Les commandes de base UNIX

Corrigé : TP 6 Pipes, jokers et redirections

La commande cat

1. **Comment visualiser deux fichiers l'un après l'autre à l'aide de la commande cat ?**

   Il suffit de taper les deux noms de fichiers à la suite; par exemple, pour visualier bla puis blo, on tape : ***cat bla blo***

2. **Comment faire une copie d'un fichier sans utiliser cp ni ouvrir d'éditeur de texte ?**

   On demande à cat d'afficher le contenu d'un fichier, puis de placer le résultat dans un fichier :

   cat toto > copie

3. **Utiliser cat pour écrire un peu de texte et mettre le résultat dans un fichier notes.**

   On demande à cat de rediriger sa sortie dans un fichier, puis on n'a plus qu'à taper le texte et à indiquer la fin du texte avec ^D :

   ```
   nacre ~ $ ls
   maitrise.tex
   nacre ~ $ cat > notes
   Faire la bibliographie
   Revoir le chapitre 1.
   ^D
   nacre ~ $ ls
   maitrise.tex    notes
   nacre ~ $ cat notes
   Faire la bibliographie
   Revoir le chapitre 1.
   nacre ~ $
   ```

   C'est bien sûr un moyen un peu spartiate pour écrire du texte, mais dans le cas de choses très courtes dans ce style, ce peut être plus rapide que de lancer un véritable éditeur.

4. **Quelle différence y a-t-il entre**

   cat bla blo > blu **et** cat bla blo >> blu

   Que se passe-t-il, pour chaque ligne, selon que le fichier blu existe ou n'existe pas ?

   cat bla blo > blu

   concatène les deux fichiers bla et blo, et place le résultat dans un nouveau fichier appelé blu. Si blu existe déjà, le shell affiche un message d'erreur et ne fait rien.

   cat bla blo >> blu place la concaténation de bla et blo à la fin d'un fichier blu déjà existant. S'il n'existe pas, le shell affiche un message d'erreur et en reste là, sans créer de fichier blu.

   Les deux redirections ne sont donc pas du tout équivalentes, mais leur action dépend du shell, qui a des options qui modifie leur comportement par défaut.

5. **Comment obtenir un fichier blo qui corresponde à un fichier bla dont les lignes seraient désormais numérotées ?**

   Il faut utiliser l'option -n de cat. Par exemple :

   ```
   nacre ~ $ cat bla
   Pomme
   Poire
   Prune
   nacre ~ $ cat -n bla > blo
   nacre ~ $ cat blo
        1  Pomme
        2  Poire
        3  Prune
   nacre ~ $
   ```

Jokers et expressions régulières

1. **Vous avez chez vous des fichiers appelés essai1, essai2, essai3 et essai4. Comment les effacer en une seule ligne de commande ?**

   Il s'agit de donner le nom des fichiers en indiquant qu'ils commencent par essai et qu'ils finissent par un chiffre. Il y a en fait plusieurs façons de faire :

   - rm essai[1234] : les crochets signifient «l'un des caractères qui se trouvent entre crochets»;
   - rm essai[1-4] : c'est le même principe, sauf que l'on indique un intervalle au lieu d'écrire explicitement les chiffres;
   - rm essai? : le point d'interrogation désigne n'importe quel caractère unique, le point y compris, sauf si le nom du fichier commence par un point (ls ?pine dans $HOME ne vous donnera pas, par exemple, la liste des fichiers de configuration de Pine : .pinerc, etc).

     Ici, on trouvera tous les fichiers commençant par essai, suivi d'un unique caractère qui peut ne pas être un chiffre.
   - rm essai* : c'est la formulation la plus vague : effacer les fichiers dont le nom commence par essai suivi de quelque chose, c'est-à-dire d'un nombre indéfini de caractères, ou aucun.

2. **Dans mon répertoire d'accueil, j'ai un certain nombre de fichiers avec un suffixe .c. Je désire les regrouper dans un répertoire que j'appelerai C/. Quelles sont les commandes que je dois taper ?**

   On commande par créer un répertoire C avec mkdir, puis, avec mv, on déplace tous les fichiers ayant un suffixe . et dont le nom est fait d'une suite de caractères quelconques :

   ```
   nacre ~ $  ls
   hello*    zoinx*
   hello.c   zoinx.c
   nacre ~ $  mkdir C
   nacre ~ $  mv *.c C/
   nacre ~ $  ls
   C/        hello*     zoinx*
   nacre ~ $  ls C/
   hello.c   zoinx.c
   ```

Questions subsidiaires

3. **Vous désirez regrouper dans un répertoire Rangement les fichiers dont le nom contient un caractère minuscule suivi d'un caractère majuscule. Quelle(s) est/sont la/les commande(s) à donner ?**

