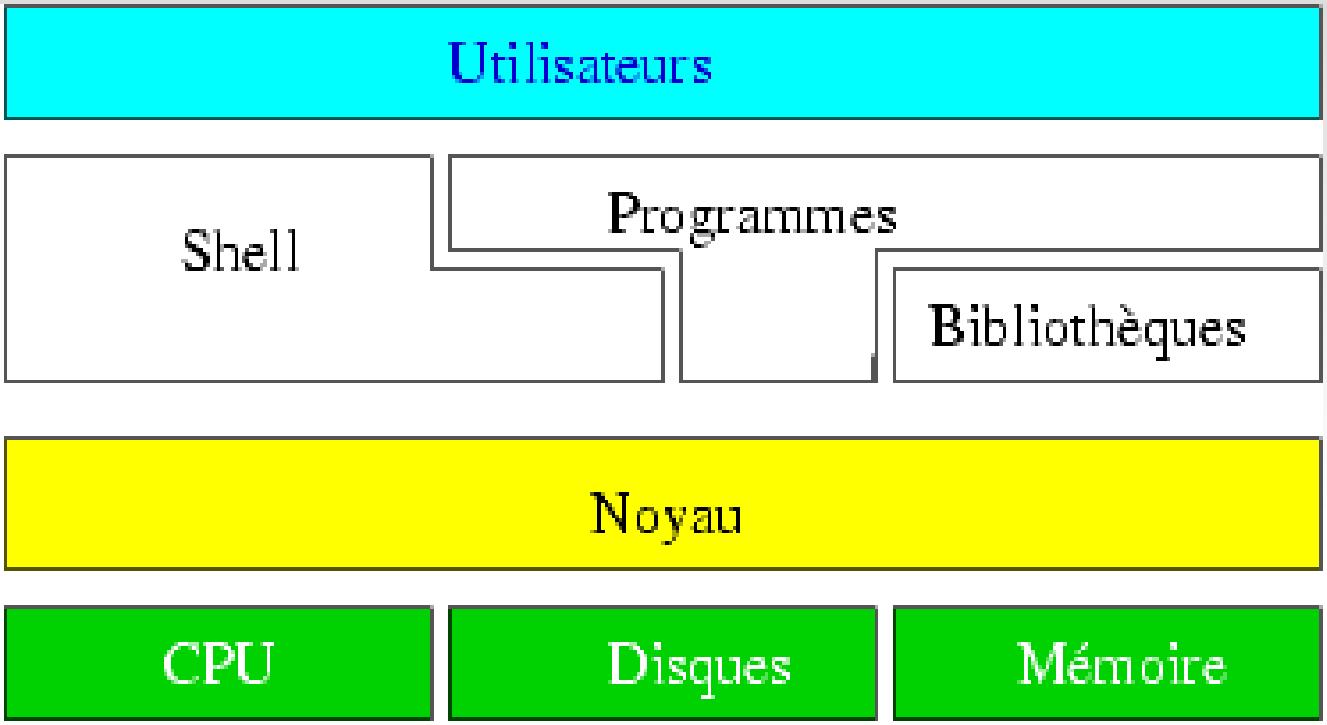
# Unix-Intro

- Crée en 1969. (pas vieux: mûr ;-)
- Pas UN unix mais DES unix
  - un arbre genealogique compliqué ...
  - Plus connus: Linux, xxBSD, MacOS ...

- Caractéristiques:
  - Multitaches (temps partagé)
  - Multi-utilisateurs (droits+extinction).
  - Interactif (et batch).
  - Avec ou sans interface graphique.
    - question de gouts ...Et d'usage

Centrale Marseille

# Vue générale




- aucun process n'accède directement à une ressource matérielle → sous le controle du kernel
  - fichiers spéciaux
- Cours = Shell = Interprêteur de commandes

# Syntaxe générale des commandes

- Les commandes se tapent dans un terminal ou shell
  - **`prompt$ cde [options] [arguments]`**
- **`prompt$`** : ce qui figure au debut de chaque ligne
- [] : facultatif
- Les commandes acceptent ou non des options et/ou des arguments
- **`$man commande`** pour tout savoir

Centrale Marseille

# Syntaxe générale des commandes (2)

- Qqs exemples:
- **`$cde [[-a value] [-b] fichier1 [fichier2]]`**
  - **`$cde`**
  - **`$cde fichier1`**
  - **`$cde fichier1 fichier2`**
  - **`$cde -b fichier1`**
  - **`$cde -a fichier1`** → pas valide

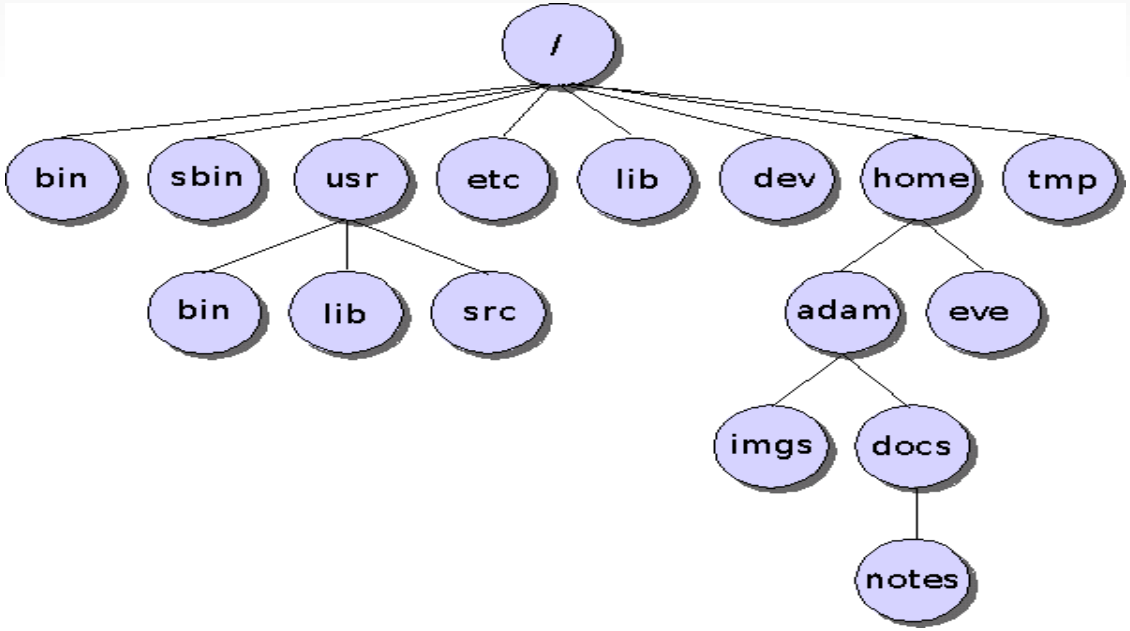
Centrale Marseille

# Fichiers

- Sous Unix, tout est fichier !!!
  - Fichier “normaux”, repertoires, peripheriques …
- Tout fichier à un nom.
  - Long max: 256 caracteres
  - Peuvent être composés de majuscules, minuscules, chiffres, symboles …
    - On évite -{ }[]#&?” .... car ils ont un sens
  - “case sensitive”: **`toto <> Toto`**
  - Extension non-significative
  - Fichiers “cachés”

# Fichiers

- Tout fichier est dans l'arborescence
  - Arborescence unique <> Windows
  - Racine: /



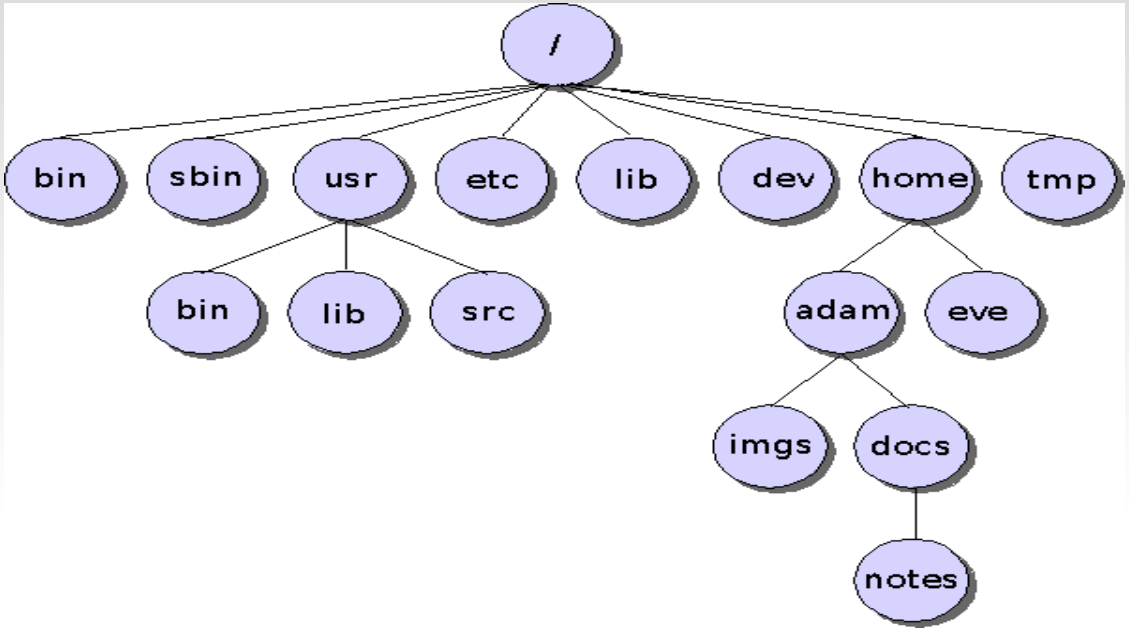

Centrale Marseille

# Fichiers

- Tout fichier est dans l'arborescence
  - Clef USB, disquettes, disques → repertoires
  - /media
  - Automontage
    - Il faut accéder à un périphérique pour le voir dans /media
- pour voir la liste des périphériques montés:
  - **`%mount [nbreuses options]`**
- idem + tailles (total, used, available)
  - **`%df [-k] [-h]`**

