Faculté de médecine de Rennes

UNIX : Introduction

Bruno Pouliquen (Bruno.Pouliquen@univ-rennes1.fr)
Denis Delamarre (Denis.Delamarre@univ-rennes1.fr)
Pierre Le Beux (Pierre.Lebeux@univ-rennes1.fr)

UNIX : Introduction

But 3

Historique 4

Connexion 5

Mot de passe 6

Syntaxe générale des commandes 7

Manuel en ligne : man 8

Quelques commandes : who, finger 9
date, cal 10

write 11

Mail 12

Système de fichier 14

Dénomination des fichiers 15

Structure du système de fichier 16

Chemins d'accès 17

Répertoire de travail 17

Quelques commandes : cd 18

mkdir, rmdir : Création/destruction de répertoires 19

ls 20

cat 21

Recherche : grep 22
egrep 23
sed 24

Manipulation des fichiers : copie 25
Liens 25
Mouvements : mv 26
Destruction : rm 26

Redirection d'entrée/sortie 27
Enchaînement de commandes: | 28

Protection des fichiers 29

Droits d'accès 30

Bibliographie 31

Principales commandes 32

But

UNIX : Système d'exploitation, c'est à dire :

Un logiciel permettant une utilisation efficace et commode d'un ordinateur.

- **Vision simplifiée du matériel**
- **Catalogue de fonctionnalités**
- **Exploite au mieux la puissance de la machine**

Historique

- 1969 : Ken Thompson et Dennis Ritchie créent un nouveau système inspiré de MULTICS (Multiplexed Information and Computing System).
- Version 4 écrite en langage C
- FAMILLE UNIX (nom déposé par ATT), environnement de temps partagé multi-utilisateurs et multi-processus.
- Famille BSD 4.3 (universitaire)
- Famille UNIX SYSTEME V (industriel)

Normalisation internationale menée par l'OSF (Open Software Foundation), regroupant les principaux constructeurs (IBM, DEC, Bull, HP ...).

Apparition d'un Unix gratuit : Linux

Nombreuses améliorations du système initial, notamment avec l'ajout de composantes :

- réseau : UUCP, puis TCP/IP (notamment le protocole NFS permettant de partager des ressources entre machines)
- interface : X11 (interface graphique), puis OSF/MOTIF
- langages : Les « shells » (Bourne-shell, Korn-shell, C-shell)

Connexion

Il faut se connecter au système UNIX via le protocole réseau « telnet » :
Le système procède alors à votre identification (« login ») :

Exemple :

```
telnet noemed.univ-rennes1.fr
login : dupond
password : milou
```

Remarques :

- Le nom est limité à 8 caractères,
- Le mot de passe également,
- Il y a différenciation des minuscules et majuscules,
- Le mot de passe n'apparaît évidement pas à l'écran

Lorsque la connexion est établie apparaît alors l'invite (« prompt ») de la machine :

```
noemed%
```

Ensuite pour se déconnecter il suffit de taper la commande UNIX « logout »

Mot de passe

Lors de la première connexion il est bon (et même obligatoire) de changer son mot de passe, qui vous sera personnel.

La séquence est la suivante :

```
noemed% passwd
Changing password for dupond on noemed.
Old password: milou
New password: loumi
Retype new password: loumi
```

A partir de ce moment le nouveau mot de passe est actif.

Remarques :

Afin d'éviter de se faire « craquer » son mot de passe, il est conseillé d'utiliser un mot de passe ayant les caractéristiques suivantes :

- Plus de 6 caractères
- Au moins deux caractères non alphabétiques (0,1,2...,9,&,',!,%,@, ...)
- Eviter les mots du langage courant

Exemples de mauvais mots de passe :

- milou
- ab
- azerty
- robert35
- 1234
- brest
- …

Syntaxe générale des commandes

Le système d'exploitation UNIX offre à l'utilisateur un ensemble de commandes, l'appel de ces commandes répond toujours à la même syntaxe :

`commande [options] [paramètres]`

La commande est un mot clé en minuscules où toute faute de frappe constitue une erreur de syntaxe et provoque donc un message d'erreur (bien faire attention aux espaces).

Exemple :

```
noemed% faitcequejeveux
faitcequejeveux:Command not found
```

Elle (ou il ?) n'a rien compris !

Une commande peut être limitée à un mot clé ou peut être suivie d'options et/ou de paramètres facultatifs.

Les paramètres et les options obéissent également à une certaine syntaxe.

Pour les options sur toutes les commandes Unix, elles sont en règle générale toujours précédées d'un tiret (-) et suivies d'au moins un caractère. Il peut y avoir plusieurs options.

Exemple :

```
noemed% ls -a
```

(**l**i**s**t **a**ll : lister tout)

Paramètres:

Pour les paramètres ou arguments, il n'y a pas de syntaxe générale:

Certaines commandes ne comporte pas de paramètres.

D'autres en ont un obligatoire (nom de fichier par exemple).

D'autres ont des paramètres facultatifs avec une syntaxe précise.