   On commence par créer le répertoire Rangement avec mkdir. Pour désigner les noms des fichiers, il faut indiquer la notion de «minuscule». On pourrait écrire explicitement l'alphabet entre crochets, pour dire «l'un de ces caractères», puis faire de même avec les majuscules. Mais on gagne du temps en utilisant des intervalles ([a-z] et [A-Z]). Le reste du nom du fichier, avant et après la minuscule puis la majuscule, est indéfini. On écrit donc :

   nacre ~ $ mkdir Rangement

   nacre ~ $ mv *[a-z][A-Z]* Rangement/

4. **Même chose avec les fichiers dont le nom contient trois voyelles à la suite.**

   Le principe est le même, sauf que l'on indique explicitement les voyelles entre crochets :

   nacre ~ $ mkdir Rangement

   nacre ~ $ mv *[aeiou][aeiou][aeiou]* Rangement/

5. **En utilisant ls et grep, affichez la liste des fichiers dans /bin dont le nom correspond à un motif donné.**

   On peut procéder de deux façons : utiliser ls seul et des jokers, ou rediriger ls dans grep et utiliser les expressions régulières de grep.

    Attention  Dans la suite du corrigé, on suppose que l'on se trouve déjà dans /bin/.

   On met des apostrophes autour des expressions de grep pour les protéger contre le shell. Enfin, on ne détaille pas les expressions régulières; rappel des points importants à propos de grep :

   - Le chapeau (^) en début d'expression désigne le début de la ligne;
   - Le dollar en fin d'expression désigne la fin de la ligne;
   - Le point désigne n'importe quel caractère;
   - L'étoile signifie «zéro ou plus de fois le caractère qui précède»; c'est un multiplicateur;
   - Comme avec les jokers du shell, [^m] veut dire «sauf le caractère m».

   On constate que grep est plus complexe mais bien plus puissant que les jokers du shell.

	Avec ls seul	Avec ls et grep
Commence par «a» et dont la deuxième lettre est «s» ou «t»	ls a[st]*	ls | grep '^a[st].*'
Contient «un» et se termine par «t»	ls *un*t	ls | grep '.*un.*t$'
Contient «gre» ou «st»	ls *(gre|st)*	ls | grep '\(gre|st\)'
Contient exactement deux lettres «m»		ls | grep '[^m]*m[^m]*m[^m]*'
Contient au moins deux lettres «m»		ls | grep '.*m.*m.*'
Contient au moins quatre caractères et aucun chiffre		ls | grep '^[^0-9]\{4,\}$'
Est constitué de deux lettres exactement	ls ??	ls | grep '^..$'
Commence et finit par un chiffre	ls [0-9]*[0-9]	ls | grep '^[0-9].*[0-9]$'

6. **Comment éliminer les lignes vides dans un fichier ? Comment éliminer les lignes ne contenant que des blancs ? Comment éliminer toutes les lignes blanches ?**

   Une ligne vide est différente d'une ligne ne contenant que des blancs, c'est-à-dire des espaces ou des tabulations, même si pour un oeil humain cela revient au même. La commande de recherche sera différente selon le cas :

   - **Lignes vides** : c'est «rien entre le début et la fin de la ligne»; on écrit donc :
     grep '^$' *fichier*
   - **Lignes ne contenant que des blancs** : on utilise une classe de caractère préexistante : [:space:]. On écrit donc :
     grep '^[[:space:]]$' *fichier*

   Pour éliminer toutes les lignes blanches pour un oeil humain, on combine les deux expressions et on utilise l'option -v qui inverse le sens de la recherche. On n'a plus qu'à rediriger la sortie dans un fichier. On écrit donc :

   grep -v '\(^[[:space:]]$\|^$\)' fichier1 > fichier2

   en remplaçant éventuellement le motif par '^$' ou ^[[:space:]]$' selon que l'on veut ôter les lignes vides ou les lignes contenant des blancs.

head et tail

1. **Récupérer les lignes 5 à 9 d'un fichier de 12 lignes**. Il y a deux solutions :

   1. Prendre les neuf premières lignes, et n'en garder que les cinq dernières (de 5 à 9 inclus) : head -9 fichier | tail -5
   2. Prendre les 8 dernières lignes (de 5 à 12 inclus) et n'en garder que les 5 premières :
      tail -8 fichier | head -5

2. **Comment afficher la cinquième ligne d'un fichier ?**

   On utilise la commande head pour extraire les cinq premières lignes du fichier, puis la commande tail pour ne conserver que la dernière des cinq : head -5 fichier | tail -1

