Metallica Sonate 9 si majeur.mp3'
```

\ suivi de **espace**

► Texte entre ' ' (simples **quotes**) : le texte n'est pas interprété par le shell
(peut contenir des caractères spéciaux), considéré comme **un seul mot**

Manipulation d'archives



Les outils d'archivage et de compression UNIX

- ❖ On distingue généralement l'**assemblage** d'un ensemble de fichiers / répertoires en un seul fichier (**archivage**) ...
 - ▶ commande **tar**
- ❖ ... et la phase de **compression** de l'archive :
 - ▶ commandes **zip**, **gzip**, **bzip2**, ...

les fichiers désarchivés appartiennent à celui qui les désarchive, peu importe l'endroit (répertoire) ou le créateur de l'archive.

La commande **tar** :

```
tar [c|t|x][z][v]f fichArchive.tar <fic1> <fic2> <rep1> <rep2> ...
```

c	(create) crée une nouvelle archive (efface d'éventuels fichiers / répertoires présents dans l'archive auparavant)
t	(list t) affiche le contenu de l'archive
x	(extract) extrait le(s) fichier(s) désigné(s) de l'archive et le(s) restaure en local sur le système. Désigner un répertoire par son nom aboutit à désarchiver son contenu récursivement.
z	(zip) pour une archive compressée
v	(verbose) affiche des informations lors de la manipulation de l'archive
f device	(regular file) tar utiliser l'argument device comme nom d'archive

Les outils d'archivage et de compression UNIX

- ❖ On distingue généralement l'**assemblage** d'un ensemble de fichiers / répertoires en un seul fichier (**archivage**) ...
 - ▶ commande **tar**
- ❖ ... et la phase de **compression** de l'archive :
 - ▶ commandes **zip**, **gzip**, **bzip2**, ...

les fichiers désarchivés appartiennent à celui qui les désarchive, peu importe l'endroit (répertoire) ou le créateur de l'archive.

La commande **tar** :

```
tar [c|t|x][z][v]f fichArchive.tar <fic1> <fic2> <rep1> <rep2> ...
```

```
mi4@polux:~$ ls -l
-rw-r--r-- 1 mi4 student 4689 oct. 23 14:35 fichier1.txt
-rw-r--r-- 1 mi4 student 8687 oct. 22 09:38 fichier2.txt
-rw-r--r-- 1 mi4 student 389 oct. 24 18:09 fichier3.txt
mi4@polux:~$ tar cvf archive.tar fichier*.txt
fichier1.txt
fichier2.txt
fichier3.txt
mi4@polux:~$ ls -l
-rw-r--r-- 1 mi4 student 14265 oct. 25 09:02 archive.tar
-rw-r--r-- 1 mi4 student 4689 oct. 23 14:35 fichier1.txt
-rw-r--r-- 1 mi4 student 8687 oct. 22 09:38 fichier2.txt
-rw-r--r-- 1 mi4 student 389 oct. 24 18:09 fichier3.txt
```

Les outils d'archivage et de compression UNIX

- ❖ On distingue généralement l'**assemblage** d'un ensemble de fichiers / répertoires en un seul fichier (**archivage**) ...
 - ▶ commande **tar**
- ❖ ... et la phase de **compression** de l'archive :
 - ▶ commandes **zip**, **gzip**, **bzip2**, ...

les fichiers désarchivés appartiennent à celui qui les désarchive, peu importe l'endroit (répertoire) ou le créateur de l'archive.

La commande **tar** :

```
tar [c|t|x][z][v]f fichArchive.tar <fic1> <fic2> <rep1> <rep2> ...
```

```
polytech@polux:~$ ls -l
-rw-r--r-- 1 polytech student 4689 oct. 23 14:35 archive.tar
polytech@polux:~$ tar xvf archive.tar
fichier1.txt
fichier2.txt
fichier3.txt
polytech@polux:~$ ls -l
-rw-r--r-- 1 polytech student 4689 oct. 26 17:35 archive.tar
-rw-r--r-- 1 polytech student 4689 oct. 23 14:35 fichier1.txt
-rw-r--r-- 1 polytech student 8687 oct. 22 09:38 fichier2.txt
-rw-r--r-- 1 polytech student 389 oct. 24 18:09 fichier3.txt
polytech@polux:~$
```

Les outils d'archivage et de compression UNIX

- ❖ On distingue généralement l'**assemblage** d'un ensemble de fichiers / répertoires en un seul fichier (**archivage**) ...
 - ▶ commande **tar**
- ❖ ... et la phase de **compression** de l'archive :
 - ▶ commandes **zip**, **gzip**, **bzip2**, ...

La commande **gzip** :

gzip [-d] [-l] [-r] fichier(s)

-d	décomprime le fichier (mais ne désassemble pas si c'est une archive)
-l	(list) donne une liste d'informations sur chaque fichier (taille avant compression, taille après compression, nom des fichiers, ...)
-r	(recursive) procède récursivement, i.e. traite (séparément) tous les fichiers trouvés dans la sous-arborescence

```
mi4@polux:~$ ls -l
-rw-r--r-- 1 mi4 student 5629 oct. 23 15:35 archive.tar
mi4@polux:~$ gzip archive.tar
mi4@polux:~$ ls -l
-rw-r--r-- 1 mi4 student 328 oct. 23 16:45 archive.tar.gz
```

Les outils d'archivage et de compression UNIX

- ❖ On distingue généralement l'**assemblage** d'un ensemble de fichiers / répertoires en un seul fichier (**archivage**) ...
 - ▶ commande **tar**
- ❖ ... et la phase de **compression** de l'archive :
 - ▶ commandes **zip**, **gzip**, **bzip2**, ...

La commande **gzip** :

gzip [-d] [-l] [-r] fichier(s)

-d	décomprime le fichier (mais ne désassemble pas si c'est une archive)
-l	(list) donne une liste d'informations sur chaque fichier (taille avant compression, taille après compression, nom des fichiers, ...)
-r	(recursive) procède récursivement, i.e. traite (séparément) tous les fichiers trouvés dans la sous-arborescence

```
mi4@polux:~$ gzip -d archive.tar.gz
mi4@polux:~$ ls -l
-rw-r--r-- 1 mi4 student 328 oct. 24 09:32 archive.tar
-rw-r--r-- 1 mi4 student 328 oct. 23 16:45 archive.tar.gz
```

Gestionnaire de paquets



**DEBIAN / UBUNTU
PACKAGE MANAGEMENT**

Installation de nouvelles applications

- ❖ Installation centralisée par la « logithèque » de la distribution Linux
- ❖ Accessible en ligne de commandes par la commande **apt**



→ Gestion des dépendances !

- ❖ Nécessite les droits de super-utilisateur (**root**) :

sudo apt list

sudo apt install <paquet>

sudo apt search <mot-clé>

sudo apt remove <paquet>

- ❖ Commande **dpkg** : pour installation / création / suppression d'un seul paquet (pas de gestion des dépendances).

- ❖ Installation par récupération d'une archive (**.tar.gz**) suivie souvent des commandes

./configure ; make ; make install

TP : Exos 3, 4 & 5
(25 minutes)

Protection des informations

- ❖ Gérer quel utilisateur accède à quels fichiers / dossiers.
- ❖ L'ensemble des opérations autorisées sur un objet constitue ses **droits d'accès**.
- ❖ Les droits peuvent évoluer dans le temps.
Ex : un projet en cours / fini.



Les droits d'accès

- ❖ Tout fichier ou répertoire a un **propriétaire** et un **groupe** par défaut, le créateur du fichier et le groupe auquel appartient le créateur du fichier
- ❖ Les droits d'accès à un fichier (ou répertoire) sont paramétrables par le propriétaire du fichier pour trois classes d'utilisateurs (**ugo**)
 - ▶ lui-même = le propriétaire du fichier (**user**) 
 - ▶ les membres du groupe (**group**) 
 - ▶ les autres utilisateurs du système (**others**) 

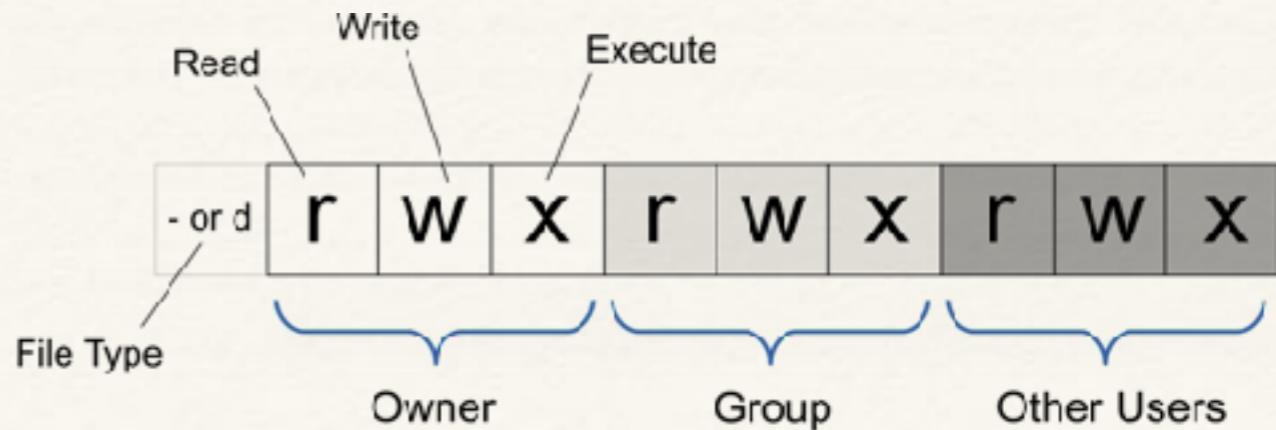
Les droits d'accès

3 types d'accès sont possibles (rwx) pour chacune des 3 classes d'utilisateurs.

Significations différentes fichiers / répertoires.

- ▶ L'accès en lecture (**read**)
 - permet de lire le contenu du fichier
 - permet de lister le contenu du répertoire
- ▶ L'accès en écriture (**write**)
 - permet de modifier le contenu du fichier
 - permet d'ajouter ou de supprimer des fichiers et des sous-répertoires dans le répertoire
- ▶ L'accès en exécution (**execute**)
 - permet d'exécuter le fichier (si le fichier est un programme)
 - permet de traverser le répertoire

Les droits d'accès



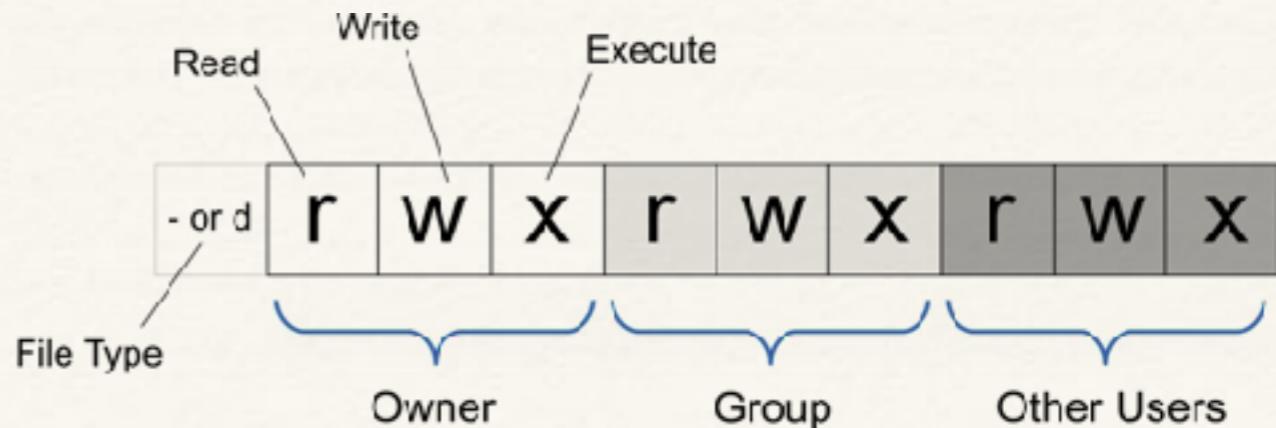
- ▶ Les droits des fichiers et répertoires peuvent être affichés par une commande Linux particulière. Laquelle ?