Manuel en ligne : man

Sous UNIX chaque commande est documentée. Cette documentation est accessible à l'aide de la commande « man » (pour manuel) suivi de la commande dont on veut consulter la documentation.

Exemple :

```
noemed% man cat
CAT(1V)         USER COMMANDS           CAT(1V)

NAME
     cat - concatenate and display

SYNOPSIS
     cat [ - ] [ -benstuv ] [ filename...  ]

DESCRIPTION
     cat reads each filename in sequence and displays it
on  the standard output.  Thus:
(...)
```

La commande cat permet de visualiser le contenu d'un ou plusieurs fichiers.

Quelques commandes : who, finger

Information sur les utilisateurs :

Qui est là et qui êtes vous : (who)

```
noemed% who
dupond   ttyp0   Feb 18 11:44   (lichen)
seka     ttyp1   Feb 18 07:37   (plb)
poulique ttyp3   Feb 18 08:54   (dim)
picault  ttyp4   Feb 18 09:08   (asterix)
```

On peut également demander qui on est :

```
noemed% whoami
dupond
```

ou

```
noemed% who am i
noemed!dupond  ttyp0   Feb 18 11:44 (lichen)
```

La commande finger est quasiment identique :

```
noemed% finger
Login       Name             TTY Idle   When    Where
dupond   Dupond Albert (DES  p0    2 Wed 11:44 lichen
seka     Seka Louis-Paul     p1 5:03 Wed 07:37 plb
poulique Pouliquen Bruno- ex p3 2:22 Wed 08:54 dim
picault  Picault Christelle  p4 3:29 Wed 09:08 asterix
```

Mais elle donne plus d'informations sur un utilisateur particulier :

```
noemed% finger picault
Login name: picault             In real life: Picault
Christelle (Stage DEA Image-IA Caen)
Directory: /users/picault             Shell: /bin/csh
On since Feb 18 09:08:36 on ttyp4 from asterix
3 hours 33 minutes Idle Time
Mail last read Wed Feb 18 09:09:25 1997
No Plan.
```

date, cal

Consulter la date du jour : date

Pour obtenir la date et l'heure en anglais

```
noemed% date
Thu Sep 25 15:43:46 WET 1997
```

La date standard est en anglais et sous forme : jour, mois, n° du jour, heure, zone horaire et année

La commande Calendrier (cal) :

Cette commande permet de visualiser le calendrier d'une année ou d'un mois en anglais

syntaxe :

```
cal [no-jour no-mois] année
```

Exemple :

```
noemed% cal 1 96
   January 96
 S  M Tu  W Th  F  S
                1  2
 3  4  5  6  7  8  9
10 11 12 13 14 15 16
17 18 19 20 21 22 23
24 25 26 27 28 29 30
31
```

On peut en principe visualiser les calendriers de l'an 1 jusqu'à l'an 9999. Vous pouvez ainsi connaître le jour de la semaine de votre naissance !

write

Envoi d'un message directement à un autre utilisateur connecté.

```
noemed% write poulique
Est-ce qu'on peut utiliser l'imprimante ?
<Control-D>
```

<Control-D> représente pour UNIX le caractère de fin de fichier

L'utilisateur à qui est destiné ce message le verra apparaître sur son écran...

En retour, il peut répondre :

```
message from poulique@noemed on ttyp0 at 15:55 ...
Non, elle n'est pas encore configurée...
```

Mail

Messagerie électronique permettant la communication par boîte aux lettres entre utilisateurs

Envoi de mail :
syntaxe : **mailx *utilisateur***
Subject: ***Sujet du message***
corps du message
<un point en début de ligne>

Exemple :

```
noemed% mailx dupond
Subject: Impression
La configuration de l'imprimante est finie
vous pourrez l'utiliser demain
.
```

Lecture de sa boîte aux lettre :

mailx (sans paramètre)
Si aucun message n'est présent dans la boîte aux lettres il affiche :

```
No mail for dupond
```

Sinon il affiche la liste des messages (uniquement les sujets), chaque message ayant un numéro il suffit de taper ce numéro pour lire son contenu...

```
noemed% mailx
Mail version SMI 4.0 Thu Jul 23 13:52:20 PDT 1992  Type ? for
help.
"/usr/spool/mail/dupond": 2 messages 2 new
>N  1 administ@noemed.univ-rennes1.fr Wed Feb 18 16:14   10/381
Commande ls
 N  2 poulique@noemed.univ-rennes1.fr Wed Feb 18 16:15   10/355
Impression
? _
```

mail : suite

Pour lire le premier message :

```
& 1
Message  1:
From administ Wed Feb 18 16:14:36 1997
Received: by noemed.univ-rennes1.fr; Wed, 18 Feb 97
16:14:35 GMT (4.1/VERsept92)
Date: Wed, 18 Feb 97 16:14:35 GMT
From: administ@noemed.univ-rennes1.fr (Administrateur
des Macintosh )
To: dupond@noemed.univ-rennes1.fr
Subject: Commande ls
Status: R

Est-ce que tu connais l'option permettant de lister
toutes les informations
dans la commande ls ?
```

Pour répondre directement à un message : r

```
& r
To: administ@noemed.univ-rennes1.fr
Subject: Re:  Commande ls
Non, je n'en sais rien du tout !
.
```

Pour supprimer le message courant : d

Pour quitter le programme mailx : q

Pour quitter sans toucher aux messages : x

ATTENTION: selon les unix ce programme s'appelle mail ou mailx

Système de fichier

Qu'est-ce qu'un fichier :

Un fichier est une suite non structurée de caractères, stockée sur une mémoire auxiliaire.

