# Les filtres UNIX

## Modifier les données d'un fichier

### Coupe un fichier en morceau : split

La commande **split** permet de couper un fichier en morceau (en plusieurs fichiers), en tapant :

```
split -10 mon-fichier fichier
```

Vous allez créer les fichiers **fichieraa**, **fichierab**, **fichierac**, ... qui contiendront tous 10 lignes. Le premier **fichieraa** contient les 10 premières lignes, ainsi de suite.

### Trier des fichiers : sort

Soit le fichier **carnet-adresse** suivant :

```
maurice:29:0298334432:Crozon
marcel:13:0466342233:Marseille
robert:75:0144234452:Paris
yvonne:92:0133444335:Palaiseau
```

Le premier champ représente le nom, le deuxième le département, le troisième le numéro de téléphone et le dernier la ville. Attention le premier champ est noté 0, le deuxième 1, ainsi de suite.

En faisant **sort** sans argument :

```
sort carnet-adresse
```

Par défaut il va trier sur le premier caractère et ranger donc dans l'ordre alphabétique :

```
marcel:13:0466342233:Marseille
maurice:29:0298334432:Crozon
robert:75:0144234452:Paris
yvonne:92:013344433:Palaiseau
```

Si vous voulez trier sur le deuxième champ (le département), vous devez d'abord indiquer que le **:** est le caractère qui sépare deux champs (par défaut c'est l'espace), avec l'option **-t :**. Vous devez ensuite indiquer que vous trier un chiffre avec l'option **-n** (numérique). Pour indiquer qu'on veut trier le deuxième champ, il faut marquer qu'on veut trier à partir du second champ (**+1**) jusqu'au troisième (**-2**). Soit le résultat suivant ;

```
sort -n -t : +1 -2 carnet-adresse
```

On obtient :

```
marcel:13:0466342233:Marseille
maurice:29:0298334432:Crozon
robert:75:0144234452:Paris
yvonne:92:013344433:Palaiseau
```

Avec la commande :

```
sort -t : +3 -4 +0 carnet-adresse
```

Vous allez trier suivant le quatrième champ (numéro 3), c'est à dire la ville (tri par ordre alphabétique sur le premier caractère), en mettant **+0**, il va effectuer un deuxième tri pour les villes qui commencent par le même caractère, le deuxième tri porte sur le prénom (le premier caractère).

```
maurice:29:0298334432:Crozon
marcel:13:0466342233:Marseille
robert:75:0144234452:Paris
yvonne:92:013344433:Palaiseau
```

Les options de sort sont les suivantes :

- **-b** ignore les espaces et les tabulations en début de champ
- **-d** tri sur les caractères alphanumériques (caractères, chiffres et espace) uniquement
- **-r** inverse l'ordre de tri
- **-f** pas de différence entre minuscule et majuscule
- **-tx** Le caractère **x** est considéré comme séparateur de champ
- **-u** supprime les lignes doublons
- **-n** trie sur des chiffres

En tapant la commande suivante :

```
sort -t : +3.2 +0 carnet-adresse
```

Vous allez effectuer le tri sur le troisième caractère (numéro 2, le premier a pour numéro 0) du quatrième champ (numéro 3). En cas d'égalité du premier tri, on fait un dernier tri sur le premier caractère du prénom. Soit le résultat :

```
yvonne:92:013344433:Palaiseau
maurice:29:0298334432:Crozon
marcel:13:0466342233:Marseille
robert:75:0144234452:Paris
```

## Conversion de chaîne de caractère :tr

La commande **tr** permet de convertir une chaîne de caractère en une autre de taille égale. Les options sont les suivantes :

- **-c** Les caractères qui ne sont pas dans la chaîne d'origine sont convertis selon les caractères de la chaîne de destination
- **-d** destruction des caractères appartenant à la chaîne d'origine
- **-s** si la chaîne de destination contient une suite contiguë de caractères identiques, cette suite est réduite à un caractère unique

La commande **tr** a besoin qu'on lui redirige en entrée un fichier, le résultat de la conversion s'affichant sur la sortie standard.