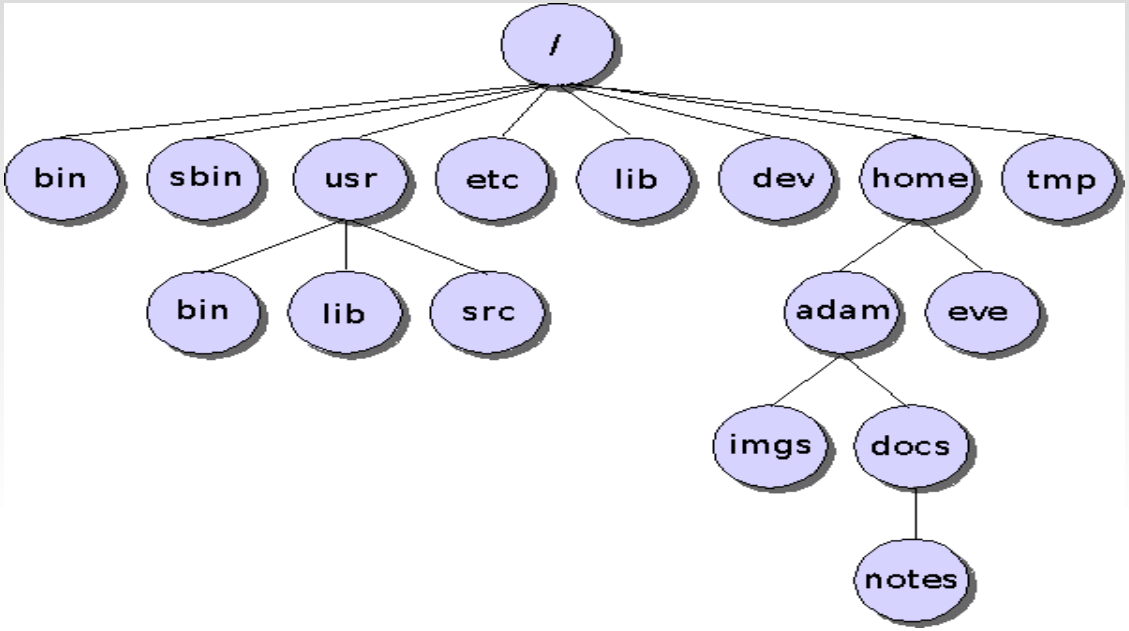
Centrale Marseille

# Fichiers – chemins absolus




- /home/adam/docs/notes/machin: chemin absolu
- désigne qqchose (repertoire ? fichier ?) qui est dans le rep notes qui est dans le rep docs qui est ...

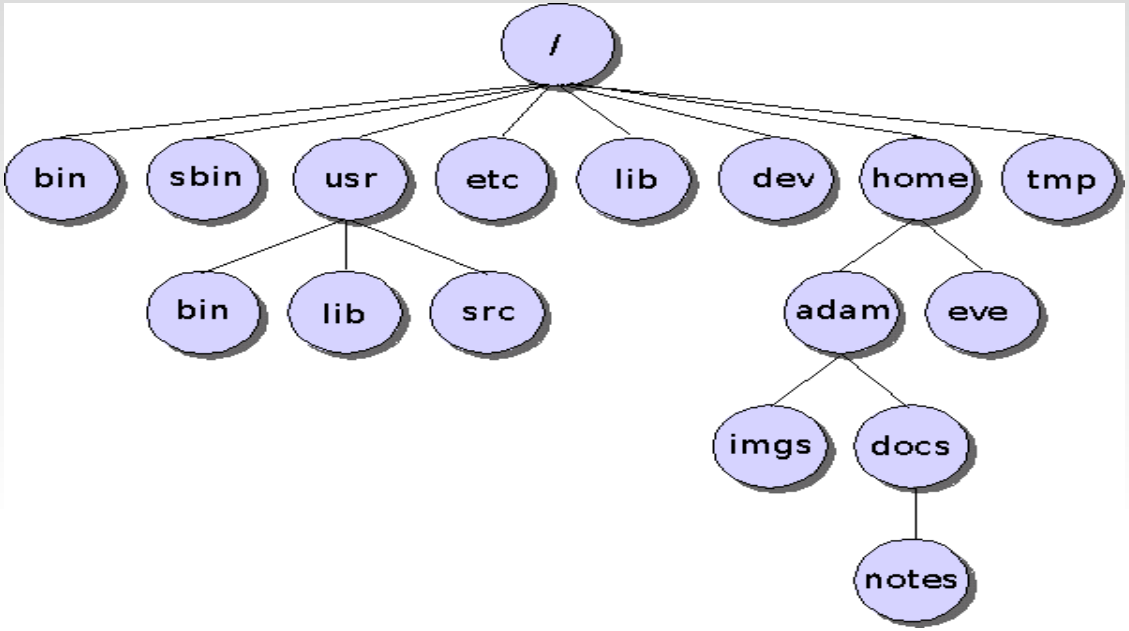
# Fichiers – chemins relatifs




- docs/notes/machin: chemin relatif
- N'a de sens que “par rapport” à /home/adam

Centrale Marseille

# Fichiers – chemins relatifs



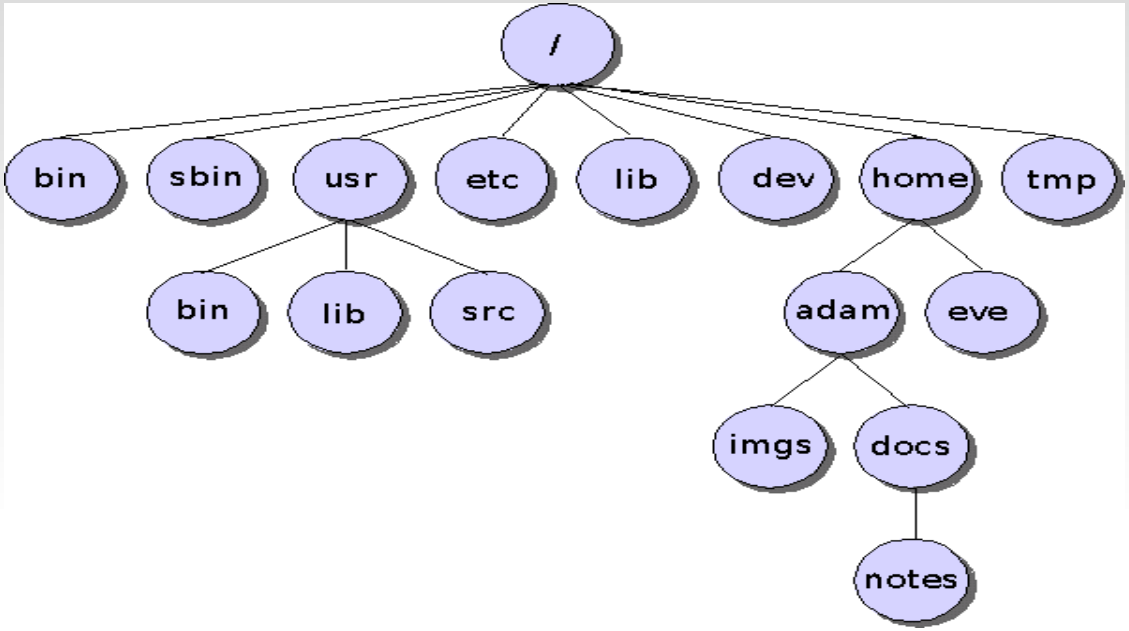

- .. = le repertoire “parent”
- ../eve est equivalent à /home/eve
  - N'a de sens que dans un sous rep de home
  - ../../etc idem

# Fichiers – chemins relatifs

- . = Repertoire courant: le repertoire dans lequel on est …
  - ./machin : le fichier ou repertoire nommé "machin" qui est dans le repertoire dans lequel je suis
  - machin : idem
  - utilisé comme argument de commande pour designer le rep courant. Ex: `$cde arg1 .`
- ./qqchose: surtout utilisé pour exécuter un programme situé dans le rep courant (c.f + tard)

Centrale Marseille

# Fichiers – chemins relatifs




- Home directory (maison …)
  - La partie de l'arborescence qui “est à vous”.
- /home/adam = “chez adam”
- ECM : pas /home

Centrale Marseille

# Fichiers – chemins relatifs

- ~ = votre Home Directory
- ~tartempion = le Home directory de tartempion
- ~tartempion/../../truc_tordu
  - Un fichier ou rep nommé truc_tordu situé 2 repertoires “au dessus” de “chez tartempion”

Centrale Marseille

# Fichiers – Metacaractères

- * : remplace tte chaine de caractères
  - **`$cde toto*`** : Lance une commande sur tout ce qui commence par toto et qui est situé dans le rep courant (.)
  - **`$cde /users/*toto*`** …
  - **`$cde */toto`** …
  - …

Centrale Marseille

# Fichiers – Metacaractères

- ? : remplace un caractère
  - **`$cde t?t?`**
- [range] : remplacé par tous les caractères compris dans le range
  - **`$cde t[aeiouy]t[aeiouy]`**
  - **`$cde [A-Z]*`**
  - ....