En fait pour UNIX tout est fichier :

- Un fichier de données (exemple : la liste des étudiants)
- Un programme (Toutes les commandes UNIX sont des fichiers)
- Un répertoire (qui permet de regrouper des fichiers)
- Un périphérique (l'écran ou le clavier par exemple)
- Un disque dur, une disquette, un CD-ROM
- Un lien vers un autre fichier
- Un lien vers un fichier éloigné (via le réseau).

Le système de fichiers est un ensemble de logiciels système qui gère tous les fichiers connus de la machine.
Pour gérer et retrouver ces fichiers, il faut des **répertoires ("directory"** en **anglais)**

Quand on dit que le système de fichiers est hiérarchisé, cela veut dire qu'il peut y avoir plusieurs niveaux de répertoires qui permettent de se déplacer dans un arbre de fichiers-répertoire pour arriver aux "feuilles" qui correspondent aux fichiers de données proprement dit.

Enfin, pour le système de fichiers, tout élément peut être protégé (en lecture, écriture ou exécution) voire inaccessible (protégé en lecture/écriture).

Dénomination des fichiers

Chaque fichier doit posséder un nom unique : un nom de fichier est caractérisé par une chaîne de caractères allant jusqu'à 255 caractères : lettres, chiffres, points ("."), trait de soulignement ("_")...

ATTENTION :
- les **MAJUSCULES** sont distinguées des **minuscules**
- **Eviter de commencer par les caractères spéciaux** et le blanc (espace) pour les noms de fichiers.
- il n'y a **pas de gestion de versions** de fichiers, il faut donc le gérer soi-même en changeant de noms: par exemple garder le nom suivi d'un numéro de version : lettre1, lettre2, ou datées...
- les caractères **. ou ..** correspondent à des fichiers. On peut utiliser le point comme séparateur, celui-ci est généralement réservé pour indiquer le type du fichier:

    lettre.txt, programme.c, triangle.pas

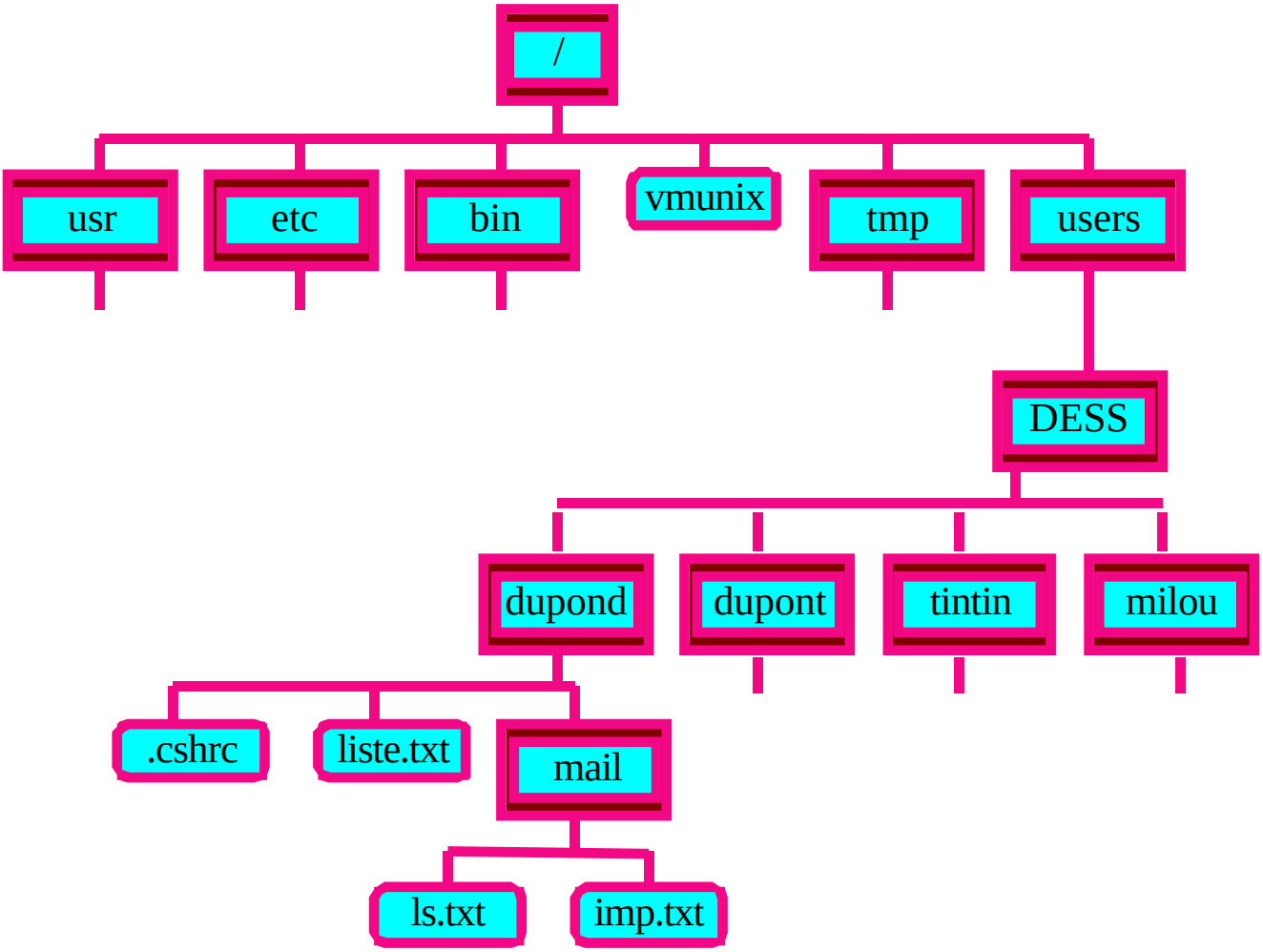
Remarque :