Soit notre fichie**r carnet-adresse** :

```
maurice:29:0298334432:Crozon
marcel:13:0466342233:Marseille
robert:75:0144234452:Paris
yvonne:92:013344433:Palaiseau
```

Pour remplacer le **:** par un **#**, nous taperons :

```
tr " : " " # " < carnet-adresse
```

Pour faire la même chose on peut aussi bien éditer le fichier avec **cat** et rediriger par pipe vers **tr**, en

tapant :

```
cat carnet-adresse | tr " : " " # "
```

On peut utiliser des métacaractères. En tapant :

```
cat carnet-adresse | tr " [a-f] " " [A-F] "
```

Vous allez remplacer les caractères de **a** à **f** de minuscule en majuscule. Soit:

```
mAuriCE:29:0298334432:Crozon
mArCEl:13:0466342233:MArsEillE
robErt:75:0144234452:PAris
yvonnE:92:013344433:PAlAisEAu
```

# Edition de fichiers avec critères

## Editer un fichier par la fin : tail

Si vous avez un fichier très long, et que vous voulez visualiser que la fin, vous disposez de la commande **tail** :

La syntaxe est la suivante, si vous tapez :

```
tail +10 mon-fichier
```

Vous obtenez toutes les lignes du fichier de la 10eme jusqu'à la fin.

```
tail -10 mon-fichier
```

Vous obtenez les 10 dernières lignes à partir de la fin.

Vous pouvez indiquer si votre unité est la ligne (par défaut), le bloc ou le caractère avec l'option **-t**

```
tail -10 -c mon-fichier
```

Vous obtenez les 10 derniers caractères du fichier.

## Editer un fichier par le début : head

Si vous avez un fichier très long, et que vous voulez visualiser que le début, vous disposez de la commande **head** :

La syntaxe est la suivante, si vous tapez :

```
head +10 mon-fichier
```

Vous obtenez toutes les lignes du fichier de la 10eme jusqu'au début.

```
head -10 mon-fichier
```

Vous obtenez les 10 premières lignes à partir du début.

Vous pouvez indiquer si votre unité est la ligne (par défaut), le bloc ou le caractère avec l'option -t

```
head -10 -c mon-fichier
```

Vous obtenez les 10 premiers caractères du fichier.

## Compter les lignes d'un fichier : wc

La commande **wc** permet de compter le nombre de ligne d'un fichier, mais aussi le nombre de mot ou de caractères.

```
wc -l mon-fichier
```

Cette commande va donner le nombre de lignes contenues dans le fichier **mon-fichier**. Pour avoir le nombre de mot l'option est **-w**, l'option **-c** compte le nombre de caractères.

La commande **wc** sans option donne à la fois le nombre de ligne, le nombre de caractères et le nombre de mots.

Si vous voulez connaître le nombre de fichier dans un répertoire, la commande sera donc :

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

## Edition de champ d'un fichier : cut

La commande **cut** permet d'extraire certains champs d'un fichier. Les options sont les suivantes :

- **-c** extrait suivant le nombre de caractères
- **-f** extrait suivant le nombre de champs
- **-dx** Le caractère **x** est le séparateur de champ

Avec la commande **cut**, contrairement à sort, le premier champ a comme numéro 1, le deuxième 2 est ainsi de suite.

Nous prendrons toujours notre fichier **carnet-adresse** :

```
maurice:29:0298334432:Crozon
marcel:13:0466342233:Marseille
robert:75:0144234452:Paris
yvonne:92:013344433:Palaiseau
```

La commande :

```
cut -c-10 carnet adresse
```

Va extraire les 10 premiers caractères de chaque ligne, on obtient :

```
maurice:29
marcel:13:
robert:75:
yvonne:92:
```

La commande :

```
cut -c2-5 carnet adresse
```

Va extraire les deuxième au cinquième caractère de chaque ligne.

```
auri
arce
ober
vonn
```

La commande :

```
cut -c25-
```

Va extraire du 25eme caractère jusqu'à la fin de chaque ligne.