Centrale Marseille

# Commandes

- Pour commencer, qqs questions existentielles:
  - Qui suis je ?: $**`whoami / $id [-a]`**
  - Je veux devenir un autre !!!:
    - **`$su - login`**
    - **`$logout` ou CTRL D**
  - Ou suis je ?( sur quelle machine ): **`$hostname`**
  - Ou suis je ?( dans quel repertoire): **`$pwd`**
  - Qui est sur la même machine ?: **`$who`**
  - Quelle heure est il ?: **`$date`**

What's the meaning of life ?: :h 42 sous vi

Centrale Marseille

# Cdes de manipulation fichiers & repertoires

- **`$pwd`** : “quel est le rep courant ?”
- **`$cd [rep]`** : aller dans le repertoire spécifié
- **`$mkdir [options] rep [rep2 …]`**
- **`$ls [options] [arguments]`**: afficher ce qu'il y a dans un repertoire
  - **`$ls`**
  - **`$ls /`**
  - **`$ls *.doc`**

Centrale Marseille

# Cdes de manipulation fichiers & repertoires

- **$cp [options] nom1 [nom2 … nom3] nom4**
  - **$cp fic1 fic2**
  - **$cp fic1 fic2 rep**
  - **$cp rep1/* rep2**
  - **....**
- **$mv [options] nom1 [nom2 … nom3] nom4**
  - **$mv fic1 fic2**
  - **$mv fic1 fic2 rep**
  - **$mv rep1/* rep2**
  - **....**

Centrale Marseille

# Cdes de manipulation fichiers & repertoires

- **$rmdir [options] rep1 [rep2 … rep3]**
  - Peu utilisé car limitations (c.f TP)
- **$rm [options] nom1 [rep2 … rep3]**
  - **$rm ../toto***
  - **$rm ***

Centrale Marseille

# Complement `$ls -l`

```
-rw-r--r-- 1 root    admin_cri     353 2005-05-11 15:19 t3
-rwxr-xr-x 1 root    admin_cri   47624 2005-05-11 15:19 t4
drwx------ 2 root    admin_cri    4096 2005-05-11 15:19 tmp
lrwxrwxrwx 1 root    root            3 2008-10-09 15:15 toto -> tst
-rw-r----- 1 root    admin_cri     590 2005-05-11 15:19 toto.trtr
```

- `[-dlbcp]: - fichier, d repertoire, l lien, [bcp] speciaux (devices)`
- `[r-][w-][x-] [r-][w-][x-] [r-][w-][x-]:` droits pour owner, group, others
- `Inode` ( pas etudié)
- `owner`
- `group`
- `taille ( n'a de sens que pour les fichiers normaux !!)`
- `date_modif`
- `nom`

Centrale Marseille

# Droits sur les fichiers

- vous pouvez choisir si vous voulez que vos fichiers soient :
  - lisibles **(r)**
  - et/ou modifiables **(w)**
  - et/ou empecher que d'autres utilisateurs lancent vos programmes **(x)**
- Pour vous **(rwx)**, un groupe **(rwx)**, les autres **(rwx)**
  - **rwxrwxrwx**

Centrale Marseille

# Droits sur les repertoires

- vous pouvez choisir si vous voulez que certains utilisateurs puissent:
  - Voir ce qu'il y a dans un repertoire **`(r)`**
  - Modifier ce repertoire (et ce qu'il contient) **`(w)`**
  - Aller dans ce repertoire (traverser) **`(x)`**

- Generalement, on ne dissocie pas **`(r)`** et **`(x)`**
- Pour changer le groupe:
  - **`$chgrp groupe nom1 [nom2 ...]`**

Centrale Marseille

# Modification des droits

- Changer le proprietaire: (réservé à root)
  - **`$chown user nom1 [nom2 ...]`**
- Pour changer le groupe: (si vous app. au groupe)
  - **`$chgrp groupe nom1 [nom2 …]`**
- Changer les droits:
  - **`$chmod .........(c.f suite)`**

Centrale Marseille

# Changer les droits – 1

**Notation symbolique:**

- **`$chmod [opts] qui operation droits quoi`**
- Utilisateur:
  - **`u proprietaire (user)`**
  - **`g group`**
  - **`o les autres`**
- Operation:
  - **`+ ajout d'un droit`**
  - **`- suppression d'un droit`**
  - **`= affectation d'un droit`**

Ex: **`$chmod g+w,o-rwx fichier`**

Centrale Marseille

# Changer les droits - 2

- Notation octale:
  - Savoir compter jusqu'a 7 …

```
421
rwx=4+2+1=7
r-x=4+1=5
…
$chmod 644 fichier
$chmod 750 repertoire
```

Centrale Marseille

# Droits – tableau recap

| | rep src | rep dst | fic src | fic dst | rep traversés |
|---|---|---|---|---|---|
| Aller dans un rep | | --x (r-x) | | | --x (r-x) |
| Lister contenu rep | | r-- (r-x) | | | --x (r-x) |
| Copier fic | r-x | rwx | r-- | rw- | r-x |
| Supprimer fichier | | rwx | rw- | rw- | r-x |
| Deplacer | rwx | rwx | rw- | rw- | r-x |

# Droits - compléments

- Lorsqu'on crée un fichier ou un repertoire, il y a des droits par defaut
- **`$umask val_octale`**
- **`val_octale`** sont les droits supprimés aux droits implicites
- Droits implicites: 666 fichier, 777 repertoire
- **`$umask 022`**
  - **`666 - 022 = 644 fichier`**
  - **`777 - 022 = 755 rep`**

Centrale Marseille

# Fichiers spéciaux - liens

- But: Créer un “Alias/raccourci” → lien
- **`$ln -s celui_qui_existe nom_du_lien`**

- Ex:
- **`$ln -s ~olivier/coursunix ~/coursunix`**
- **`$ls -l ~`**

```
lrwxrwxrwx 1 op op 18 2009-09-22 16:31 coursunix ->
  ~olivier/coursunix
```

Centrale Marseille

# Qqs commandes

- **`$file fic`**
  - Affiche le type de fichier (ascii text, executable, audio …)
- **`$cat fic`**
  - Affiche le contenu d'un fichier texte (ascii)
- **`$more fic`**
  - Idem mais page par page
- Pour modifier un fichier texte il faut un editeur:
  - Mode texte: **`vi, emacs, nano, joe ...`**
  - Mode graph: **`gvim, gedit, nedit ...`**

Centrale Marseille

# Qqs commandes - Imprimer

- **File → Print  ;-)**
- **$lp [opt] [-d nom_imp]fichier(s)**
  - Attention, seulement fichiers texte et postcript !
- **$lpstat -p [opts]**
  - Affiche le nom imprimantes + états (**num_job**)
- **$cancel num_job**
  - Annule le job d'impression
- **$lpoptions -l nom_imprimante**
- **$gtklp, $xpp, $qtcups** (graphique)

Centrale Marseille

# Qqs commandes

- **`$a2ps -P imprimante fic`**
  - **`Convertit le fichier texte avant de l'imprimer`**
- **`$diff fic1 fic2`**
  - Affiche les differences entre deux fichiers
- **`$diff -bru rep1 rep2`**
  - Idem entre deux repertoires
  - Peut être utilisé pour créer des patches

Centrale Marseille

# Qqs commandes - find

- **`$find a_partir_de condition action`**
  - Condition: **`-name “qqchose”, -newer “qqchose”, -size, -user, -type ....`**
  - Action: **`-print , -delete, -exec`** …
- Ex:
  - **`$find ~ -name “toto*” -print`**
  - **`$find rep -newer “truc” -delete`**

Centrale Marseille

# Qqs commandes - Archiver

- Une archive est un fichier d'un format particulier (.tar) qui peut contenir plusieurs fichiers et/ou repertoires.
- Utile pour communiquer
- **`$tar [options] arguments`**
- Créer une archive: **`-c`**
  - **`$tar -cvf nom_archive quoi`**
  - **`Ex: $tar -cvf projet.tar repertoire`**
- Afficher ce que contient une archive: **`-t`**
  - **`$tar -tvf nom_archive`**
- Extraire ce que contient une archive: **`-x`**
  - **`$tar -xvf nom_archive [nom]`**

Centrale Marseille

# Qqs commandes - Compresser

- Attention:
  - une archive n'efface pas la source.
  - Une archive n'est pas compressée.
- Compresser: plusieurs algorithmes, plusieurs commandes:
  - **`$compress/uncompress: .Z`**
  - **`$gzip/gunzip: .gz`**
  - **`$bzip2/bunzip2: .bz2`**

Centrale Marseille

# Qqs commandes – Archiver / Compresser

- On peut desormais manipuler directement des archives compressées:
- **`$tar -cvzf nom_archive_compressée nom`**
  - **`Ex: tar -cvzf projet.tar.gz rep`**
  - **`(ou .tgz)`**
- **`$tar -cvjf nom_archive_compressée nom`**
  - **`Ex: tar -cvjf projet.tar.bz2 rep`**
  - **`(ou .tbz2)`**
- Idem pour afficher ou extraire ....

Centrale Marseille

# Commandes / processes

A) Qu'est ce qu'une commande ou un programme ?

B) Comment lance-t-on un programme?

Par quelle “magie”, lorsque vous tapez le nom d'une commande/programme dans un terminal, se passe-t-il qqchose ?

Centrale Marseille

# Commandes / processes

A) Une commande/programme est un fichier executable situé “qqpart” dans l'arborescence (dont les droits d'execution(x) sont positionnés )

B)

1)soit on décrit explicitement le chemin jusqu'a cet executable:

Ex:

```
$/usr/local/bin/programme
$ bin/mes_pgms/programme
```

Centrale Marseille

# Commandes / processes

2)soit cet executable est situé dans un des repertoires définis par le **PATH**

**PATH** est une variable qui définit une liste ordonnée de repertoires dans lesquels le système cherche les éxecutables.

```
$echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/
  sbin:/usr/bin:/sbin:/bin:/usr/games
$which firefox
/usr/bin/firefox
```

sinon: "**not found**"

Centrale Marseille

# Commandes / processes

**Modifier le PATH:**

```
$ PATH=$PATH:nouveau_repertoire
```

Attention ! Vous risquez de ne plus pouvoir lancer de commandes si vous vous trompez !!!