On peut référencer plusieurs fichiers en utilisant le caractère * :

`*.txt`	désigne	`liste.txt, toto.txt, titi.2.1.txt`
`t*.txt`	désigne	`toto.txt, titi.2.1.txt`
`t*`	désigne	`toto.txt, titi.2.1.txt, temporaire…`

Structure du système de fichier

La structure est arborescente: la racine est le répertoire /, les nœuds sont les répertoires et les feuilles sont, en général, les fichiers



Chemins d'accès

LES NOMS DE CHEMINS : (**pathname**)

Il suffit de donner les noms des fichiers-répertoire de chaque niveau séparé par des caractères / (slash).

Exemples :

/bin 1er niveau du répertoire bin
/users/DESS 2ème niveau : répertoire DESS.
/users/DESS/dupond/mail/imp.txt : Le fichier imp.txt se trouvant dans le répertoire mail du répertoire dupond du répertoire DESS du répertoire users...

Répertoire de travail

ANNUAIRE DE TRAVAIL ET ANNUAIRE PERSONNEL

Lorsque l'on se connecte sur une machine UNIX (**login)** l'utilisateur est directement placé au niveau de son répertoire personnel (home directory):

```
/users/DESS/vous
```

Ce répertoire devient alors votre répertoire de travail.

Pour connaître votre répertoire de travail utiliser la commande pwd (**p**rint **w**orking **d**irectory):

```
noemed% pwd
/users/DESS/dupond
```

Quelques commandes : cd

Changement de répertoire de travail (**c**hange-**d**ir)
Permet de se positionner à un autre endroit de la hiérarchie.

Exemple :

```
noemed% pwd
/users/DESS/dupond
noemed% cd /tmp
noemed% pwd
/tmp
```

cd sans aucun paramètre vous repositionne sur votre répertoire personnel

```
noemed% cd
noemed% pwd
/users/DESS/dupond
```

Chemin absolu, chemin relatif

Pour dénommer un fichier on peut utiliser la dénomination absolue, c'est à dire en partant de la racine (/) :
`/users/DESS/dupond/mail/imp.txt`
Mais si notre répertoire de travail est `/users/DESS/dupond` il suffira de nommer ce fichier :
`mail/imp.txt` (nommage relatif)
Et si le répertoire de travail est `/users/DESS/dupond/mail:` `imp.txt`

Répertoire spécial : `..`
Les deux points signifient « répertoire père »
Répertoire très utile pour les chemins relatifs :
`cd ../milou` devient équivalent à `cd /users/DESS/milou`

mkdir, rmdir : Création/destruction de répertoires

mkdir : Création d'un nouveau répertoire (**m**a**k**e **dir**ectory).

Exemple: création d'un répertoire sous le répertoire de travail
```
noemed% mkdir temporaire
```
ou
```
noemed% mkdir /users/DESS/dupond/temporaire
```

rmdir : Suppression d'un répertoire (qui doit être vide)

Exemple:
```
noemed% rmdir temporaire
```

ls

Liste le contenu du répertoire (**l**i**s**t)

ls sans paramètre liste le répertoire de travail
ls *répertoire* : liste le contenu du répertoire donné

Exemple :

```
noemed% ls /users/DESS/dupond
liste.txt       mail
```

Liste de tous les fichiers : (- a pour all)

```
noemed% ls -a
.               ..                  .cshrc
liste.txt       mail
```

On trouve ici les fichiers `.` et `..` utilisés par le système pour retrouver le répertoire courant et le répertoire père. De même, le fichier **.cshrc** est affiché (fichier servant à des initialisations automatiques).