```
-rw-rw-r-- mi4 student 2348 oct. 10 12:34 test.py  
drwxr-xr-x mi4 student 4096 oct. 6 08:48 Documents
```

- ▶ 10 caractères sont associés à chaque **fichier** et **répertoire** :

- le 1er = nature du fichier (**d** pour répertoire et tiret (-) pour fichier simple)
- les 9 suivants = **droits** attribués à ce fichier ou répertoire :
 - ◆ 3 pour le **propriétaire** / 3 pour le **groupe** / 3 pour les **autres**)
 - ◆ Chaque groupe de 3 précise dans l'ordre : les droits en **lecture**, en **écriture**, enfin en **exécution**.
 - ◆ Pour chaque droit : si sa **lettre** apparaît ⇒ le droit est **donné** s'il y a **un tiret (-)** = le droit est **refusé**.

Les droits d'accès



Exemple :

```
mi4@polux:~$ ls -l
-rw-rw-r--  pinlou    info      502  sept  2  9:40  fic1.txt
-rwx--x--x  monnerie   mi        205  aout  3  8:03  proj.sh
drwxr-x---  marchal    mi       4096  sept  1  7:35  UNIX
mi4@polux:~$
```

- ❖ **fic1.txt** est accessible en lecture et écriture par *pinlou* et les membres du groupe *info* et en lecture pour les autres.
- ❖ **proj.sh** est accessible en lecture, écriture et exécution par *monnerie* et en exécution seulement pour tous les autres.
- ❖ **UNIX** est un répertoire listable (droit r) et traversable (droit x) par les membres du groupe *mi* ; *marchal* peut en plus le modifier (droit w) ; les autres n'ont aucun droit.

Changer les droits d'accès

```
mi4@polux:~$ chmod <classe op perm, ...> <fic> ← Notation symbolique  
mi4@polux:~$ chmod nnn <fic> ← Notation octale
```

- ❖ classe :

- ▶ u : user
- ▶ g : group
- ▶ o : others
- ▶ a : all

- ❖ op :

- ▶ = : affectation
- ▶ - : suppr.
- ▶ + : ajout

- ❖ perm :

- ▶ r : lecture
- ▶ w : écriture
- ▶ x : exécution

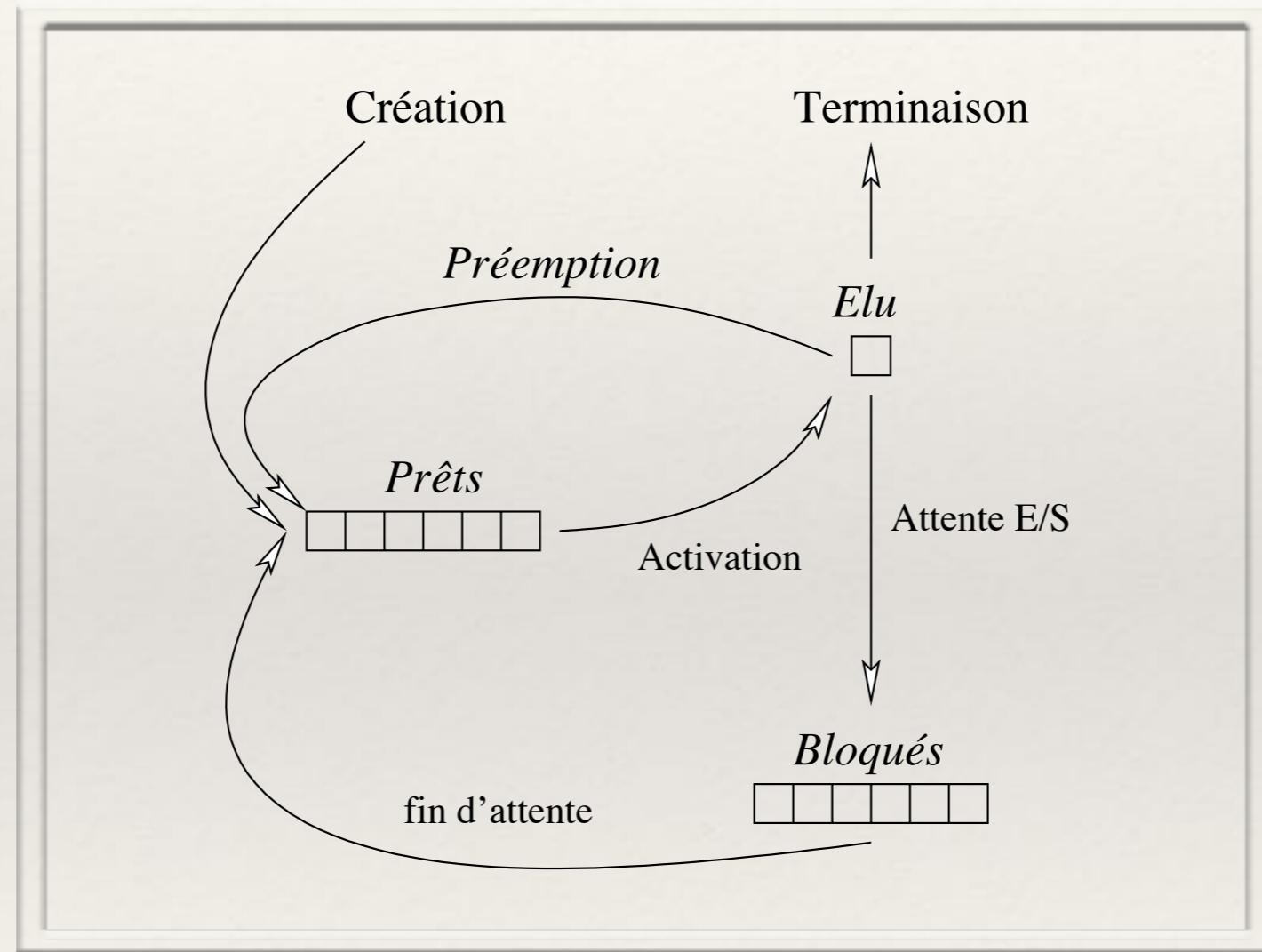
- ❖ Chaque permission = une valeur



- Sur le fichier **tp1.tex** donner tous les droits à l'utilisateur, et lecture+exécution aux autres entités :
chmod u=rwx,go=rx tp1.tex **chmod 755 tp1.tex**
- Ajouter les droits d'exécutions à toutes les entités sur le fichier **script.py** :
chmod a+x script.py

TP : exo 6
(20 minutes)

Les processus



Les processus

Un **processus** est la forme logique sous laquelle le S.E. manipule un programme en cours d'exécution (aspect dynamique).

*Équivalent aux **tâches** sous Windows*

- ❖ Chaque processus possède un **PID** (Process IDentifier), numéro **unique** attribué à chaque processus.
- ❖ Les processus UNIX sont gérés sous la forme d'une **hiérarchie**.
- ❖ Cette hiérarchie a une **racine**.
- ❖ Au démarrage de la machine, les **processus systèmes** sont créés les premiers, dont les *démons* gérants les services.
- ❖ Ensuite, ce sont les **processus utilisateurs** à partir du moment où un utilisateur s'identifie.
- ❖ De même s'il se connecte à distance depuis une autre machine.
- ❖ Quand un **processus se termine**, tous ses **descendants** aussi.

Lister les processus