Ex: **`$ PATH=~/bin`**

Attention: le PATH n'est modifié que pour le terminal dans lequel vous l'avez défini

→ **`~/.bashrc`** (c.f plus loin)

Centrale Marseille

# Commandes / processes

Lorsque vous lancez une commande, vous ne pouvez pas en lancer une autre tant que la précédente ne s'est pas terminée.

→ la commande est alors lancée en “premier plan” ou “**foreground**”

```
$ nedit ~olivier/coursunix/TP12
```

Vous pouvez interrompre (tuer) une commande avec

```
CTRL C
```

Centrale Marseille

# Commandes / processes

Vous pouvez suspendre (stopper) une commande avec

```
CTRL Z
```

le temps d'en taper une autre

```
$ls
```

Vous pouvez relancer la commande precedente en la relancant en “premier plan”

```
$fg
```

Ou en “second plan” (background)

```
$bg
```

Centrale Marseille

# Commandes / processes

Vous pouvez lancer une commande directement en second plan (background):

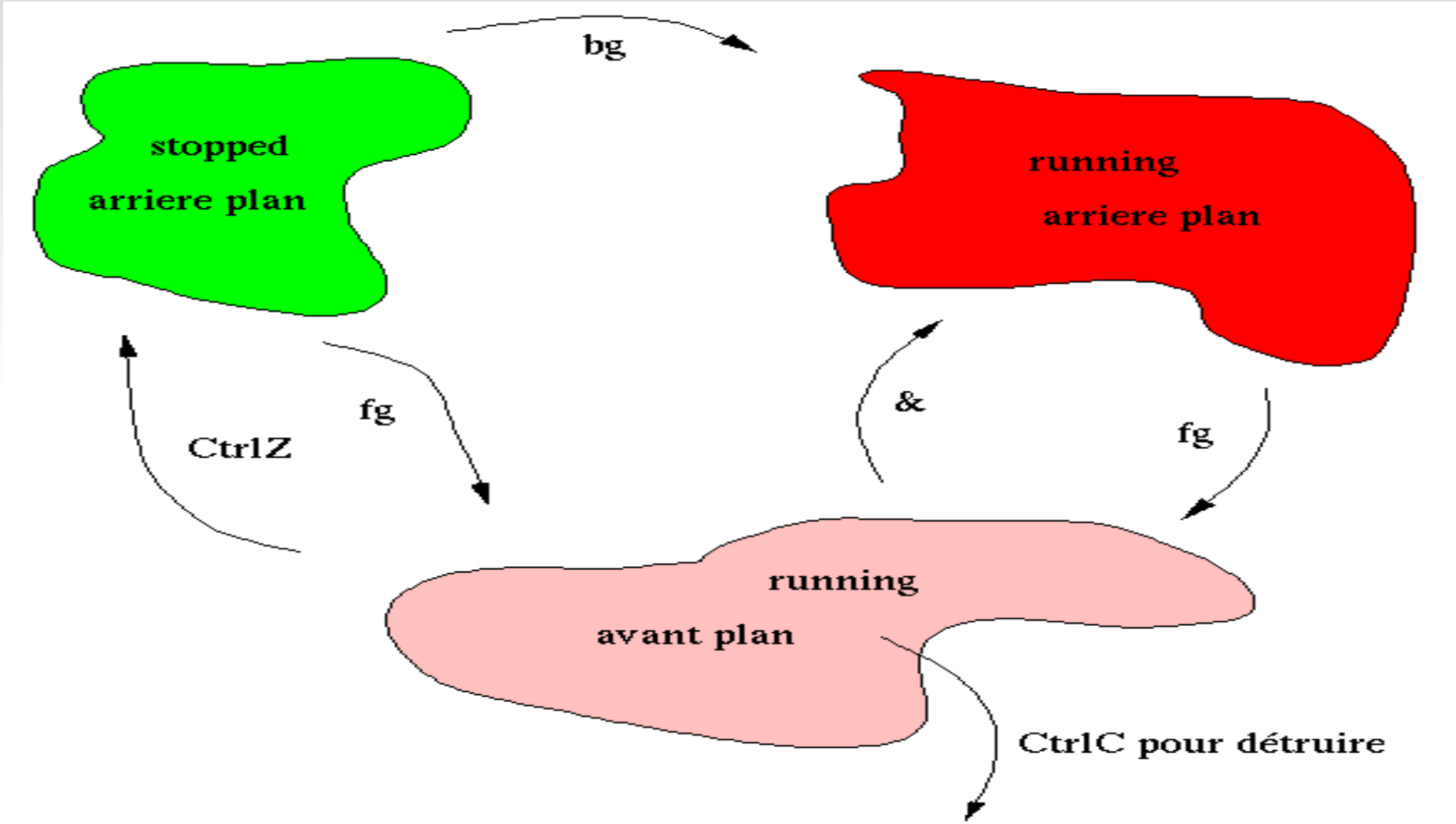
```
$ cde [opt] [arguments] &
```

Puis la reprendre en premier plan

```
$ fg ... CTRL Z etc ....
```

Centrale Marseille

# Commandes / processes




c.f: http://www.idris.fr/data/cours/unix/user/unix_u_cours.html

Centrale Marseille

# Commandes / processes

Pour lister TOUS les processes en cours d'execution:

```
$ ps auxw
```

```
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root      5106  0.0  0.1  13256  4912 ?        S    12:09   0:00
   /usr/bin/timidity -Os -iAD

root      5111  0.0  0.0   1808   532 tty1     Ss+  12:09   0:00
   /sbin/getty 38400 tty1

postfix 5204  0.0  0.0   5840  1732 ?        S    12:09   0:00 qmgr -l
   -t fifo -u

op        6637  0.0  0.2  11364  7200 ?        Rs   12:27   0:00
   /usr/bin/xterm

22456     6638  0.0  0.0   4812  2064 pts/1    Ss   12:27   0:00 bash
```

Centrale Marseille

# Commandes / processes

**Pour tuer un/plusieurs process(es):**

```
$ kill -9 PID
$ killall nom_cde
```

**Lister les n processes qui occupent le plus de CPU/MEM....**

```
$ top
```

Centrale Marseille

# Commandes / processes

Attention: si on ferme sa session, les processes sont tués !!!

```
$ nohup cde [opts] [arguments]
```

On retrouve ce qui devait s'afficher dans le fichier nohup.out

Centrale Marseille

# Commandes / processes

Lancement en différé:

```
$ at qqchose
cde1
cde2
…
CTRL D
```

```
Ex: $ at now + 3 hours
Ex: $ at midnight … (man at)
$ atq
$ atrm no_job
```

Centrale Marseille

# Commandes / processes

Périodiquement:

```
$ man crontab
```

Rendre un process moins prioritaire

```
$renice +level PID
```

Executer qd le système “n'a que ca a faire”

```
$ batch cde
```

Centrale Marseille

# Caractéristiques du bash

Il exite plusieurs interpréteurs de commande / shells

**`sh, ksh, csh, tcsh, bash, zsh`** …

Par défaut, vous utilisez le bash

Pour changer: demandez aux sysadm !!!

Ou alors tapez le nom du shell désiré ...

```
$ bash
```

Centrale Marseille

# Caractéristiques du bash

Mécanisme d'historique

```
$ history
$ !!
$ !3
$ !-2
$ !truc
$ mkdir toto
$ rm -r !$
$ mkdir toto tutu
$ rm -r !*
$ mkdir !:2
```

Centrale Marseille

# Caractéristiques du bash

Mecanisme de substitution sur la ligne de commande

```
$ cp ~oliver/coursunix/TP22 ~
$ ^er^ier
```

Centrale Marseille

# Caractéristiques du bash

Alias(es): **$ alias**

```
$ alias h='history'
$ alias fi='firefox'
$ alias ll='ls -l'
$ alias rm='rm -i'
$ alias cp='rm'
$ \cp fic1 fic2 ou $unalias cp
```

Les aliases n'existent que dans le terminal dans lequel ils ont ete définis !!!

Comment définir des aliases valables dans tous les terminaux ??? → **~/.bashrc**

Centrale Marseille

# Caractéristiques du bash

**`~/.bashrc`** est un fichier qui s'execute à l'ouverture de chaque terminal/shell.

On y place donc les commandes que l'on souhaite voir s'executer systématiquement …

Rem: **`~/.bashrc`** s'execute après **`/etc/bashrc`**

Centrale Marseille

# Qqs cdes – fichiers texte

**$head [options] [fichier]**

ex: **$head -5 fichier**: Affiche les 5 premieres lignes

**$tail [options] [fichier]**

**$tail -5 fichier**

**$tail +5 fichier**

**$tail -f fichier**

**$wc [-l] [-c] [fichier]**

Compte les lignes, caracteres

Important: si pas de fichier en argument, les Cdes opèrent sur ce qui est saisi au clavier.

Centrale Marseille

# Qqs commandes – grep

**`$grep [options] regexp [fic(s)]`**

regexp: expression rationnelle, pattern, motif.

**`$grep toto fic1`**

Affiche les lignes de fic1 contenant la chaine “toto”

**`$grep ^toto fic1 fic2`**

Affiche les lignes de fic1 et fic2 qui commencent par la chaine toto.

**`$grep toto$ *.c`**

Affiche les lignes de ts les fichiers .c qui se terminent par toto

Centrale Marseille

# Qqs commandes – grep

**Options:**

-v: tout sauf

-r: recursivement

-i: case insensitive

-n: affiche num de ligne

-l: nom fichier

....