Si l'on désire toutes les informations sur les fichiers standards, on utilisera l'option `-l` (**l**ong)

```
noemed% ls -l
```

On peut cumuler l'option `-l` avec l'option `-a`

```
noemed% ls -al    (ou ls -a -l)
drwxr-xr-x  3 dupond       512 Feb 18 17:47 .
drwxr-xr-x  2 root        7168 Feb 18 17:54 ..
-rw-r--r--  1 dupond       733 Feb 18 17:45 .cshrc
-rw-r--r--  1 dupond       327 Feb 18 17:47 liste.txt
drwxr-xr-x  2 dupond       512 Feb 18 17:48 mail
```

Ceci signifie (en résumé) que le fichier liste.txt fait 327 octets (soit 327 caractères pour un fichier de texte), et qu'il a été modifié pour la dernière fois le 18 février à 17h 47.

cat

Visualisation de fichiers (con**cat**enate)

syntaxe : `cat` ***noms-de-fichier***

Exemples :

```
cat liste.txt
cat mail/imp.txt mail/ls.txt
```

Lecture de portions de fichiers **(head, tail, more)**

```
head [-n] fichier
tail [-n] fichier
more fichier
```

Note :

Les crochets [ ] indiquent qu'il s'agit d'un paramètre facultatif (ici -n représente le nombre de lignes à visualiser). S'il n'est pas présent, il y a visualisation des 8 premières (ou dernières) lignes.

Exemples:

```
noemed% head -3 liste.txt
Bonnefoy Isabelle
Channac Bertrand
Collet Jean-Yves

noemed% tail -4 liste.txt
Sebti Mohammed
Tiennot-Trebaol Dominique
Tillard Eric
Turmel Valerie

noemed% more mail/*.txt
```

UNIX : Introduction

Recherche : grep

Syntaxe : **grep** [*options*] *motif* [*fichiers*]

Le motif peut être une simple chaîne de caractères, mais peut être également une expression contenant des caractères spéciaux

Exemple :

```
noemed% grep 'Isabelle' liste.txt
Bonnefoy Isabelle
Kauffer Isabelle
```

Quelques caractères spéciaux :

. : signifie n'importe quel caractère
^ : En début de motif signifie « commence par »
$: En fin de motif signifie « finit par »

Exemples plus complexes :

Les noms commençant par K :

```
noemed% grep '^K' liste.txt
Kauffer Isabelle
Kervran Stephane
```

Les prénoms finissant par 'ie' :

```
noemed% grep 'ie$' liste.txt
Delangle Stephanie
Lecor Nathalie
Muller Sylvie
Turmel Valerie
```

Les noms de plus de 10 caractères :

```
noemed% grep '^.......... ' liste.txt
Lebouteiller Rachel
```

Options : **-i** (pas de différenciation majuscule/minuscule)
-v (affiche les lignes qui ne sont pas reconnues)

Remarque :
On est parfois ammenés à rechercher des expressions plus complexes.
On utilisera alors un outil associé : egrep

egrep

Recherche d'expression régulière :

syntaxe: **egrep** *'expression' fichier*

Une expression régulière (regexp) est un *motif* de recherche, constitué de :

- Un caractère `egrep 'A'` Toutes les lignes contenant un A majuscule
- Un ensemble de caractères
  - [a-z] tout caractère alphabétique
  - [aeiouy] toute voyelle
  - [a-zA-Z0-9] tout caractère alphanumérique
- Un caractère spécial
  - **.** n'importe quel caractère
  - \n caractère « retour-chariot »
  - \t tabulation
  - \b espace
  - ^ début de ligne `egrep '^[0-9]'` lignes commençant par un chiffre
  - $ fin de ligne `egrep '\b$'` Lignes finissant par un espace
- Quelques opérateurs :
  - **?** 0 ou 1 fois `egrep 'anti\b?bacterien'` (avec ou sans espace)
  - ***** 0 ou n fois `egrep 'anti\b*bacterien'` (avec espaces ou sans)
  - **+** 1 ou n fois `egrep '\b[0-9]+\b'` (Un chiffre entouré d'espaces)
  - **()** parenthèses `egrep ' ([AGCT][AGCT][AGCT]-?)+ '` (reconnaît les chaînes d'ADN : ACA-AGC-AAA, mais pas CAGE ou TATA !)
  - **|** ou (inclusif) `egrep 'anti\b*(bacterien|biotique)'`

Exemples :

`egrep ' [0-9]+ *[Ff]( |$)' mon_texte` retrouve toutes les lignes contenant un prix (4F, 51 F, 123 f, ...)

`egrep '(^| )[Ll]e *[Bb]eux( |$)' rep_telephone`
retrouver « Le Beux » (mot entier, avec ou sans la première lettre majuscule, avec ou sans espace).

sed

Recherche et remplacement :

syntaxe: **`sed`** `'s/`*`expression/remplace`*`/g'` *`fichier`*

L'expression est du même type que celle définie pour egrep.