```
mi4@polux:~$ ps
  UID  PID  STAT TTY      TIME      COMMAND
1001  3552  S     pts/1   0:01:04  bash
1001  3650  S     pts/1   0:00:08  emacs
1001  3654  S     ?       0:10:14  spyder3
1001  3696  S     pts/1   0:00:13  python3
1001  4820  R     pts/1   0:00:01  ps
mi4@polux:~$
```

numéro de processus
état du processus:

Pseudo terminal associé

temps CPU utilisé
commande exécutée

D	uninterruptible sleep (usually IO)
R	running or runnable (on run queue)
S	interruptible sleep (waiting for an event to complete)
T	stopped by job control signal
t	stopped by debugger during the tracing
W	paging (not valid since the 2.6.xx kernel)
X	dead (should never be seen)
Z	defunct ("zombie") process, terminated but not reaped by its parent

NOMBREUSES options disponibles → voir le man

Gestion des processus

- ❖ Tuer un processus : **kill -SIGKILL <PID>** / **kill -9 <PID>**
- ❖ Certains processus «prennent la main» :
find / -name Cours_Info -print > liste
- ❖ Arrêter le processus lancé en *avant plan* dans un terminal : **CTRL-C**
- ❖ Lancer un processus en *arrière-plan* (i.e. sans bloquer le terminal) :
ajouter « & »
- ❖ Interrompre *momentanément* un processus lancé sans «&» : **CTRL-Z**
- ❖ Relancer un processus interrompu par **CTRL-Z** :
 - ▶ en avant plan : **fg** (foreground)
 - ▶ en arrière plan : **bg** (background)

Gestion des processus

- ❖ Tuer un processus
- ❖ Certains processus sont terminés
- ❖ Arrêter le processus
- ❖ Lancer un processus
- ❖ Interrrompre un processus
- ❖ Relancer un processus
 - ▶ en avant
 - ▶ en arrière

Numéro	Nom	Description
1	SIGHUP	Instruction (HANG UP) - Fin de session
2	SIGINT	Interruption
3	SIGQUIT	Instruction (QUIT)
4	SIGILL	Instruction illégale
5	SIGTRAP	Trace trap
6	SIGABRT (ANSI)	Instruction (ABORT)
6	SIGIOT (BSD)	IOT Trap
7	SIGBUS	Bus error
8	SIGFPE	Floating-point exception - Exception arithmétique
9	SIGKILL	Instruction (KILL) - termine le processus immédiatement
10	SIGUSR1	Signal utilisateur 1
11	SIGSEGV	Violation de mémoire
12	SIGUSR2	Signal utilisateur 2
13	SIGPIPE	Broken PIPE - Erreur PIPE sans lecteur
14	SIGALRM	Alarme horloge
15	SIGTERM	Signal de terminaison
16	SIGSTKFLT	Stack Fault
17	SIGCHLD ou SIGCLD	modification du statut d'un processus fils
18	SIGCONT	Demande de reprise du processus
19	SIGSTOP	Demande de suspension imbloquable
20	SIGTSTP	Demande de suspension depuis le clavier
21	SIGTTIN	lecture terminal en arrière-plan
22	SIGTTOU	écriture terminal en arrière-plan
23	SIGURG	événement urgent sur socket
24	SIGXCPU	

-9 <PID>

ste

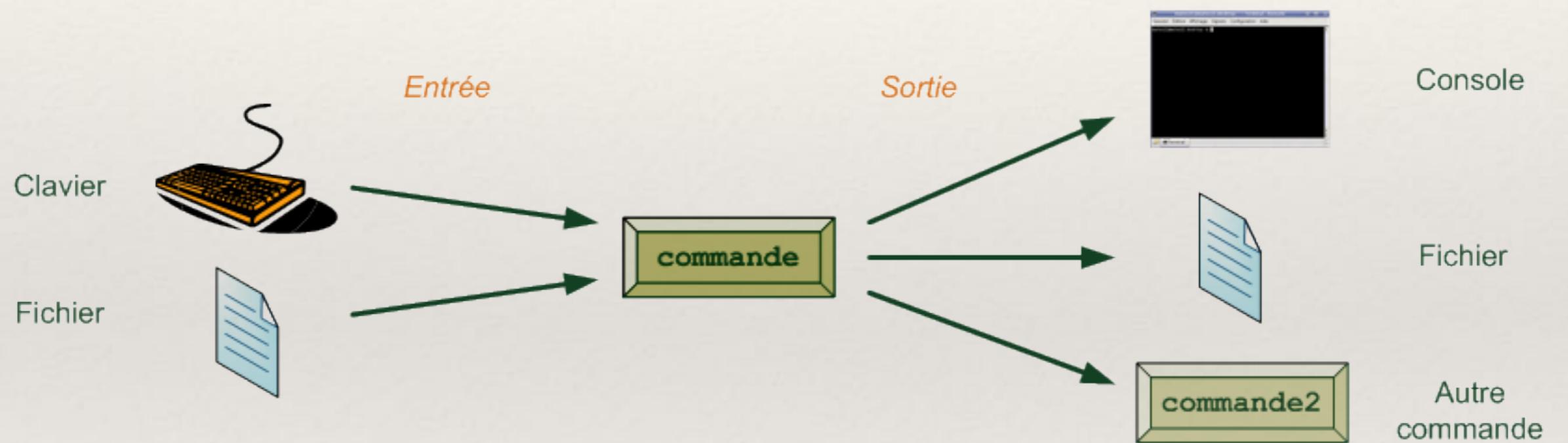
nal : CTRL-C

terminal) :

&> : CTRL-Z

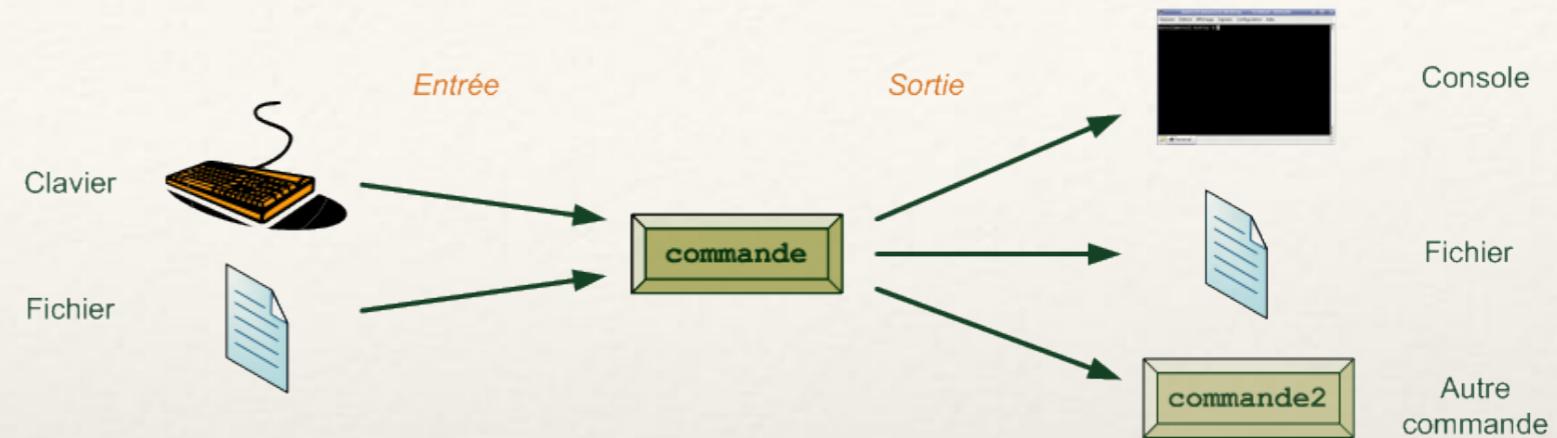
TP : exo 7
(10 minutes)

Les redirections



Les redirections / Les tubes

- ❖ Une commande utilise 3 **descripteurs de fichiers** qui sont par défaut :
 - ▶ Entrée standard
 - ▶ Sortie standard
 - ▶ Sortie erreur
- ❖ Une redirection consiste à remplacer un de ces canaux par défaut par/vers une autre commande ou un fichier.



<code>cmd < f_in</code>	redirige le fichier f_in vers l'entrée standard
<code>cmd > f_out</code>	redirige la sortie standard vers le fichier f_out
<code>cmd >> f_out</code>	redirige la sortie standard à la fin du fichier f_out
<code>cmd1 cmd2</code>	redirige la sortie standard de cmd1 vers l'entrée standard de cmd2
<code>cmd 2> f_out</code>	redirige la sortie d'erreur vers le fichier f_out
<code>cmd &> f_out</code>	redirige les sorties standard et d'erreur vers le fichier f_out

Redirections / tubes : quelques exemples

```
mi4@polux:~$ ls
Bureau Documents Images Modèles Musique Public Vidéos
mi4@polux:~$ ls > liste
mi4@polux:~$ ls
Bureau Documents Images liste Modèles Musique Public Vidéos
mi4@polux:~$ cat liste
Bureau
Documents
Images
liste
Modèles
Musique
Public
Vidéos
mi4@polux:~$ ls .. > liste
mi4@polux:~$ cat liste
lost+found
mi4
polytech
```

Vidéos

```
mi4@polux:~$ ls .. > liste
```

```
mi4@polux:~$ cat liste
```

lost+found

mi4

Polytech

```
mi4@polux:~$ ls >> liste
```

```
mi4@polux:~$ cat liste
```

lost+found

mi4

Polytech

Bureau

Documents

Images

liste

Modèles

Musique

Public

Vidéos

```
mi4@polux:~$ wc -l
```

bonjour

comment

ca va bien ?

CTRL + D

Vidéos

```
mi4@polux:~$ ls .. > liste
```

```
mi4@polux:~$ cat liste
```

lost+found

mi4

Polytech

```
mi4@polux:~$ ls >> liste
```

```
mi4@polux:~$ cat liste
```

lost+found

mi4

Polytech

Bureau

Documents

Images

liste

Modèles

Musique

Public

Vidéos

```
mi4@polux:~$ wc -l
```

bonjour

comment

ca va bien ?

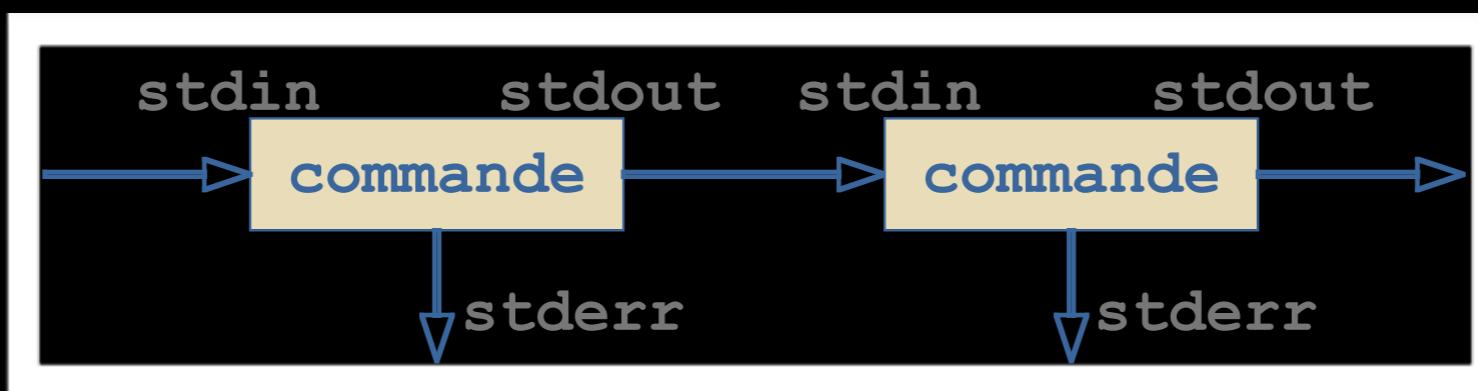
3