Centrale Marseille

# Qqs commandes – grep

**Expressions rationelles:**

```
^, $
[A-Z], [d-f], [3-8]
[aeiouy]
[^A-Z] est différent de ^[A-Z]
[:alpha:] <=> A-Za-z
   [[:alpha:]] <=> [A-Za-z]
```

etc ...

Centrale Marseille

# Qqs commandes – cut

## Afficher certaines "colonnes"

On parle plutôt de champs et de séparateurs

(**esp , ; $ % x** …)

**totoxtutuxtitixtata**

Si x est le separateur **toto tutu titi tata**

Si u est le separateur **totoxt t xtitixtata**

**$cut [opts] [fic(s)]**

**-d delimiteur (separateur)**

**-f ch1,ch2**

**-f ch1-chn (-f ch1- …)**

**-c idem pour les caractères**

**$cut -f1,3 -d: /etc/passwd**

Centrale Marseille

# Qqs commandes – sort

**Trier selon certains champs: `$sort [opts]`**

```
$sort [opts] -k champ -t del [fic(s)]
```

`-n` : tri numerique et non alphanumerique

`-r`: reverse

```
$sort -n -k 3 -t: /etc/passwd
```

Centrale Marseille

# Stdin, stdout, stderr

Chaque commande “communique” au travers de 3 flux:
**`stdin, stdout, stderr`**

**`Stdout:`** Sortie standard.

Ou s'affiche le résultat d'une commande: en général sur le terminal.

**`Stderr:`** Sortie d'erreurs

Ou s'affichent les erreurs d'une commande: en général sur le terminal aussi.

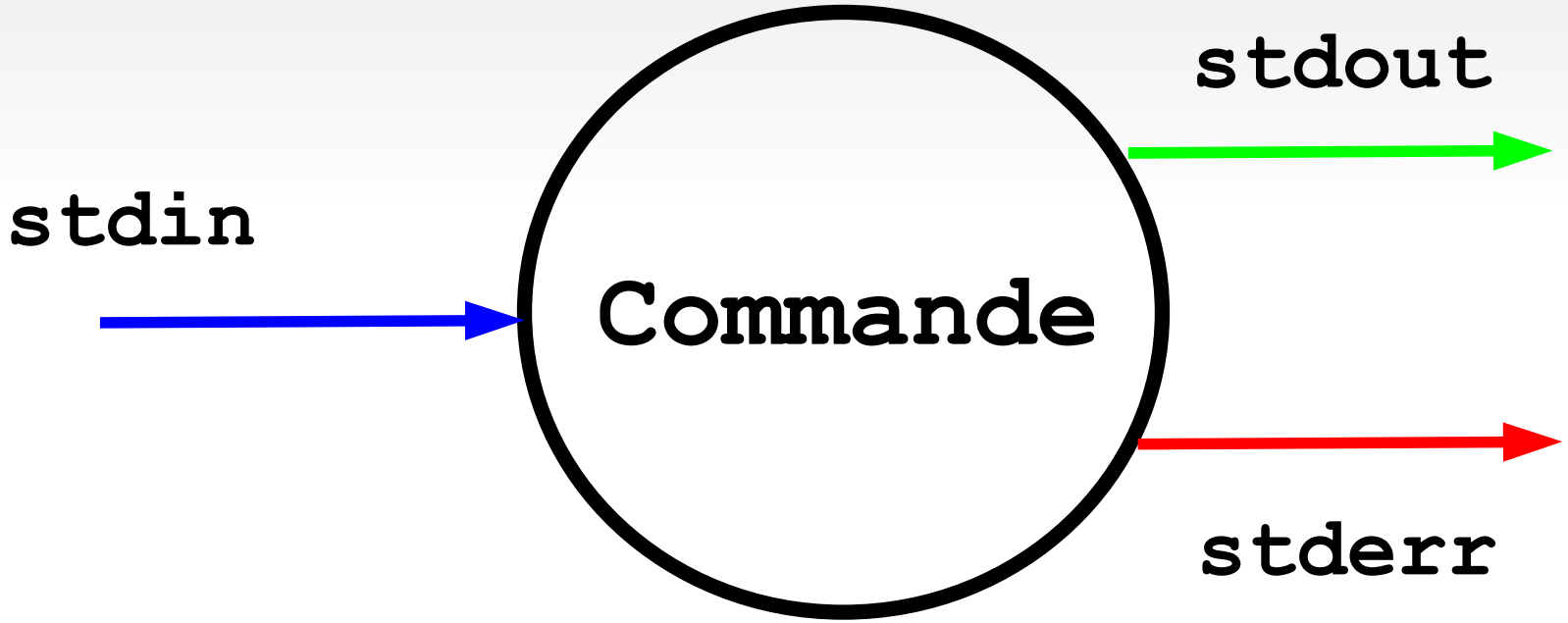
**`Stdin:`** Entrée standard.

En général le clavier.