`sed` `'s/antibiotique/antibacterien/g'` Remplace toute occurrence de « antibiotique » par « antibacterien ».

options :
`sed` `'s/`*`expression/remplace`*`/'` `ne remplace qu'une fois par ligne.`

`sed` `'s/...(`*`exp2`*`).../...\1.../'` le symbole `\1` signifie que l'on garde le contenu de ce qui était dans la parenthèse.

Exemples :

`sed` `'s/lebeux/le beux/g' annuaire` On force un espace

`sed` `'s/beurre doux/beurre sale/g'` Traducteur normand->breton

`sed` `'s/mot de passe.*//g' doc` Enlève les mots de passe de la doc

`sed` `'s/tel 99/tel 02.99/g' annuaire > nvannuaire`
Pour le passage en nouvelle numérotation
Problème pour la ligne « untel 99 ans » remplacée à tord. Et pour « téléphoner au 99.28.42.15 » qui n'est pas remplacée

`sed` `'s/99(.[0-9][0-9])+/02.99\1/g' annuaire > nvannuaire`
Même chose que précédemment mais plus robuste.

Manipulation des fichiers : copie

Commande cp (**c**o**p**y)
Syntaxe : **`cp`** *`fichier nouveau-fichier`*
Exemple :

```
noemed% cp liste.txt nouvelle_liste.txt
noemed% ls -l
total 3
-rw-r--r--  1 dupond 309 Feb 19 09:52 liste.txt
drwxr-xr-x  2 dupond 512 Feb 19 09:12 mail
-rw-r--r--  1 dupond 309 Feb 19 10:06 nouvelle_liste.txt
```

Le système a bien créé un nouveau fichier avec les mêmes caractéristiques que l'ancien

Liens

Commande ln (**l**i**n**k) :
Syntaxe : **`ln -s`** ***`fichier référence`***

Permet de faire une copie logique du fichier, en général on utilise les liens symboliques (option -s) :

```
noemed% ln -s liste.txt lst
noemed% ls -l
total 4
-rw-r--r--  1 dupond 309 Feb 19 09:52 liste.txt
lrwxrwxrwx  1 dupond   9 Feb 19 10:10 lst -> liste.txt
drwxr-xr-x  2 dupond 512 Feb 19 09:12 mail
-rw-r--r--  1 dupond 309 Feb 19 10:06 nouvelle_liste.txt
```

Le fichier lst n'est qu'un fichier virtuel, qui ne contient que la référence vers le fichier réel. Avantage : Toute modification du fichier d'origine est répercutée sur le fichier virtuel.
Les liens sont souvent utilisés pour les répertoires:
/users/DESS est un lien vers /pub/users/DESS

Mouvements : mv

Commande mv (**m**o**v**e)
Syntaxe : **mv *ancien nouveau***

Deux utilisations essentielles : renommer un fichier , et déplacer un fichier d'un répertoire vers un autre.

Exemples :
Renommer un fichier :

```
noemed% mv nouvelle_liste.txt nvliste.txt
noemed% ls -l
total 4
-rw-r--r--  1 dupond  309 Feb 19 09:52 liste.txt
lrwxrwxrwx  1 dupond    9 Feb 19 10:10 lst -> liste.txt
drwxr-xr-x  2 dupond  512 Feb 19 09:12 mail
-rw-r--r--  1 dupond  309 Feb 19 10:06 nvliste.txt
```

Déplacer un fichier :

```
noemed% mv mail/imp.txt .
```

Le fichier imp.txt est replacé dans le répertoire courant.

Cette commande aurait pu s'écrire:

```
noemed% cd mail
noemed% mv imp.txt ..
```

Destruction : rm

Commande rm (**r**e**m**ove)
syntaxe : **rm *fichiers***
Exemple :

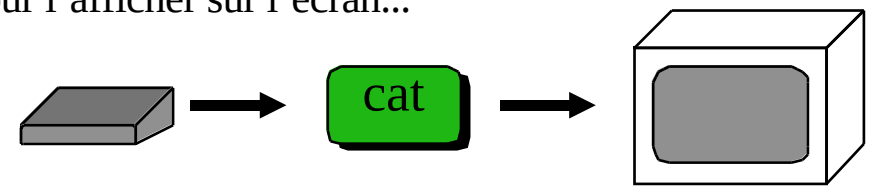
```
noemed% rm nvliste.txt lst
```

UNIX : Introduction

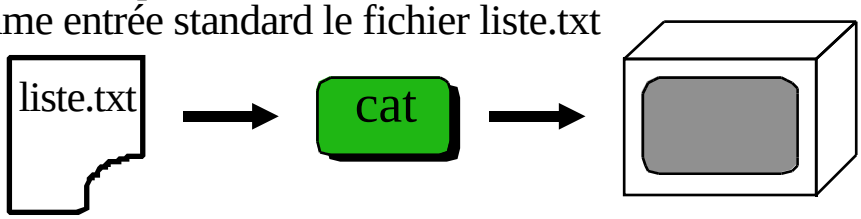
Redirection d'entrée/sortie

Tout processus (programme) sous UNIX possède deux organes de communication standards appelés « Entrée standard » et « Sortie standard ».