```
mi4@polux:~$ wc -l < liste
```

11

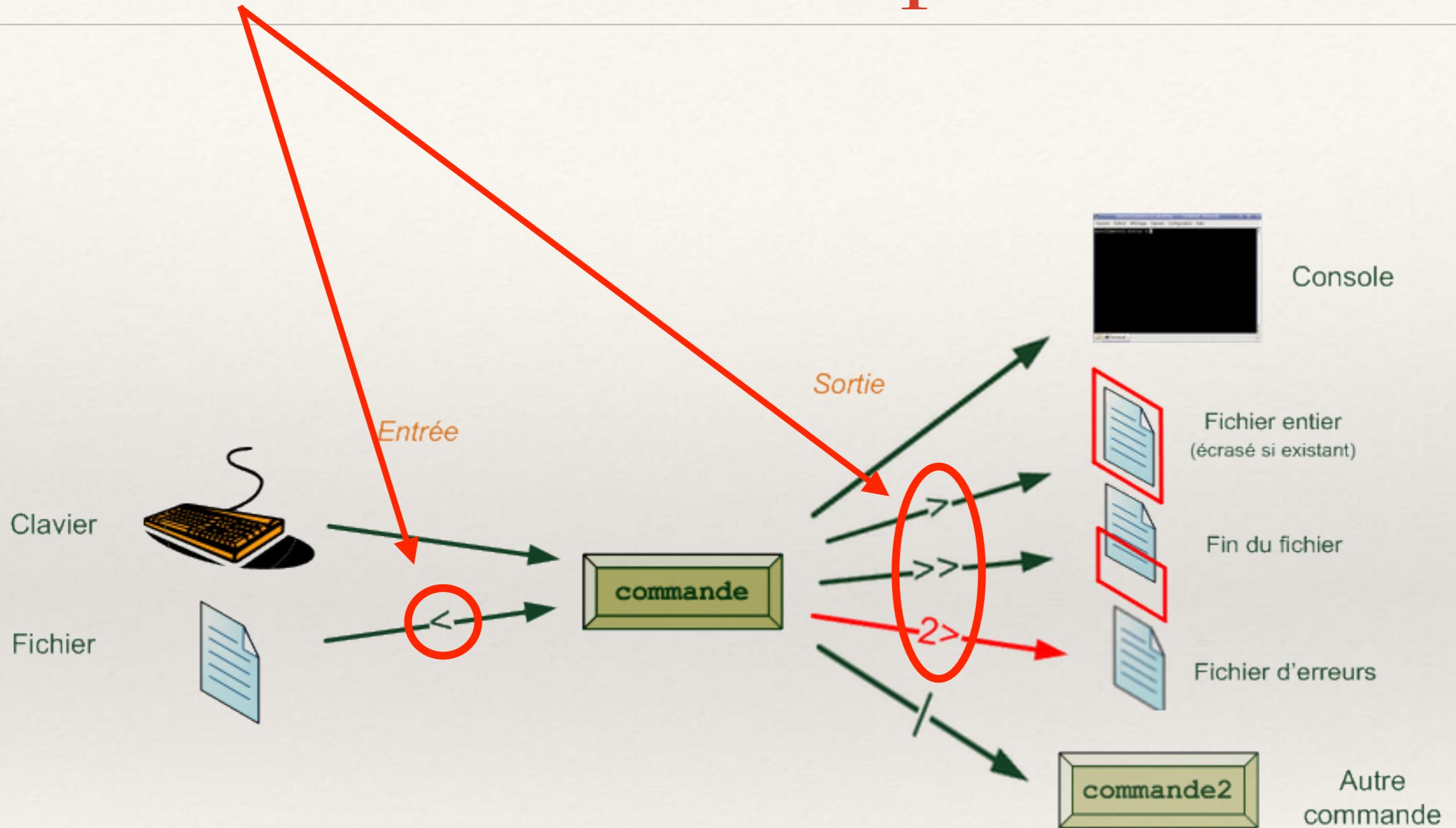
```
bonjour  
comment  
ca va bien ?  
3  
mi4@polux:~$ wc -l < liste  
11  
mi4@polux:~$ ls > liste ; wc -l < liste ; rm liste  
8  
mi4@polux:~$ ls | wc -l  
8
```

Se lit « pipe » (tube)

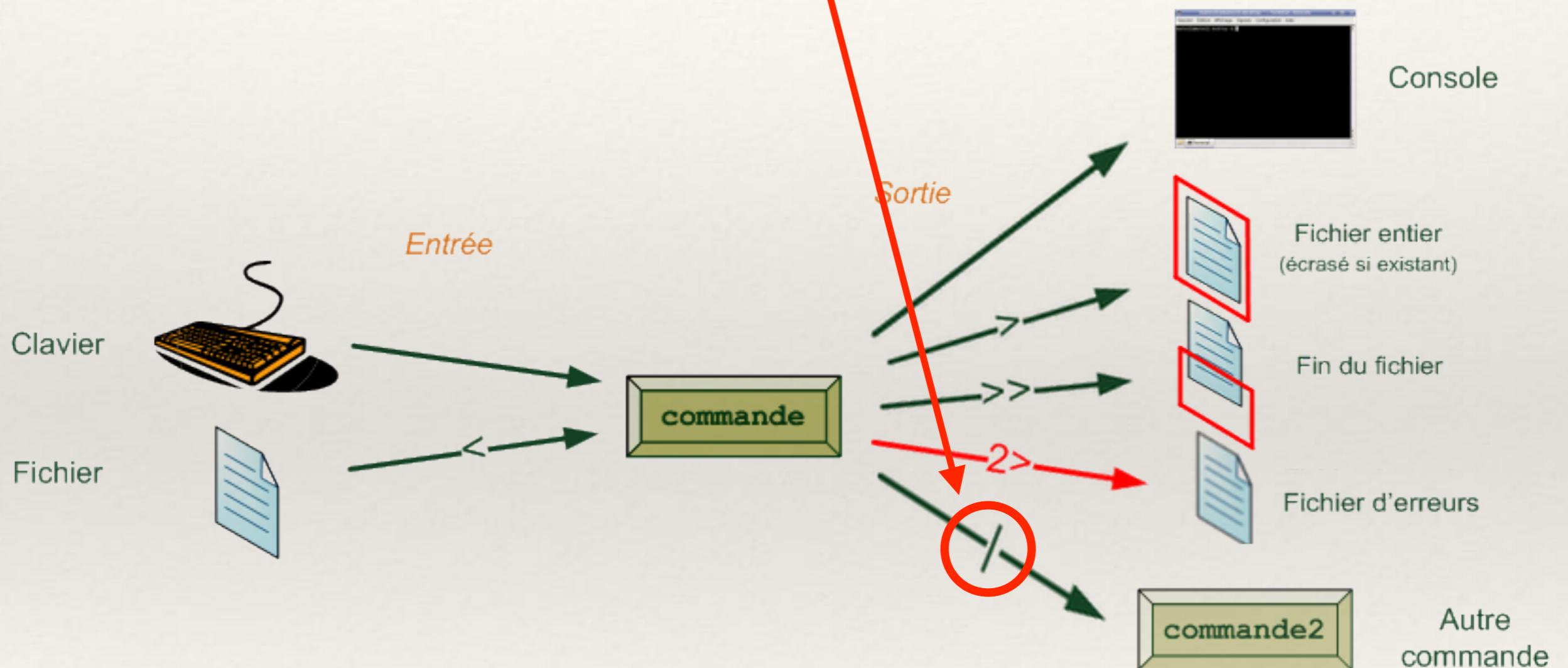


bonjour
comment
ca va bien ?
3
mi4@polux:~\$ wc -l < liste
11
mi4@polux:~\$ ls > liste ; wc -l < liste ; rm liste
8
mi4@polux:~\$ ls | wc -l
8
mi4@polux:~\$ last
mi4 tty7 :0 Mon Oct 23 10:59 gone - no logout
reboot system boot 4.10.0-19-generi Mon Oct 23 10:59
mi4 tty7 :0 Fri Oct 20 17:21 - 09:59 (2+11:38)
reboot system boot 4.10.0-19-generi Fri Oct 20 22:20 - 09:59
polytech tty7 :0 Fri Oct 20 16:09 - 17:09 (01:00)
admin tty7 :0 Fri Oct 20 12:07 - 12:10 (00:02)
mi4@polux:~\$ last | cut -d ' ' -f 1
mi4
reboot
mi4
reboot
polytech
admin
mi4@polux:~\$

Redirections / tubes : pour résumer

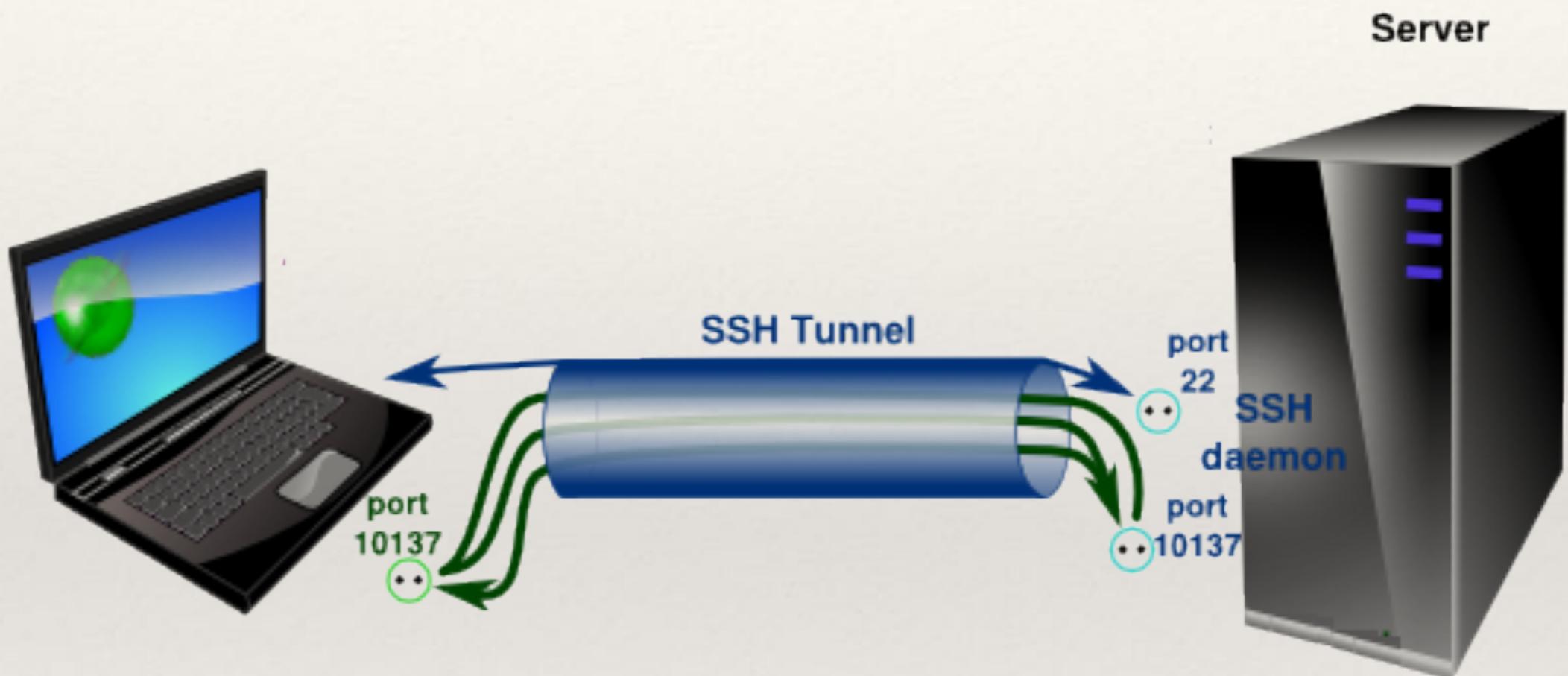


Redirections / tubes : pour résumer



TP : exo 8
(15 minutes)

Connexion à distance



Connexion à distance

- ❖ Exécution de commandes sur machine distante :
ssh pinlou@castor.isim.inra
Si on me demande un mot de passe, lequel est-ce ?
- ❖ On est alors connecté au Système de fichiers de la machine distante
- ❖ On peut ensuite exécuter des commandes sur la machine distante sans être physiquement devant.
- ❖ *Exemples* : se connecter à un serveur de calcul pour lancer des traitements coûteux en temps (intégration ou analyse de données sur un entrepôts de données), se connecter chez un hébergeur, ...

Exemple de connexion à distance