Centrale Marseille

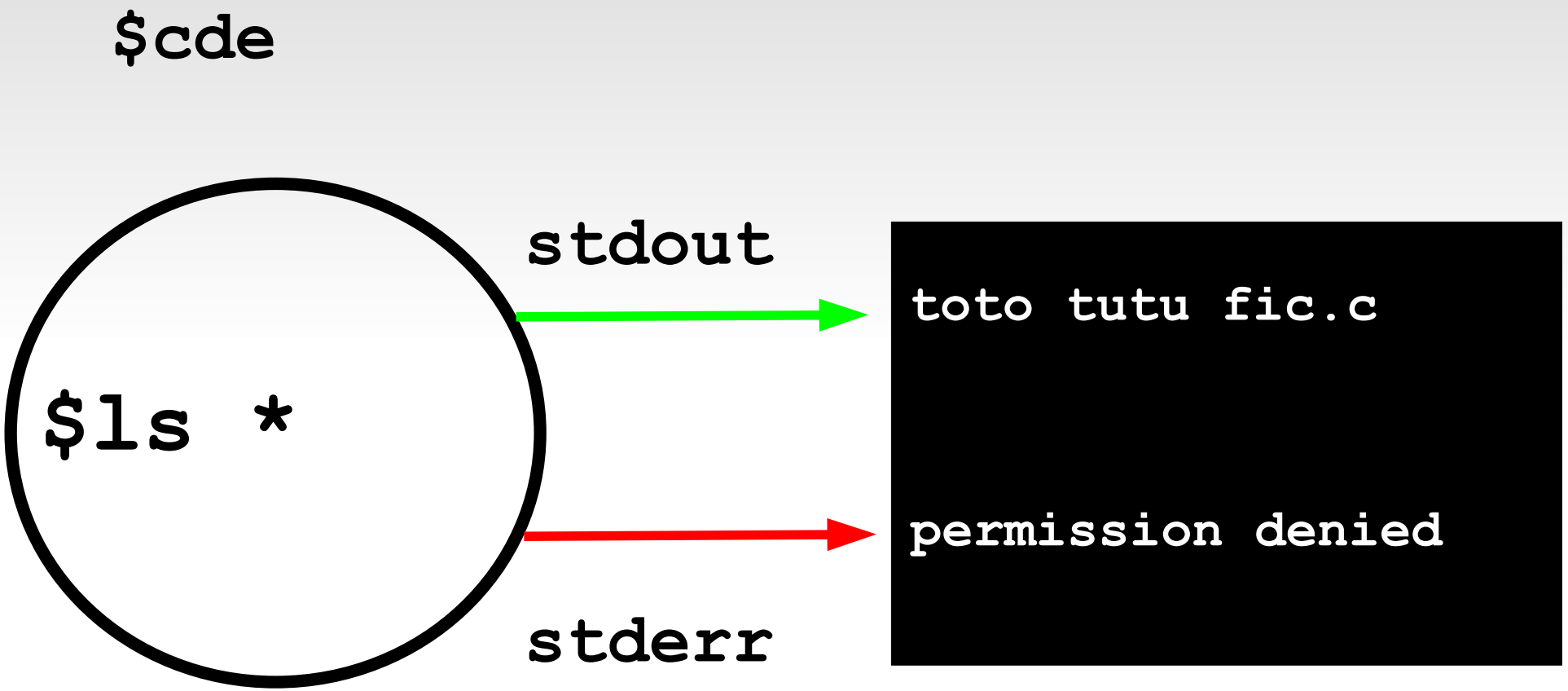
# stdin, stdout, stderr

Représentation



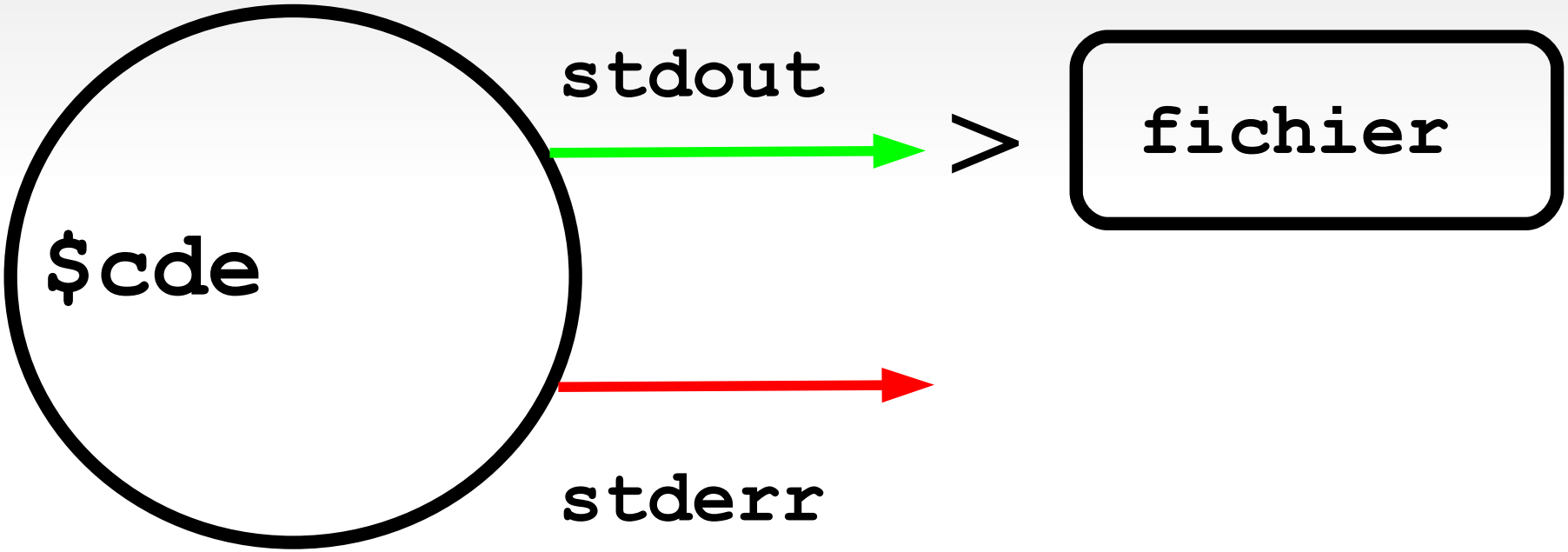

Centrale Marseille

# stdout, stderr



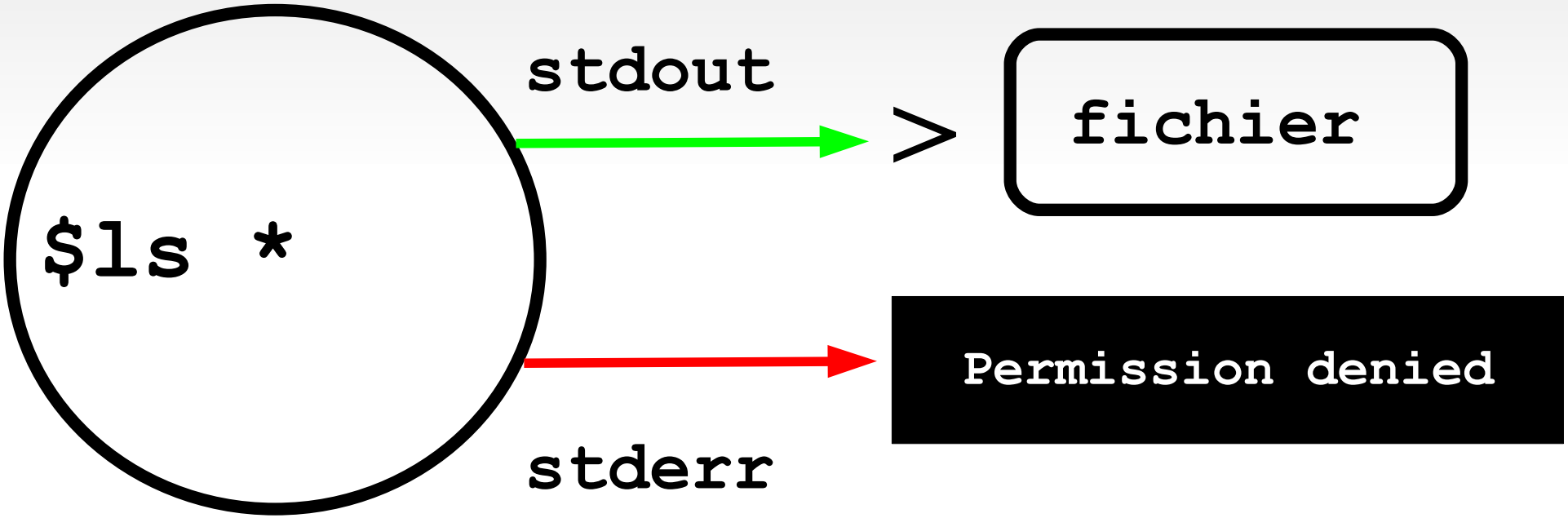
# redirection stdout

```
$ cde > fichier
```



# redirection stdout

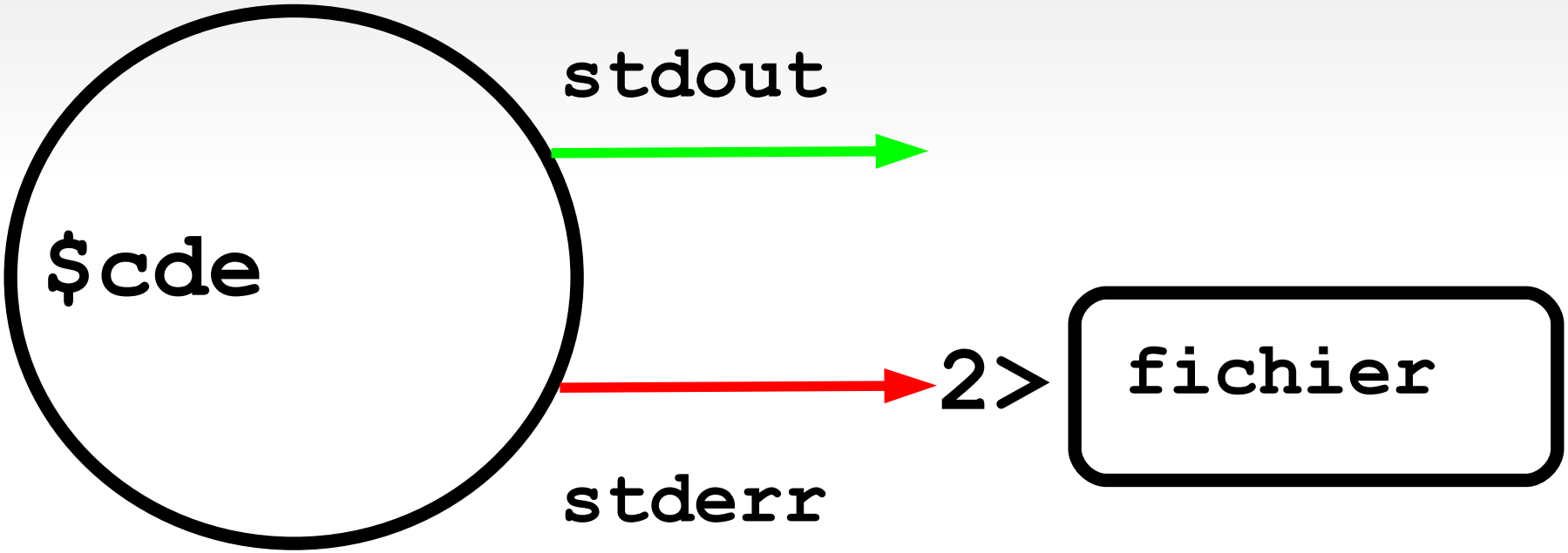
```
$ ls * > fichier
```




Remarque: >> concatène la sortie standard à un fichier existant

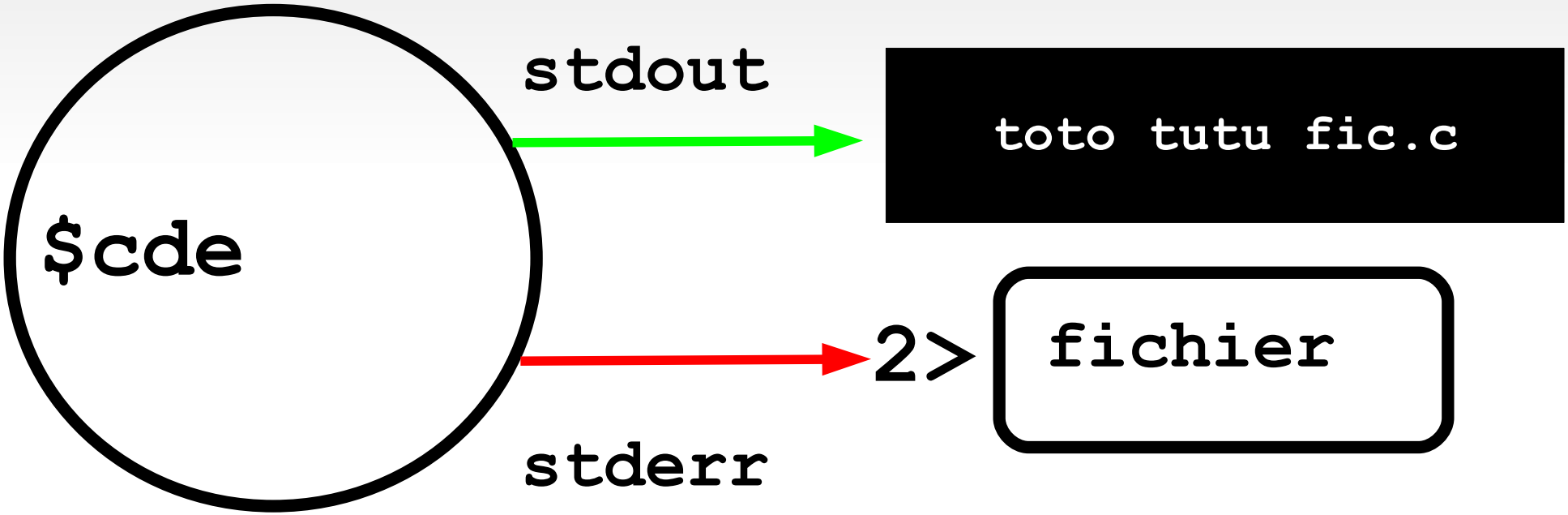
# redirection stderr

```
$ cde 2 > fichier
```