Exemple : la commande **cat** prend tout caractère tapé au clavier pour l'afficher sur l'écran...



Quand on tape la commande `cat liste.txt`, la commande prend comme entrée standard le fichier liste.txt

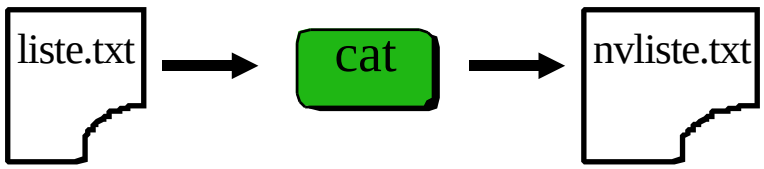


Pour détourner l'entrée standard on utilise le caractère <, pour la sortie standard le caractère >.

`cat liste.txt` est équivalente à `cat < liste.txt`

Pour copier le fichier liste.txt on peut utiliser la commande :

```
cat < liste.txt > nvliste.txt
```



On peut également créer un nouveau fichier en utilisant la commande :

```
cat > fichier.txt
```

Remarque :

On peut également rajouter des lignes dans un fichier en utilisant la commande : **cat >> fichier.txt**

Enchaînement de commandes: |

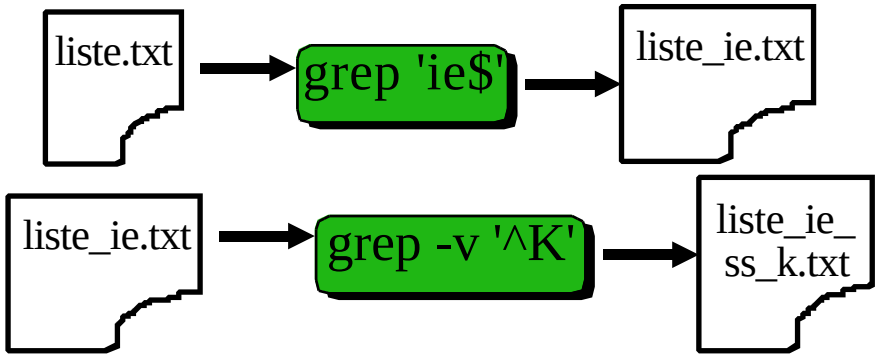
On est souvent amené à enchaîner plusieurs commandes UNIX. Par exemple : avec la commande grep on voudrait connaître toutes les lignes qui finissent par "ie" mais qui ne commencent pas par K.

Deux solutions :
1) Deux commandes et un fichier intermédiaire

```
grep 'ie$' liste.txt > liste_ie.txt
```

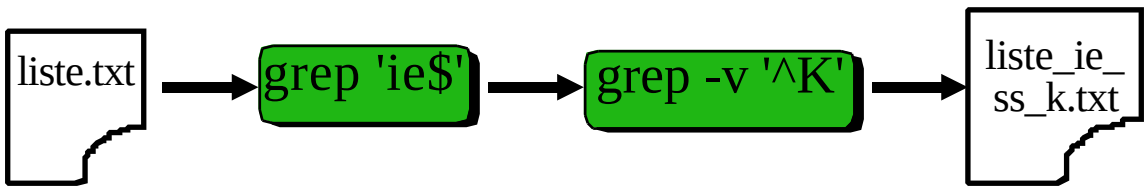
puis

```
grep -v '^K' liste_ie.txt > liste_ie_ss_k.txt
```



2) Utilisation du "pipe" (caractère "|")permettant de rediriger la sortie standard d'une commande vers l'entrée standard de la suivante.

```
grep 'ie$' liste.txt | grep -v '^K' > liste_ie_ss_k.txt
```



Protection des fichiers

Le système de fichiers UNIX gère des droits d'accès caractérisés par trois possibilités d'utilisation :

- Lecture (R pour **r**ead)
- Ecriture (W pour **w**rite)
- Exécution (X pour e**x**ecute)

Pour un répertoire, X indique la possibilité d'entrer dans ce répertoire

Chaque fichier est caractérisé par trois triplets d'accès qui décrivent les droits.

- du propriétaire (u pour **u**sers)
- du groupe (g pour **g**roupe)
- des autres (o pour **o**thers)

Les droits associés à un fichier apparaissent lorsque l'on utilise la commande ls -l.
Exemple :

```
noemed% ls -lg
```
(g pour lister également le groupe possesseur)

```
total 2
-rw-r--r-- 1 dupond dess97 309 Feb 19 09:52 liste.txt
drwxr-x--- 2 dupond dess97 512 Feb 19 10:22 mail
```

Le type de fichier est caractérisé par **d** pour un répertoire et **-** (tiret) pour un fichier normal.