La commande :

```
cut -d: -f1,4 carnet adresse
```

Va extraire le premier et quatrième champ, le **:** fixant le séparateur de champ. On obtient :

```
maurice:Crozon
marcel:Marseille
robert:Paris
yvonne:Palaiseau
```

La commande :

```
cut -d : -f3- carnet adresse
```

Va extraire du troisième champ jusqu'au dernier champ, soit :

```
0298334432:Crozon
0466342233:Marseille
0144234452:Paris
0133444335:Palaiseau
```

## Fusion de fichier : paste

La commande **paste** permet la fusion de lignes de fichiers. Les options sont les suivantes :

- **-dx** Le caractère **x** définit le séparateur de champ
- **-s** Les lignes sont remplacées par des colonnes

Soit le fichier **carnet-adresse** :

```
maurice:29:0298334432:Crozon
marcel:13:0466342233:Marseille
robert:75:0144234452:Paris
yvonne:92:013344433:Palaiseau
```

Et le fichier travail :

```
ingénieur
pâtissier
facteur
vendeuse
```

En tapant la commande :

```
paste -d : carnet-adresse travail
```

Vous obtenez :

```
maurice:29:0298334432:Crozon:ingénieur
marcel:13:0466342233:Marseille:pâtissier
robert:75:0144234452:Paris:facteur
yvonne:92:013344433:Palaiseau:vendeuse
```

Vous pouvez évidemment rediriger le résultat vers un fichier.

## Extraction de lignes communes de deux fichiers : comm

Cette commande permet d'extraire les lignes communes à deux fichiers, soit le fichier **carnet-**

**adresse** :

```
maurice:29:0298334432:Crozon
marcel:13:0466342233:Marseille
robert:75:0144234452:Paris
yvonne:92:013344433:Palaiseau
```

Et **carnet-adresse2**

```
olivier:29:0298333242:Brest
marcel:13:0466342233:Marseille
myriam:30:0434214452:Nimes
yvonne:92:013344433:Palaiseau
```

La commande :

```
comm carnet-adresse carnet-adresse2
```

Nous donnera :

```
marcel:13:0466342233:Marseille
yvonne:92:013344433:Palaiseau
```

# Comparaison de fichiers

## Comparer deux fichiers : cmp

La commande **cmp** indique si deux fichiers sont identiques. En tapant :

```
cmp fichier1 fichier2
```

Si les deux sont identiques, la commande ne génère aucune sortie, s'ils sont différents la commande indique la position de la première différence (ligne et caractère), avec une sortie du genre :

```
fichier1 fichier2 differ : char 34, line 2
```

## Edition des différences entre deux fichiers : diff

Cette commande permet de rechercher les différences entre deux fichiers. La syntaxe est la suivante **diff fichier1 fichier2**, **diff** fait en sorte de vous donner des indications pour que le **fichier1** soit identique au **fichier2**. Soit le fichier **carnet-adresse** :

```
olivier:29:0298333242:Brest
marcel:13:0466342233:Marseille
myriam:30:0434214452:Nimes
yvonne:92:013344433:Palaiseau
toto:12:0434231122:Rodez
```

et **carnet-adresse2**

```
olivier:29:0298333242:Brest
marcel:13:0466342233:Gardagnes
myriam:30:0434214452:Nimes
yvonne:92:013344433:Palaiseau
```

La commande :

```
diff carnet-adresse carnet-adresse2
```

Génère comme sortie :

```
2c2
< marcel:13:0466342233 :Marseille
---
<marcel:13:0466342233 :Gardagnes
5d
>toto :12 :0434231122 :Rodez
```

Ce qui nous indique que pour **carnet-adresse** soit identique à **carnet-adresse2**, il faut que la deuxième ligne du premier fichier soit échangée (c pour change) contre la ligne du second. Il faut aussi supprimer (d pour delete) la cinquième ligne du premier fichier.

Dans d'autres exemples, on pourrait avoir aussi une sortie du genre 10,15c 12,17 ce qui signifie que pour que le premier fichier soit identique au second, les lignes 10 à 15 doivent intégralement échangées contre les lignes 12 à17 du second fichier.