```
mi4@polux:~$ pwd
/home/mi4
mi4@polux:~$ ls
Bureau  Documents  Images  Modèles  Musique  Public  Téléchargements  Vidéos
mi4@polux:~$ ssh pinlou@castor.isim.intra
pinlou@castor.isim.intra's password:
pinlou@castor:~> pwd
/users/pinlou
pinlou@castor:~> ls -l
drwxr-xr-x 2 pinlou info 1024 oct. 17 14:18 Calculs
drwxr-xr-x 3 pinlou info 1024 oct. 23 11:03 Documents
-rwxr-xr-x 3 pinlou info 1024 sept. 09 11:03 optim.sh
pinlou@castor:~> optim.sh
5698524 824885 697320
pinlou@castor:~> exit
logout
Connection to castor.isim.intra closed.
mi4@polux:~$
```

Transfert de fichiers depuis/vers une machine distante

→ **ftp nomserveur.site.domaine**

Permet de parcourir les fichiers distants

Commandes : **cd, put, get, delete, ...**

→ **scp <fichierSource> <fichierDestination>**

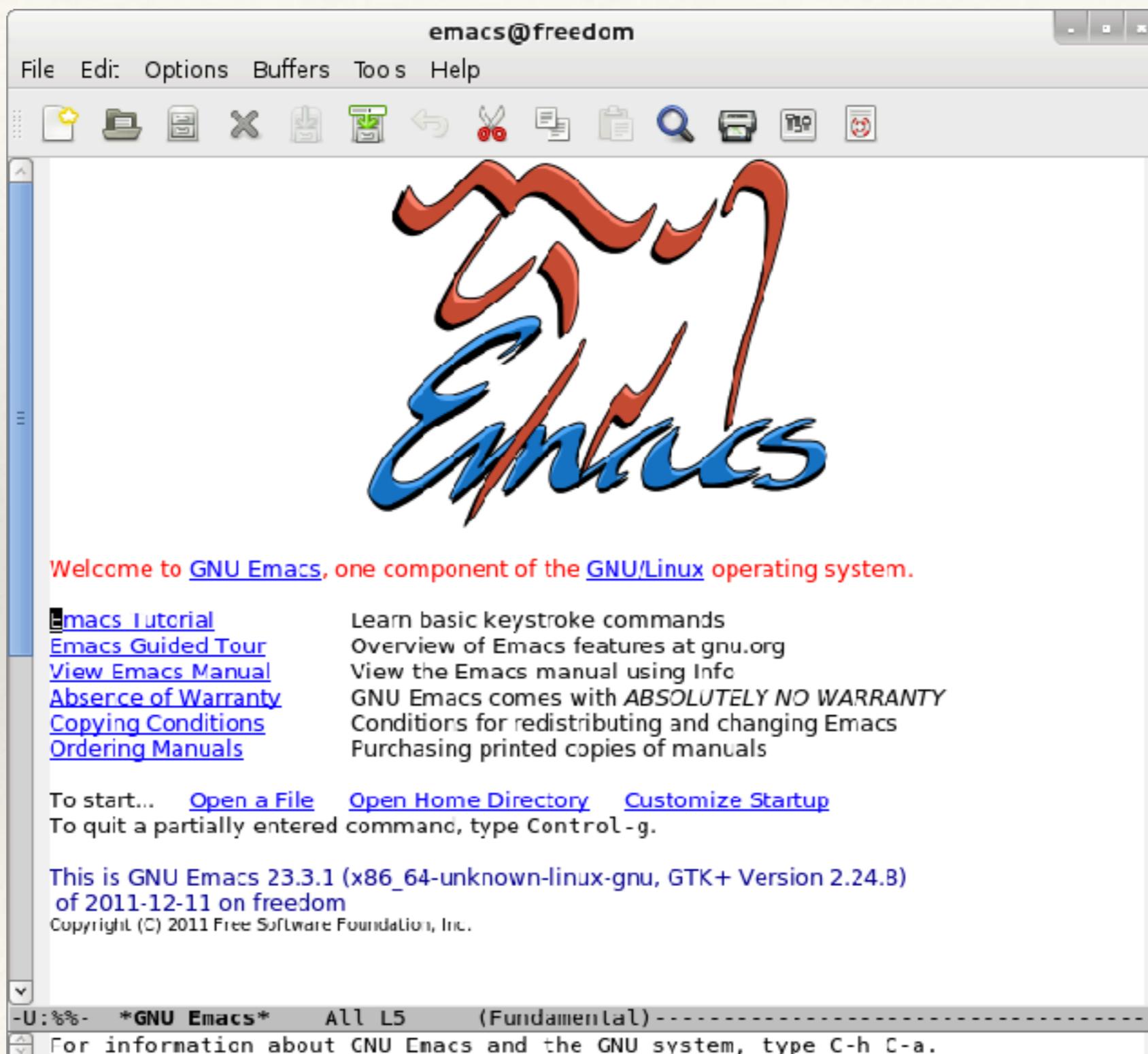
Chaque argument a le format suivant :

[**nomlogin@**] [**machine.site.domaine:**] [**chemin/fichier**]

scp donnees.csv pinlou@castor.isim.intra:Calculs/

TP : exo 9
(20 minutes)

Emacs



L'éditeur Emacs



Ce choix vous procurera quelques avantages :

- ▶ C'est un très bon outil pour la programmation :
 - ◆ colorisation syntaxique pour tout langage de programmation
 - ◆ indentation du code automatique
 - ◆ comparateur de codes
 - ◆ outils de déboggage de code
 - ◆ inclut un terminal de commande
- ▶ Emacs est gratuit et multi-plateforme (rentabilité de l'effort)
- ▶ Il est extrêmement paramétrable (raccourcis, modes,...)
- ▶ Il dispose de nombreux sites / forums / FAQ
- ▶ C'est un outil multi-usages (calendrier, messagerie, tetris, psychiatre, ...)

L'éditeur Emacs

- ▶ Comme tout éditeur puissant :
 - ▶ il peut être difficile à apprendre,
 - ▶ il nécessite de la mémoire,
 - ▶ les raccourcis claviers sont nombreux et parfois complexes (touches CTRL et META)

Windows:



Macintosh:



Unix:



- ▶ Bonne nouvelle :
La courbe d'apprentissage monte rapidement après les premières séances ...

À quoi ressemble Emacs ?



Notations des raccourcis claviers

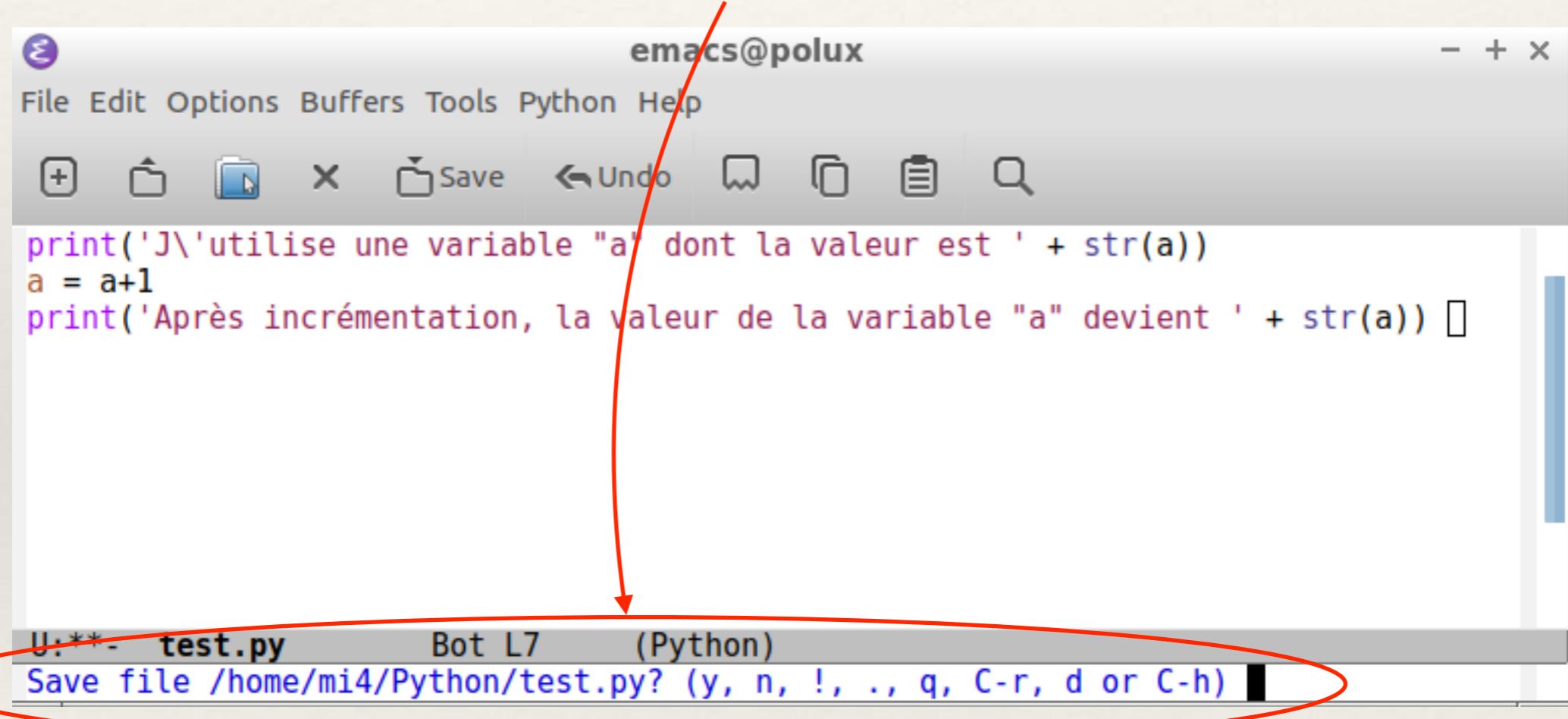
- ❖ La notation emacs pour les raccourcis : séquence des touches (préfixées des modificateurs le cas échéant).
- ❖ Les **modificateurs** standards sont
 - ▶ **C** pour **CTRL**,
 - ▶ **M** pour **META** (**Alt** sur la plupart des claviers),
 - ▶ **S** pour **Shift** (Maj)
- ❖ Exemples :
 - ▶ **C-x C-f** : « maintenir la touche CTRL, puis appuyer sur x et f »
 - ▶ **C-x f** : « maintenir CTRL, appuyer sur x, lâcher CTRL, appuyer sur f »
 - ▶ **C-M-f** : « maintenir CTRL et Meta, puis appuyer sur f »

Vocabulaire Emacs

- ❖ Tout document est édité dans une zone mémoire appelée **buffer**.
- ❖ Chaque **buffer** correspond(ra) à un fichier.
- ❖ Plusieurs **buffers** peuvent être montrés à l'écran à la fois
 - ▶ en séparant la zone d'édition,
 - ▶ ou bien être ouverts dans de nouvelles fenêtres (frames)
- ❖ Rappel : l'édition effectuée à l'écran n'affecte que le buffer en mémoire principale (RAM). Pour répercuter les changements dans le fichier correspondant sur disque, il faut sauver le buffer :
 - ▶ menu **Files** puis **Save Buffer**
 - ▶ **C-x C-s** (Save buffer)

Erreurs fréquentes

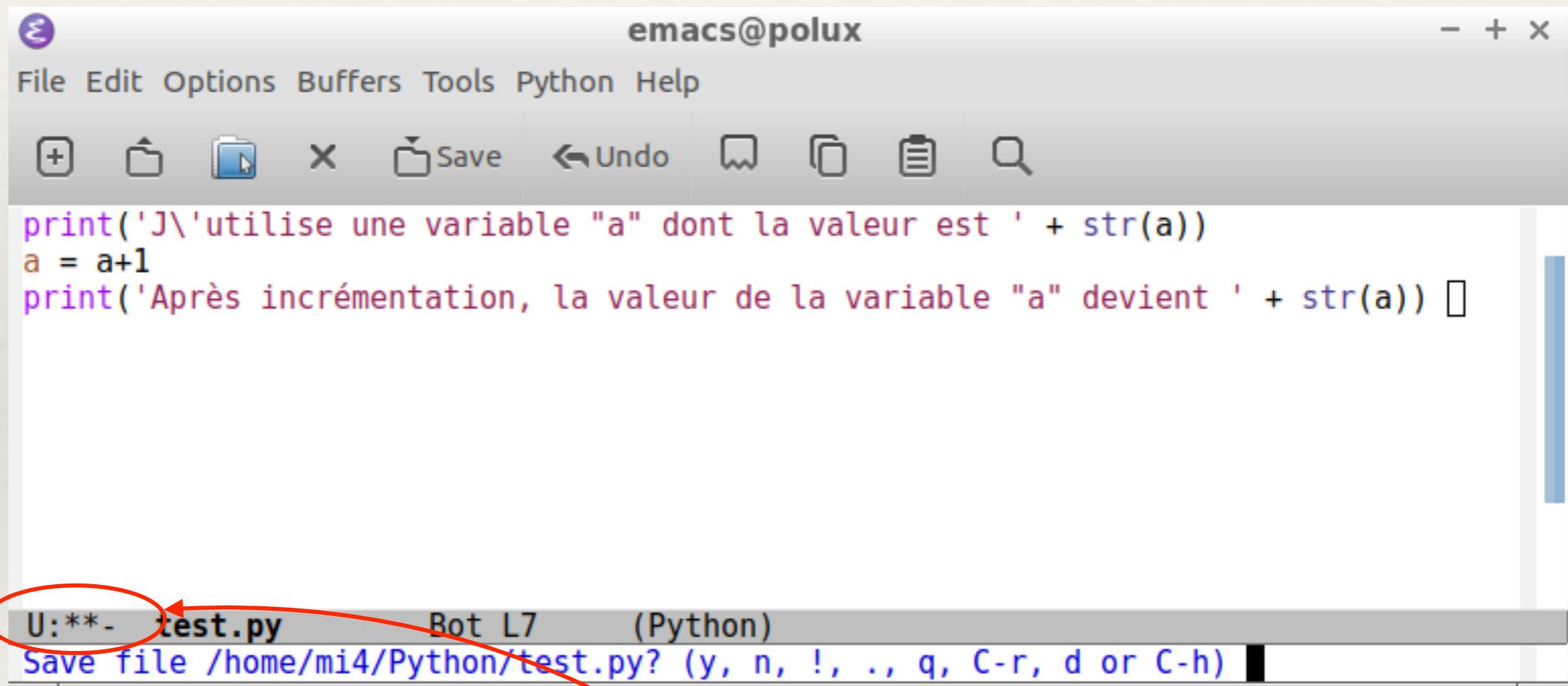
- ❖ Ne pas remarquer qu'Emacs peut vous poser des questions dans le **minibuffer**



- ❖ Oublier que le buffet n'est pas sauvegardé sur le disque !

Erreurs fréquentes

- ❖ Ne pas remarquer qu'Emacs peut vous poser des questions dans le **minibuffer**



- ❖ Oublier que le buffet n'est pas sauvegardé sur le disque !

N'ayez crainte ...

- ❖ La commande à toujours se rappeler : **C-g**
Vous sort de toute situation compliquée ou délicate :
demande à emacs d'arrêter ce qu'il est en train de faire
(à utiliser de façon répétée parfois avant de se faire obéir)

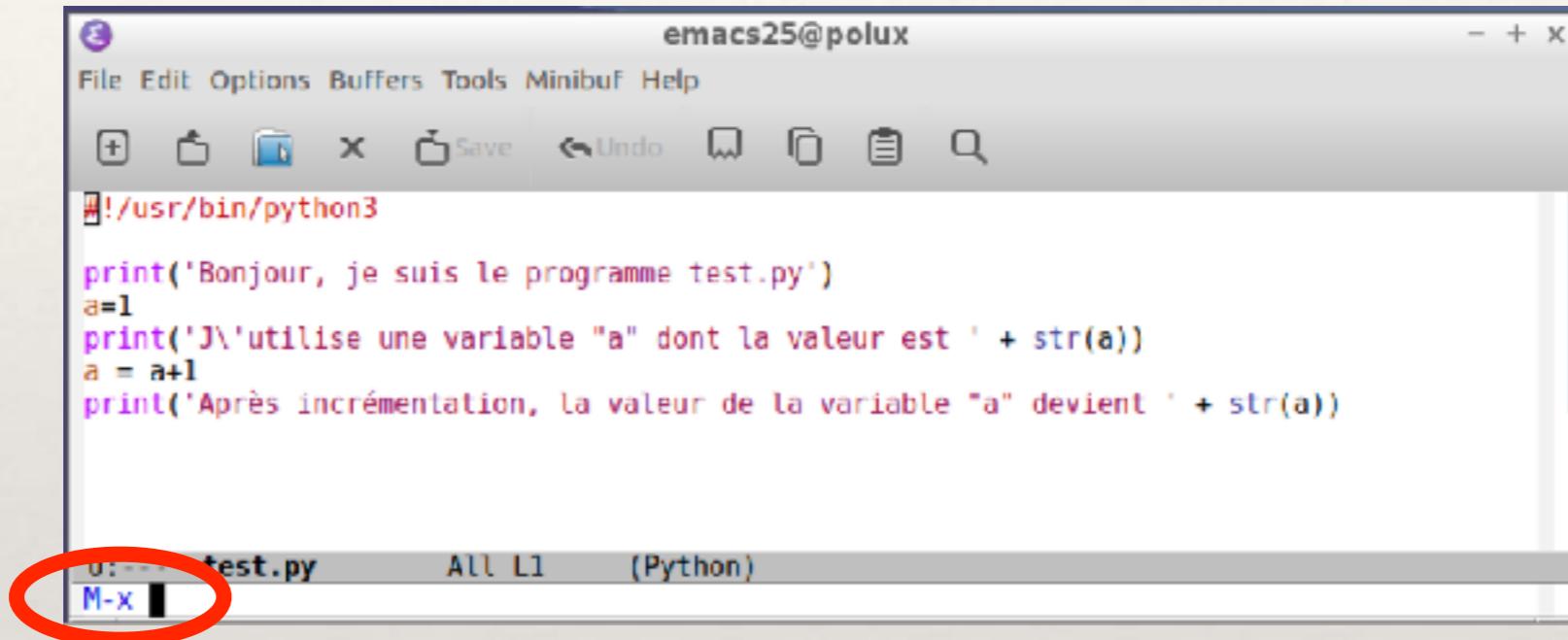
- ❖ Obtenir de l'aide : **C-h a <mot-clé>**
Liste l'ensemble des commandes relatives aux mot-clé
(avec les raccourcis associés)

Débuter un travail avec Emacs

- ❖ Débuter l'édition d'un fichier (Open | New) :
 - ▶ depuis un terminal : **emacs nomfichier &**
 - ▶ depuis emacs : **C-x C-f** (find file)
- ❖ Arrêter l'édition d'un fichier :
 - ▶ depuis le menu emacs : **Files → Kill Buffer**
 - ▶ raccourci : **C-x C-k** (kill buffer)
- ❖ Quitter emacs :
 - ▶ depuis le programme emacs : **Files → Exit Emacs**
 - ▶ raccourci : **C-x C-c** (close emacs)

Les commandes du minibuffer

- ❖ La clé **M-x** permet de passer dans le minibuffer pour lancer des commandes emacs



- ❖ On peut ensuite taper une commande emacs
Ex : **replace-string** / **save-buffer** / **find-file** /
- ❖ La touche  assure la complétion automatique, comme dans un Terminal

Les principaux raccourcis

❖ Mouvements du curseur :

- ▶ **c-d** delete char
- ▶ **c-a** beginning of line
- ▶ **c-e** end of line

❖ Copier/Couper/Coller :

- ▶ **C-[space]** set a mark (start of region)
- ▶ **C-w** cut the region from mark to current cursor location
- ▶ **M-w** copy region
- ▶ **C-y** paste region
- ▶ **C-k** cut end of line

❖ Chercher/Remplacer :

- ▶ **c-s** find word forward
- ▶ **c-r** find word in reverse
 - ◆ Use **c-s** and **c-r** to jump forward\back
 - ◆ Press **[enter]** to stop the cursor on the current match
- ▶ **M-%** replace with prompt

GNU Emacs Reference Card

(for version 25)

Starting Emacs

To enter GNU Emacs 25, just type its name: `emacs`

Leaving Emacs

suspend Emacs (or iconify it under X) `C-z`
exit Emacs permanently `C-x C-c`

Files

read a file into Emacs `C-x C-f`
save a file back to disk `C-x C-s`
save all files `C-x s`
insert contents of another file into this buffer `C-x i`
replace this file with the file you really want `C-x C-v`
write buffer to a specified file `C-x C-w`
toggle read-only status of buffer `C-x C-q`

Getting Help

The help system is simple. Type `C-h` (or `F1`) and follow the directions. If you are a first-time user, type `C-h t` for a **tutorial**.

remove help window `C-x 1`
scroll help window `C-M-v`
apropos: show commands matching a string `C-h a`
describe the function a key runs `C-h k`
describe a function `C-h f`
get mode-specific information `C-h m`

Error Recovery

abort partially typed or executing command `C-g`
recover files lost by a system crash `M-x recover-session`
undo an unwanted change `C-x u, C-_ or C-/`
restore a buffer to its original contents `M-x revert-buffer`
redraw garbaged screen `C-l`

Incremental Search

search forward `C-s`
search backward `C-r`
regular expression search `C-M-s`
reverse regular expression search `C-M-r`
select previous search string `M-p`
select next later search string `M-n`
exit incremental search `RET`
undo effect of last character `DEL`
abort current search `C-g`

Use `C-s` or `C-r` again to repeat the search in either direction. If Emacs is still searching, `C-g` cancels only the part not matched.

Motion

entity to move over	backward	forward
character	<code>C-b</code>	<code>C-f</code>
word	<code>M-b</code>	<code>M-f</code>
line	<code>C-p</code>	<code>C-n</code>
go to line beginning (or end)	<code>C-a</code>	<code>C-e</code>
sentence	<code>M-a</code>	<code>M-e</code>
paragraph	<code>M-{</code>	<code>M-}</code>
page	<code>C-x [</code>	<code>C-x]</code>
sexp	<code>C-M-b</code>	<code>C-M-f</code>
function	<code>C-M-a</code>	<code>C-M-e</code>
go to buffer beginning (or end)	<code>M-<</code>	<code>M-></code>
scroll to next screen		<code>C-v</code>
scroll to previous screen		<code>M-v</code>
scroll left		<code>C-x <</code>
scroll right		<code>C-x ></code>
scroll current line to center, top, bottom		<code>C-l</code>
goto line		<code>M-g g</code>
goto char		<code>M-g c</code>
back to indentation		<code>M-m</code>

Killing and Deleting

entity to kill	backward	forward
character (delete, not kill)	<code>DEL</code>	<code>C-d</code>
word	<code>M-DEL</code>	<code>M-d</code>
line (to end of)	<code>M-O C-k</code>	<code>C-k</code>
sentence	<code>C-x DEL</code>	<code>M-k</code>
sexp	<code>M-- C-M-k</code>	<code>C-M-k</code>
kill region		<code>C-w</code>
copy region to kill ring		<code>M-w</code>
kill through next occurrence of <i>char</i>		<code>M-z char</code>
yank back last thing killed		<code>C-y</code>
replace last yank with previous kill		<code>M-y</code>

Marking

set mark here	<code>C-@ or C-SPC</code>
exchange point and mark	<code>C-x C-x</code>
set mark <i>arg</i> words away	<code>M-@</code>
mark paragraph	<code>M-h</code>
mark page	<code>C-x C-p</code>
mark sexp	<code>C-M-@</code>
mark function	<code>C-M-h</code>
mark entire buffer	<code>C-x h</code>

Query Replace

interactively replace a text string using regular expressions	<code>M-%</code>
Valid responses in query-replace mode are	<code>M-x query-replace-regexp</code>
replace this one, go on to next	<code>SPC or y</code>
replace this one, don't move	<code>,</code>
skip to next without replacing	<code>DEL or n</code>
replace all remaining matches	<code>!</code>
back up to the previous match	<code>^</code>
exit query-replace	<code>RET</code>
enter recursive edit (<code>C-M-c</code> to exit)	<code>C-r</code>

Multiple Windows

When two commands are shown, the second is a similar command for a frame instead of a window.	
delete all other windows	<code>C-x 1</code> <code>C-x 5 1</code>
split window, above and below	<code>C-x 2</code> <code>C-x 5 2</code>
delete this window	<code>C-x 0</code> <code>C-x 5 0</code>
split window, side by side	<code>C-x 3</code>
scroll other window	<code>C-M-v</code>
switch cursor to another window	<code>C-x o</code> <code>C-x 5 o</code>
select buffer in other window	<code>C-x 4 b</code> <code>C-x 5 b</code>
display buffer in other window	<code>C-x 4 C-o</code> <code>C-x 5 C-o</code>
find file in other window	<code>C-x 4 f</code> <code>C-x 5 f</code>
find file read-only in other window	<code>C-x 4 r</code> <code>C-x 5 r</code>
run Dired in other window	<code>C-x 4 d</code> <code>C-x 5 d</code>
find tag in other window	<code>C-x 4 .</code> <code>C-x 5 .</code>
grow window taller	<code>C-x ^</code>
shrink window narrower	<code>C-x {</code>
grow window wider	<code>C-x }</code>

Formatting

indent current line (mode-dependent)	<code>TAB</code>
indent region (mode-dependent)	<code>C-M-\`</code>
indent sexp (mode-dependent)	<code>C-M-q</code>
indent region rigidly <i>arg</i> columns	<code>C-x TAB</code>
indent for comment	<code>M-;</code>
insert newline after point	<code>C-o</code>
move rest of line vertically down	<code>C-M-o</code>
delete blank lines around point	<code>C-x C-o</code>
join line with previous (with arg, next)	<code>M-^</code>
delete all white space around point	<code>M-\`</code>
put exactly one space at point	<code>M-SPC</code>
fill paragraph	<code>M-q</code>
set fill column to <i>arg</i>	<code>C-x f</code>
set prefix each line starts with	<code>C-x .</code>
set face	<code>M-o</code>

Case Change

uppercase word	<code>M-u</code>
lowercase word	<code>M-l</code>
capitalize word	<code>M-c</code>
uppercase region	<code>C-x C-u</code>
lowercase region	<code>C-x C-l</code>

The Minibuffer

The following keys are defined in the minibuffer.	
complete as much as possible	<code>TAB</code>
complete up to one word	<code>SPC</code>
complete and execute	<code>RET</code>
show possible completions	<code>?</code>
fetch previous minibuffer input	<code>M-p</code>
fetch later minibuffer input or default	<code>M-n</code>
regexp search backward through history	<code>M-r</code>
regexp search forward through history	<code>M-s</code>
abort command	<code>C-g</code>
Type <code>C-x ESC ESC</code> to edit and repeat the last command that used the minibuffer. Type <code>F10</code> to activate menu bar items on text terminals.	

Variables d'environnement et fichiers de configuration