On s'apperçoit que le fichier liste.txt est lisible par tout le monde (utilisateur, groupe et autres), mais qu'il n'est modifiable que par l'usager dupond.
Par contre le répertoire mail n'est pas lisible pour les autres utilisateurs.
Il est lisible par les usagers du groupe dess97 `rwxr-x---`, mais pas modifiable.
L'usager dupond a tous les droits sur le répertoire mail : `rwxr-x---`

Droits d'accès

Cette opération est possible par le propriétaire du fichier grâce à la commande chmod (**ch**ange-**mod**e) :

syntaxe: **chmod** [*droits*] *fichier*

Les droits sont précisés sous forme de paramètres : u (**u**sers), g (**g**roup), o (**o**thers), a (**a**ll) avec :

- \+ Droit accordé
- \- Droit enlevé
- = Droit fixé

Suivi du droit associé :

- r (lecture)
- w (écriture)
- x (exécution)

Exemples :

```
noemed% chmod o-r liste.txt
noemed% ls -l liste.txt
-rw-r----- 1 dupond         309 Feb 19 09:52 liste.txt

noemed% chmod a=rw liste.txt
noemed% ls -l liste.txt
-rw-rw-rw-  1 dupond         309 Feb 19 09:52 liste.txt

noemed% chmod g+w mail/*
noemed% ls -l mail
total 2
-rw-rw-r--  1 dupond         365 Feb 19 09:12 imp.txt
-rw-rw-r--  1 dupond         391 Feb 19 09:12 ls.txt
```

Bibliographie

(sommaire)

UNIX pour l'utilisateur : Commande et langages de commandes, J.L. Nebut, édition technip.
· Description de l'interface entre l'utilisateur et le système

La programmation sous UNIX, J.-M. Rifflet, ed. Mc Graw-Hill
· Présentation des principales caractéristiques et des appels système.

Les bases de l'administration système, Æleen Frisch, 2e Edition, avril 1996, ed. O'reilly, ISBN : 2-84177-008-7, 768 pages, 320F.

UNIX, une introduction en bref, A. Strohmeier & P. Kipfer, PPUR 1993.

Le système UNIX (traduction), S. Bourne, 1985, InterEditions.

L'environnement de programmation UNIX (traduction), B. W. Kernighan & R. Pike, 1986, InterEditions.

UNIX : introduction, B. Pouliquen, P. Le Beux, D. Delamarre, 1997, disponible sous Web à l'adresse (URL) : http://www.med.univ-rennes1.fr/~poulique/cours/unix

Guide UNIX, Marc Schaefer, 1994
URL : http://www.pasteur.fr/other/computer/unix/unixguide_html

Linux : documentations en Français
http://uhp.u-nancy.fr/linux/docs-france.html

Linux in a nutshell (manuel de référence, version française), J. P. Hekman, 1997, ed O'Reilly, (isbn: 2-84177-031-1)

UNIX : Introduction

Principales commandes

Commande	Explication	page
cal	Calendrier	10
cat	affichage d'un fichier	21
cd	changement de répertoire	18
chmod	changement des droits sur un fichier	30
cp	copie de fichier	25
date	affiche la date courante	10
df	Espace disponible sur les partitions (file system) ("df -k ." : connaître l'espace encore disponible)	
diff	affichage de différences entre deux fichiers ("diff fichier1 fichier2")	
du	Espace utilisé (y compris dans les sous-répertoires)	
egrep	recherche de motif dans un fichier	23
emacs	éditeur pleine page	
file	affiche le type de chaque fichier	
find	recherche de fichiers (find . -name toto.txt)	
finger	information sur les utilisateurs connectés	9
ftp	transfert de fichiers entre machines	
grep	recherche d'une chaîne de caractères dans un fichier	22
gzip	compression et décompression de fichiers	
head	affichage des premières lignes d'un fichier	21
history	rappel des dernières commandes	
ln	créer un lien d'un fichier à l'autre	25
logout	déconnexion	5
ls	contenu d'un répertoire	20
Mailx (ou **mail**)	lecture ou envoi de courrier électronique	12
man	manuel en ligne	8
mkdir	création d'un répertoire	19
more	affichage page par page d'un fichier	21
mv	déplacement, renommer un fichier	26
passwd	changement de mot de passe	6
pwd	affichage du répertoire courant	17
ps	liste des processus	
quota	espace disque disponible	
rm	destruction d'un fichier	26
rmdir	destruction d'un répertoire	19
sed	recherche et remplacement	24
tail	affichage des dernières lignes d'un fichier	21
talk	discussion en direct entre utilisateurs	
tar	gestion d'archives portables	
telnet	connexion à distance	5
wc	comptage du nombre de lignes, de mots ou de caractères d'un fichier	
who	liste des utilisateurs connectés	9
whoami	qui suis-je ?	9
write	envoi d'un message à un utilisateur	11