# redirection stderr

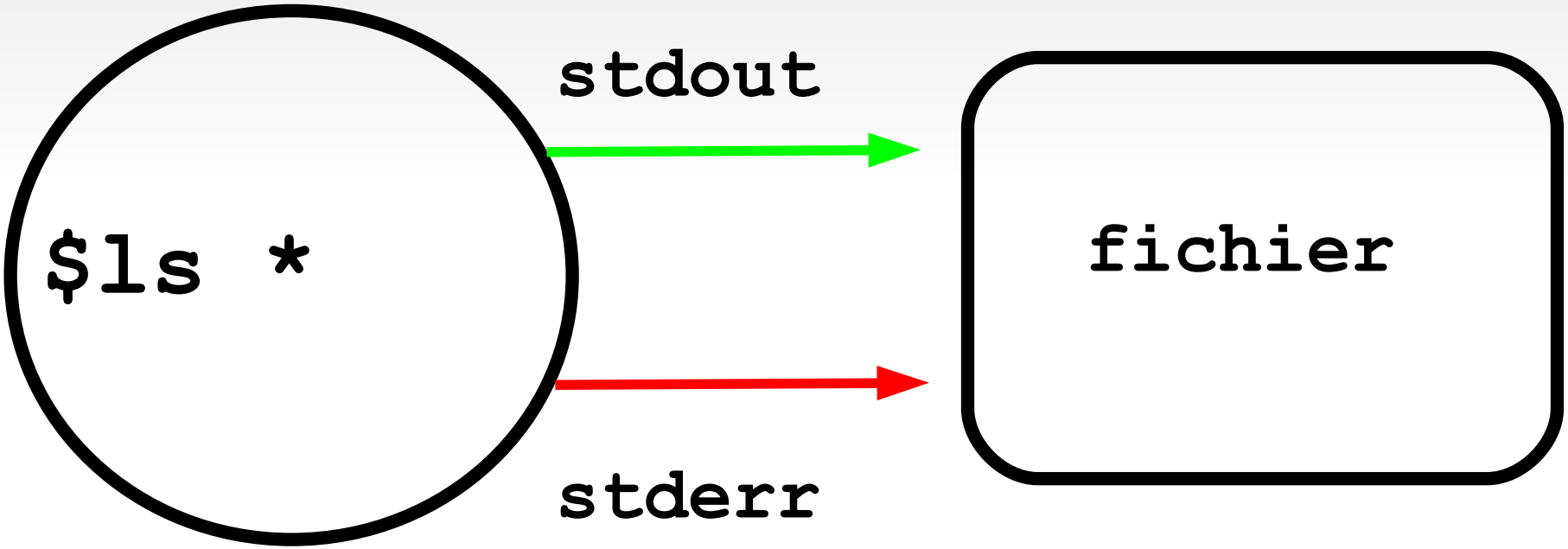
```
$ cde 2 > fichier
```




Centrale Marseille

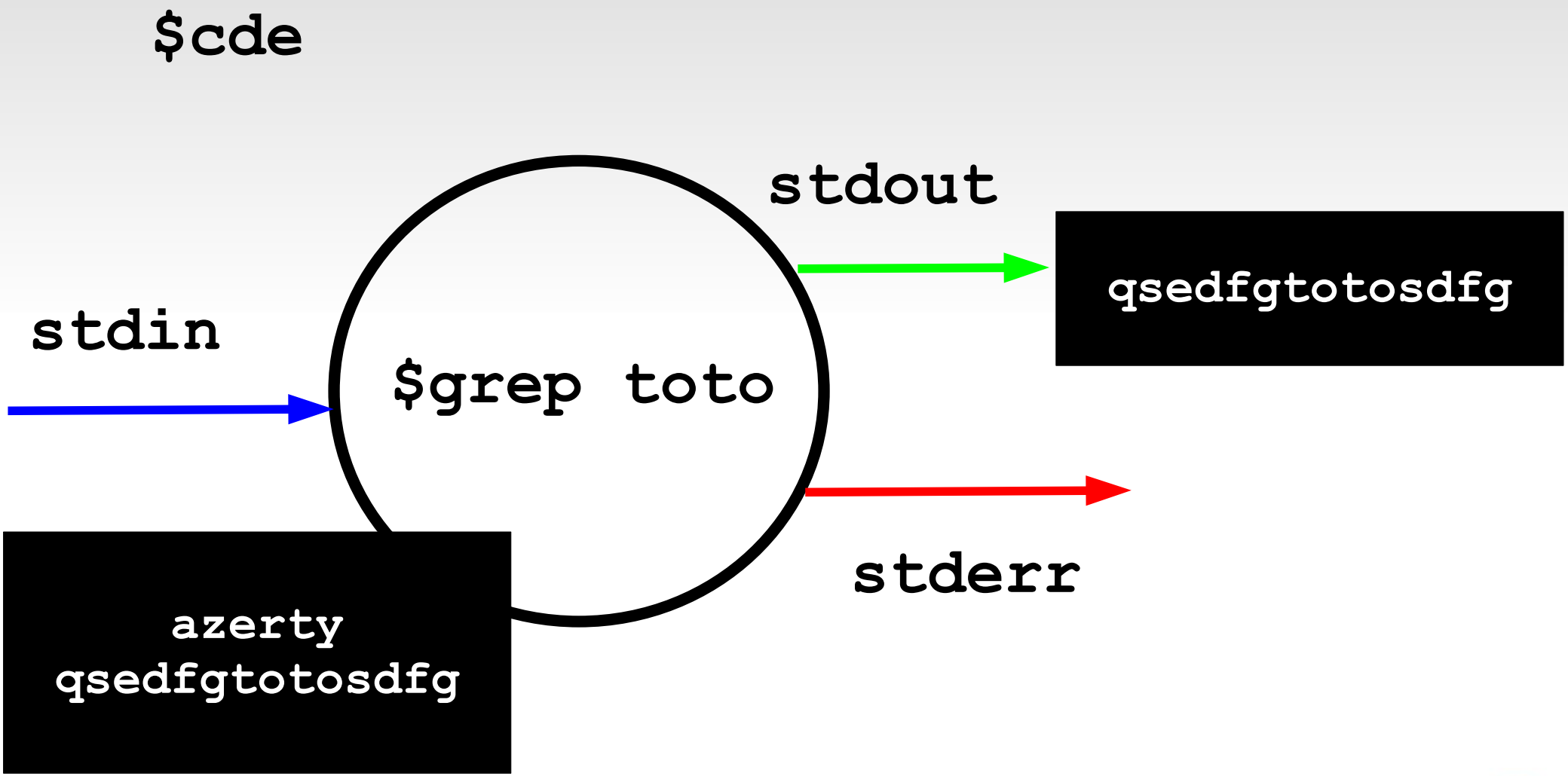
# redirection stdout

```
$ cde > fichier 2>&1
```