```
TERM_PROGRAM=Apple_Terminal
TERM=xterm-256color
SHELL=/bin/bash
TMPDIR=/var/folders/yf/ycf6k41j57q0jvc9szlp7h740000gn/T/
Apple_PubSub_Socket_Render=/private/tmp/com.apple.launchd.A1N9Vlq6GM/Render
TERM_PROGRAM_VERSION=343.7
TERM_SESSION_ID=4D14E640-346C-4B29-9CB2-DD0461DC8629
USER=veryv
SSH_AUTH_SOCK=/private/tmp/com.apple.launchd.jZ2NSQ15Ox/Listeners
__CF_USER_TEXT_ENCODING=0x1F5:0x0:0x0
PATH=/Users/veryv/google-cloud-sdk/bin:/usr/local/bin:/usr/local/sbin:/usr/bin:/bin:/usr/sbin:
PWD=/tmp
XPC_FLAGS=0x0
XPC_SERVICE_NAME=0
HOME=/Users/veryv
SHLVL=1
LOGNAME=veryv
DISPLAY=/private/tmp/com.apple.launchd.flexq1QCHU/org.macosforge.xquartz:0
_= /usr/bin/env
```

Variables d'environnement

- ❖ Des informations sur l'utilisateur, le contexte de travail, la machine, ... sont stockées dans des variables :
 - ▶ informations liées au S.E. et aux logiciels installés (**OSTYPE**, **JAVA_HOME**, **PATH**, ...)
 - ▶ informations liées à l'utilisateur (**USER**, **HOME**, **PWD**, **SHELL**,...)
- ❖ Ces variables communiquent des informations entre programmes.