Centrale Marseille

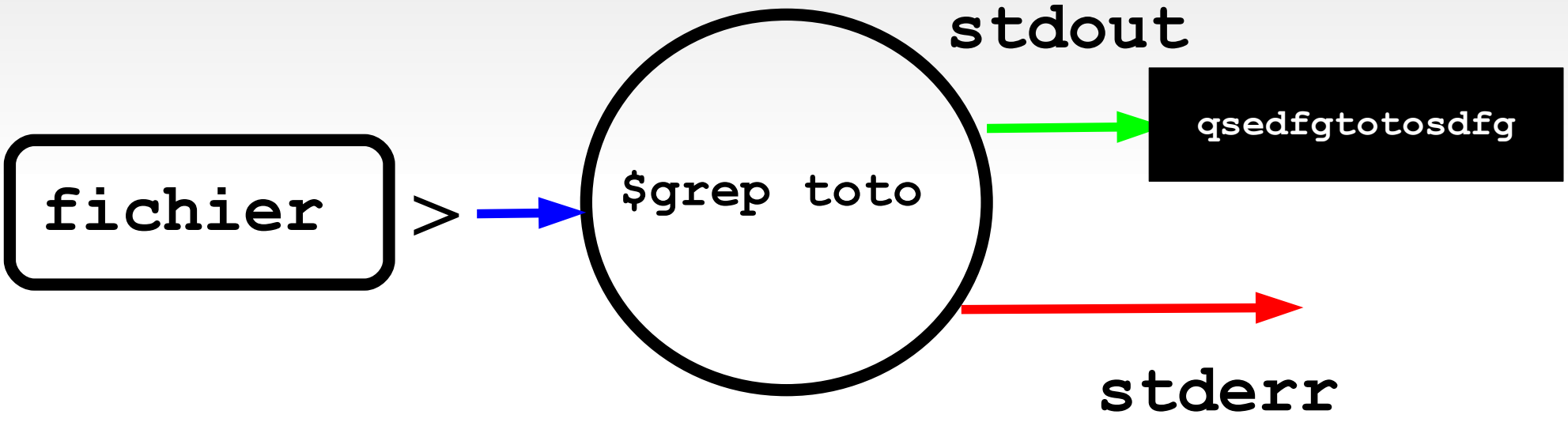
# stdin




Centrale Marseille

# stdin

```
$cde < fichier
```




Remarque: dans ce cas précis de commande

**`$grep toto < fichier <=> $grep toto fichier`**

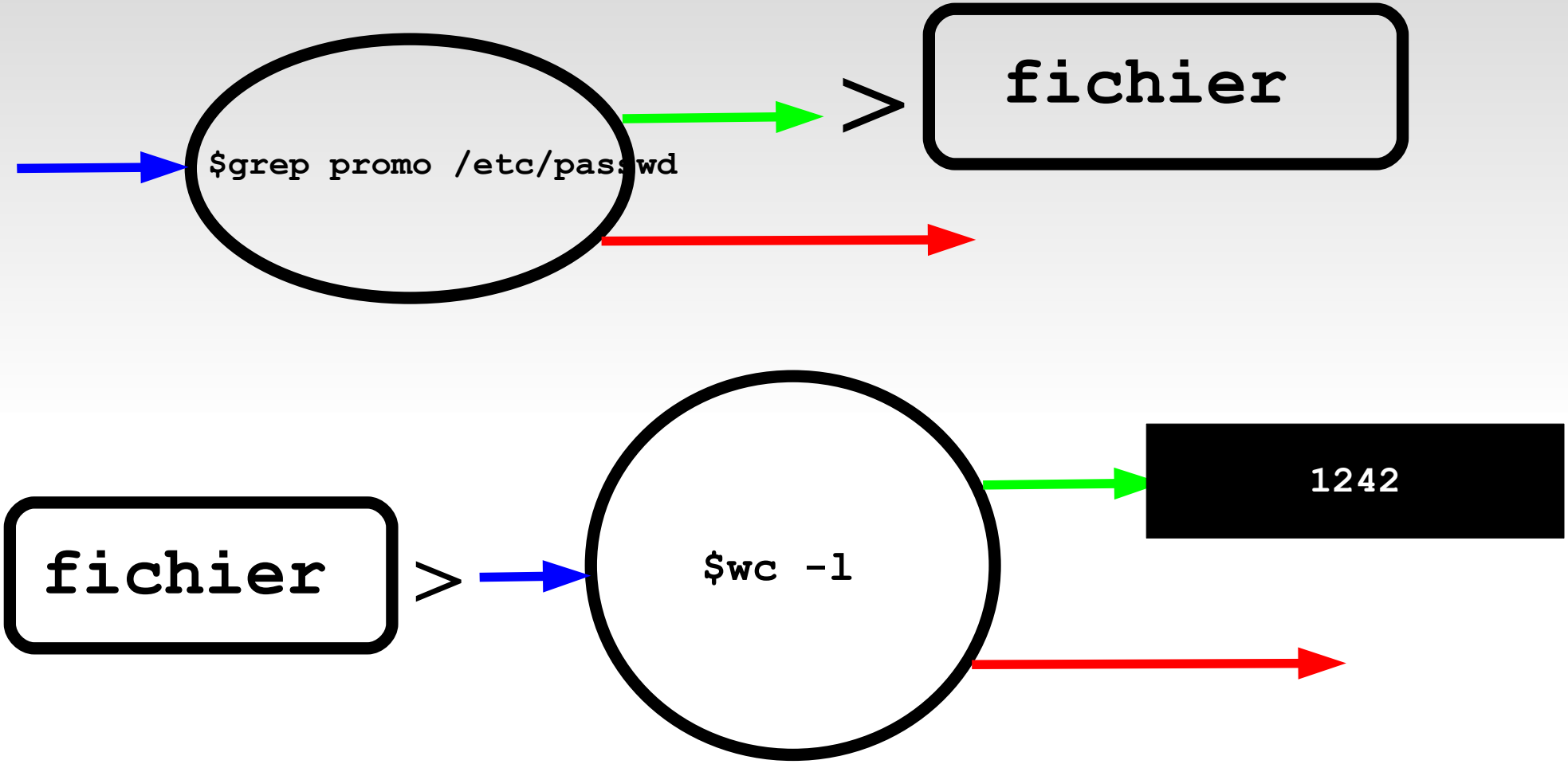
Centrale Marseille

# Qqs exemples:

```
$cat /etc/passwd
root:x:0:0:root:/root:/bin/zsh
dgeo:x:10102:13001:Desvernay Geoffroy:/users/info/dgeo:/bin/zsh
vajasse:x:30000:30001:ajasse virginie:/users/promo2006/vajasse:/bin/bash
rfortrie:x:20142:20001:Fortrie Remy:/users/prof/rfortrie:/bin/bash
mhamidou:x:20144:20001:Hamidou Mohammed:/users/prof/mhamidou:/bin/bash
...
$grep promo /etc/passwd > fichier
$wc -l fichier
1242
Rem: $wc -l < fichier
```

Centrale Marseille

# Représentation

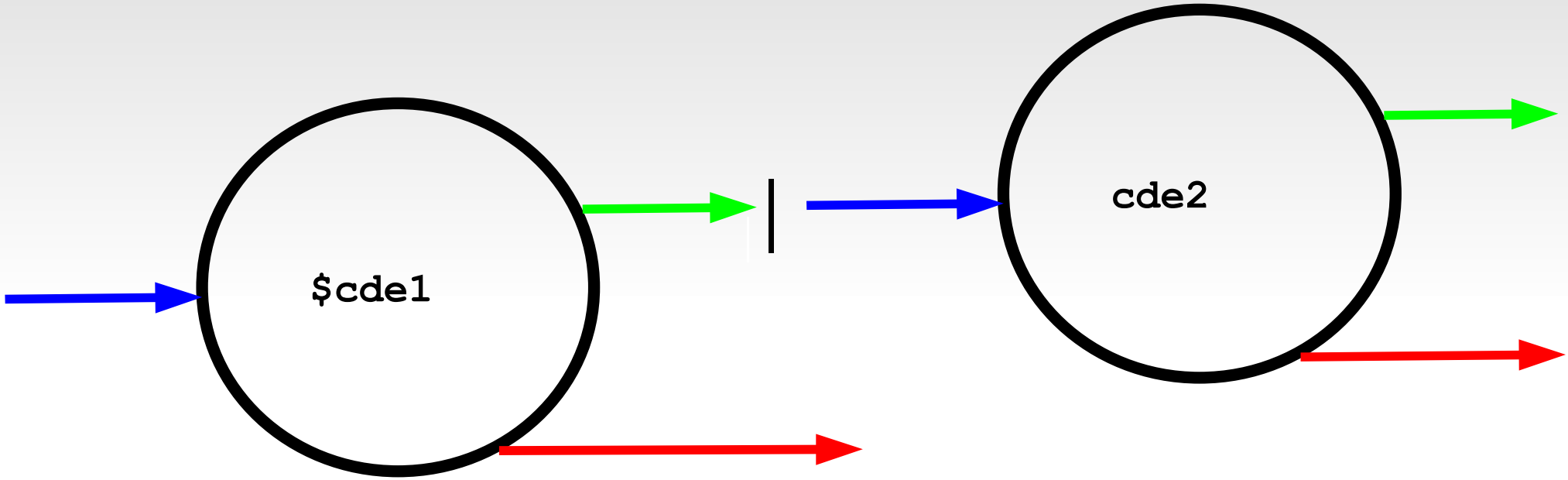



Remarque: Peut être peut on “brancher” directement la sortie standard de la première commande sur l'entrée standard de la seconde ?

Centrale Marseille

# Filtres & Pipes

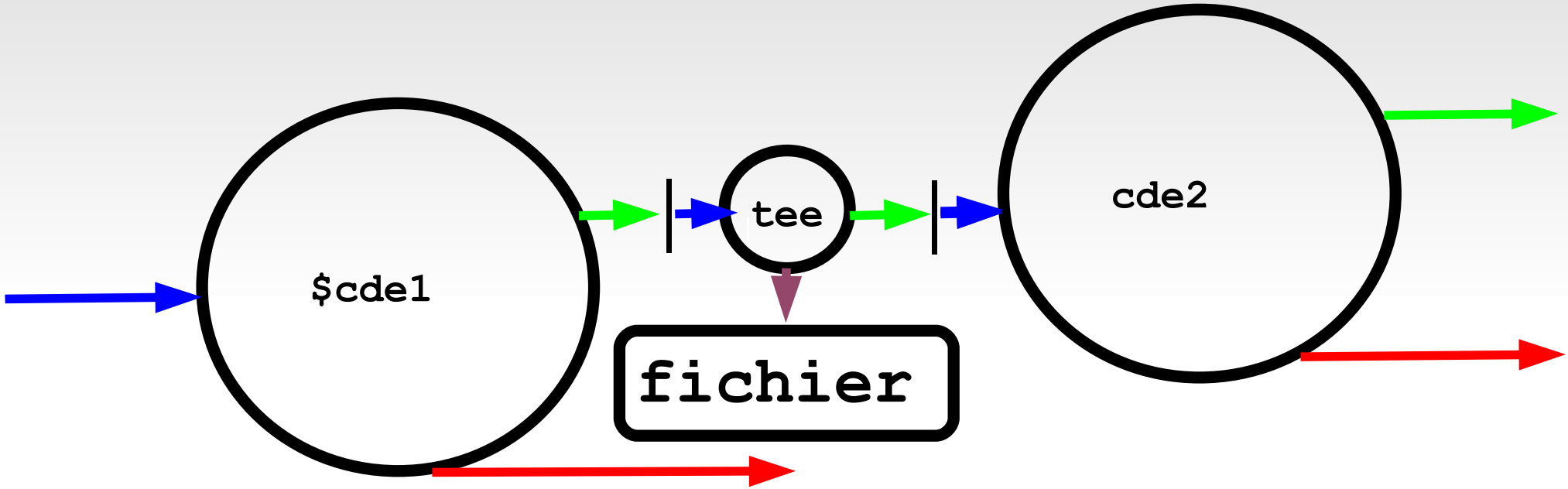
```
$cde1 | cde2
```




```
$grep promo /etc/passwd | wc -l
```

# Filtres & Pipes & derivations

```
$cde1 | tee fichier | cde2
```




```
$grep promo /etc/passwd | tee fichier |wc -l
```

Centrale Marseille