```
mi4@polux:~$ echo $HOME  
/home/mi4  
mi4@polux:~$ echo $SHELL  
/bin/bash  
mi4@polux:~$ echo $OLDPWD  
/home/mi4/Documents/Python  
mi4@polux:~$ PRENOM=Arthur  
mi4@polux:~$ echo $PRENOM  
Arthur  
mi4@polux:~$
```

echo : permet d'afficher le contenu d'une variable

VARIABLE = valeur
pour définir une variable

Variables d'environnement

- ❖ Quand un processus démarre son exécution, le processus qui le lance lui transmet les variables de son environnement sous la forme de couples :

nom = valeur

```
mi4@polux:~$ env
XDG_VTNR=7
SSH_AUTH_SOCK=/tmp/ssh-bzqijMj2OZU7/agent.1051
XDG_SESSION_ID=c2
XDG_GREETER_DATA_DIR=/var/lib/lightdm-data/mi4
USER=mi4
DESKTOP_SESSION=ubuntu
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/
sbin:/bin:/usr/games:/usr/local/games:/snap/bin
PWD=/home/mi4
HOME=/home/mi4
SSH_AGENT_PID=1146
OSTYPE=linux-gnu
QT_ACCESSIBILITY=1
```

Variable d'environnement

- ❖ La commande **export** permet d'ajouter une variable d'environnement. Une telle variable sera exportée vers ses processus fils lors de leur création.

```
mi4@polux:~$ DOC=/usr/share/doc  
mi4@polux:~$ export DOC
```

Les raccourcis (alias)

- ❖ Nous avons tendance à utiliser souvent les mêmes options : **ls -l**, **cd ..**, etc
- ❖ Il est possible de définir des **alias** pour ces combinaisons.

```
mi4@polux:~/Documents/Python/TP1$ alias up = 'cd ..'  
mi4@polux:~/Documents/Python/TP1$ up  
mi4@polux:~/Documents/Python$ ls  
test.py  
mi4@polux:~/Documents/Python$ ls -l  
total 1  
-rwxr-xr-x 1 mi4 student 225 oct. 17 14:14 test.py  
mi4@polux:~/Documents/Python$ alias ls = 'ls -l'  
mi4@polux:~/Documents/Python$ ls  
total 1  
-rwxr-xr-x 1 mi4 student 225 oct. 17 14:14 test.py  
mi4@polux:~/Documents/Python$
```

Fichiers de configuration

- ❖ Pour que les variables d'environnement et raccourcis soient **conservés** d'une fois sur l'autre, il est utile de les indiquer dans un des fichiers de configuration du shell (**.bashrc** ou **.bash_profile**)

```
mi4@polux:~$ less .bashrc
HISTCONTROL=ignoreboth
shopt -s histappend
if [ "$color_prompt" = yes ]; then
    PS1='${debian_chroot:+($debian_chroot)}[\e[01;32m]\u@\h\[
\033[00m]:[\e[01;34m]\w\[\033[00m]\$\ '
else
    PS1='${debian_chroot:+($debian_chroot)}\u@\h:\w\$ '
fi
unset color_prompt force_color_prompt
alias ll='ls -alF'
alias la='ls -A'
alias l='ls -CF'
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
```

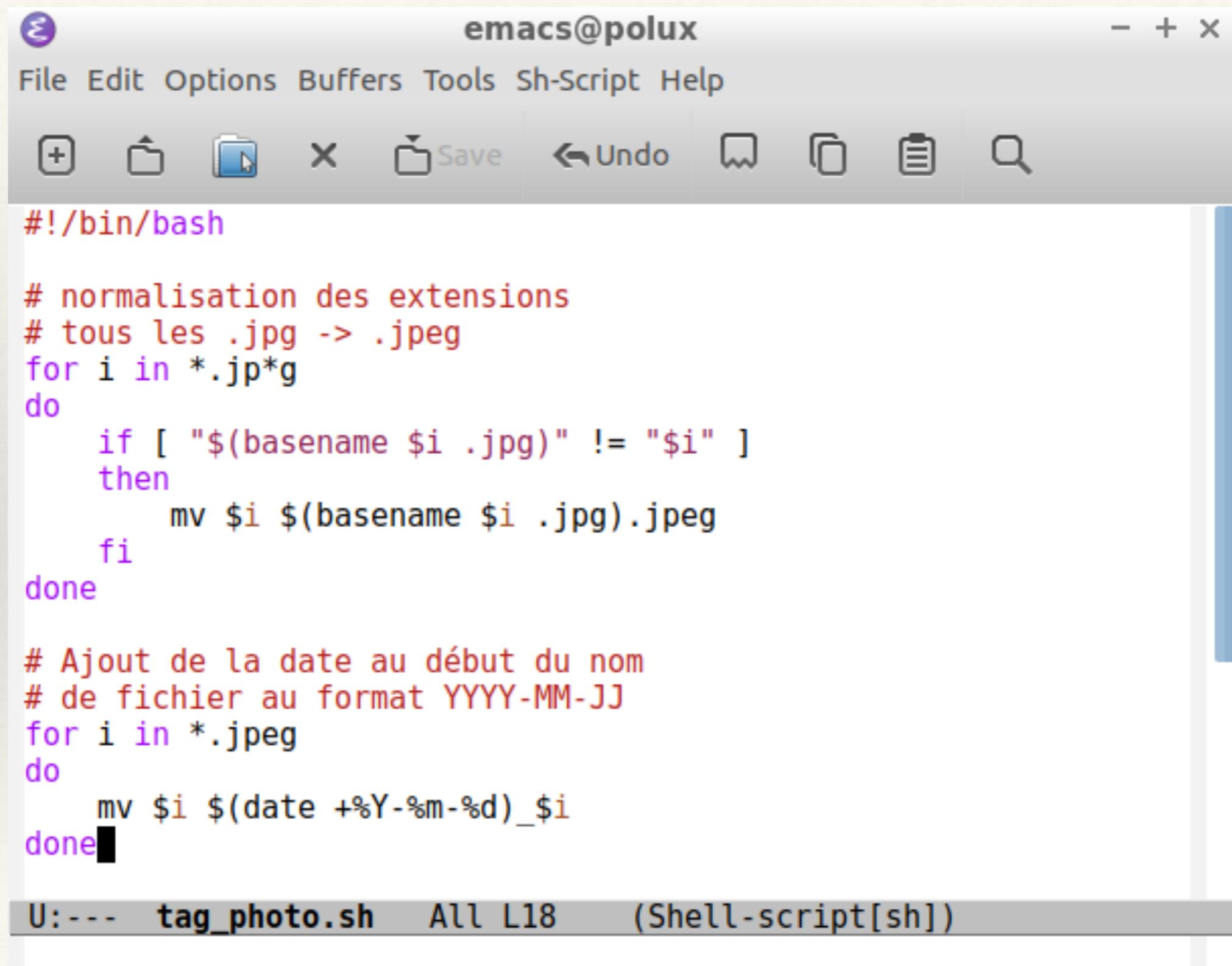
Scripts

```
#!/bin/bash
echo "Hello world!"
```

Les scripts

- ❖ Un **script** est un fichier texte contenant des commandes respectant une certaine syntaxe (ex : scripts Matlab).
- ❖ L'exécution de ces commandes, **les unes à la suite des autres**, se fait à chaque exécution du script.
- ❖ Le script lui-même s'exécute quand on tape **son nom dans un terminal** de commandes (uniquement s'il a les droits d'exécution, voir commande **chmod**).
- ❖ **Intérêt** : mémoriser une série de commandes fastidieuse et pouvoir l'exécuter avec des arguments différents d'une fois sur l'autre.

Exemple



The screenshot shows an Emacs window titled "emacs@polux". The menu bar includes "File", "Edit", "Options", "Buffers", "Tools", "Sh-Script", and "Help". The toolbar contains icons for new file, open file, save, undo, redo, copy, paste, and search. The buffer contains the following shell script:

```
#!/bin/bash

# normalisation des extensions
# tous les .jpg -> .jpeg
for i in *.jp*g
do
    if [ "$(basename $i .jpg)" != "$i" ]
    then
        mv $i $(basename $i .jpg).jpeg
    fi
done

# Ajout de la date au début du nom
# de fichier au format YYYY-MM-JJ
for i in *.jpeg
do
    mv $i $(date +%Y-%m-%d)_$i
done
```

The status bar at the bottom shows "U: --- tag_photo.sh All L18 (Shell-script[sh])".

Exemple

```
mi4@polux:~/Deplacement$ ls  
Confirmation.pdf Photo1.jpg Photo2.jpg Photo3.jpeg  
Photo4.jpg Photo5.jpeg Photo6.jpeg Photo7.jpg Photo8.jpg  
Planning.txt tag_photo.sh  
mi4@polux:~/Deplacement$ tag_photo.sh  
mi4@polux:~/Deplacement$ ls  
Confirmation.pdf 2017-10-26_Photo1.jpeg  
2017-10-26_Photo2.jpeg 2017-10-26_Photo3.jpeg  
2017-10-26_Photo4.jpeg 2017-10-26_Photo5.jpeg  
2017-10-26_Photo6.jpeg 2017-10-26_Photo7.jpeg  
2017-10-26_Photo8.jpeg Planning.txt tag_photo.sh  
mi4@polux:~/Deplacement$
```

Retour sur la variable d'env. PATH

```
mi4@polux:~/Deplacement$ ls  
Confirmation.pdf Photo1.jpg Photo2.jpg Photo3.jpeg Photo4.jpg  
Photo5.jpeg Photo6.jpeg Photo7.jpg Photo8.jpg Planning.txt  
mi4@polux:~/Deplacement$ tag_photo.sh  
tag_photo.sh : commande introuvable  
mi4@polux:~/Deplacement$ echo $PATH  
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/  
usr/games:/usr/local/games:/snap/bin:  
mi4@polux:~/Deplacement$ export PATH=$PATH:~/scripts  
mi4@polux:~/Deplacement$ echo $PATH  
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/  
usr/games:/usr/local/games:/snap/bin:../home/mi4/scripts  
mi4@polux:~/Deplacement$ tag_photo.sh  
mi4@polux:~/Deplacement$ ls  
Confirmation.pdf 2017-10-26_Photo1.jpeg 2017-10-26_Photo2.jpeg  
2017-10-26_Photo3.jpeg 2017-10-26_Photo4.jpeg  
2017-10-26_Photo5.jpeg 2017-10-26_Photo6.jpeg  
2017-10-26_Photo7.jpeg 2017-10-26_Photo8.jpeg Planning.txt  
tag_photo.sh  
mi4@polux:~/Deplacement$
```

Je range tous mes scripts
dans le dossier **~/scripts**

Scripts : les structures de contrôle

- ❖ Instructions conditionnelles :

```
if condition
    then {suite de commandes 1}
    elif condition2
    then {suite de commandes 2}
    else {suite de commandes n}
fi
```

- ❖ Dans l'expression de la condition :

! : négation **-o** : opérateur OU **-a** : opérateur ET

- ❖ Instructions d'itération (boucles) :

```
while <condition>
...
do
    {suite de commandes}
done
```

```
for i in <elm1> <elm2> <elm3>
do
    {suite de commandes}
done
```