3. **Affichez les 15 premières lignes du fichier /etc/hosts, les 15 dernières lignes, toutes les lignes à partir de la quinzième, les lignes 15 à 20.**

   - Afficher les 15 premières lignes de /etc/hosts : head -15 /etc/hosts
   - Afficher les 15 dernières lignes de /etc/hosts : tail -15 /etc/hosts
   - Afficher toutes les lignes à partir de la quinzième : tail +15 /etc/hosts
   - Affichez les lignes 15 à 20 : head -20 /etc/hosts | tail -6

   Remarque à propos du fichier /etc/hosts : les ordinateurs sur l'Internet sont désignés par une adresse IP, constituée de 4 nombres entre 0 et 255 séparés par des points. C'est cette adresse IP qui permet à un ordinateur d'envoyer un message (datagramme IP) à un autre.

   Cependant, mémoriser les adresses IP n'est pas commode pour un humain. Pour cette raison, les ordinateurs ont aussi un «nom», constitué d'un nombre variable de composantes séparées par des points (par exemple, research.att.com).

   Le mécanisme de conversion d'un nom en adresse IP (ou le mécanisme inverse) s'appelle la «résolution de noms». Elle se fait normalement par l'intermédiaire de *nameservers*, ordinateurs sur lequel tourne le programme BIND, qui se chargent de répondre aux questions de résolution de noms.

   Cependant, pour certaines machines très communément accédées depuis un ordinateur donné (par exemple, les machines de la même salle) ou bien pour éviter d'avoir à configurer un nameserver, on préfère parfois utiliser le fichier /etc/hosts : celui-ci définit un certain nombre de conversions nom -> adresse IP qui seront consultées avant les *nameservers*.

La bibliothèque qui se charge de faire la traduction, et qui consulte d'abord le /etc/hosts puis les *nameservers*, s'appelle la libresolv.

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.

Soit le 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
```

Filtres et redirections

1. **Créez un fichier dont chaque ligne commence par un chiffre, suivi d'un slash (/), puis d'un ou plusieurs mots.**

   C'est la commande sort qui permet de trier des fichiers selon un ordre donné.

   a. **Affichez les lignes de ce fichier triées en ordre croissant, suivant le nombre placé en début de ligne**

      sort sans option trie automatiquement en fonction du premier caractère, par ordre alphabétique si c'est une lettre, par ordre croissant si ce sont des chiffres. Il suffit donc de taper : ***sort*** fichier

b. **Éliminez de chaque ligne le chiffre et le caractère «/»**.

   Pour cela, sans utiliser d'éditeur, on peut utiliser la commande cut, qui élimine des champs dans une ligne. Par défaut, le séparateur de champs est une tabulation. Ici, on peut demander à cut de considérer le slash comme un séparateur de champs (option -d, «délimiteur»), et de ne conserver que le deuxième champ du fichier (option -f, *field*). On redirige ensuite la sortie dans un autre fichier :

   cut -d/ -f2 fichier > fichier2

c. **Triez ces lignes par ordre alphabétique inverse**. C'est l'option -r (*reverse*) de sort qui inverse le résultat de la comparaison. Par défaut, sort trie par ordre alphabétique, il suffit donc de taper : ***sort*** -r fichier2

2. **Comment mettre dans un fichier la liste de tous les fichiers de l'arborescence à partir du répertoire courant ?**

   C'est l'option -R de ls qui permet d'afficher récursivement le contenu d'un répertoire et de ses sous-répertoires (à ne pas confondre avec -r qui inverse le tri). Pour rediriger la sortie d'une commande dans un fichier, on utilise le caractère >. Pour mettre la liste de tous les fichiers de l'arborescence dans un fichier toto, il faut donc taper : ***ls -R > toto***

3. **Créez un fichier liste contenant la liste de tous vos fichiers, avec leur taille, leurs droits, etc**

   Il va s'agir de faire une liste récursive de l'ensemble de votre compte (option -R de ls), incluant les fichiers de configuration (option -a) et les renseignements donnés par l'option -l.

   De la longue liste obtenue, on ne veut retenir que les fichiers, dont les droits commencent par un tiret (un d identifie les répertoires). On va donc demander à grep de chercher toutes les lignes commençant par un tiret.

   Enfin, on va placer le résultat dans un fichier appelé liste. On tape donc :

   ls -alR | grep '^-' > liste

4. **Comment afficher uniquement les fichiers du répertoire courant qui sont des liens symboliques ?**

   L'option -l de ls donne une liste longue des fichiers, incluant les droits et le type des fichiers, identifiés par une lettre : rien pour un fichier, «d» pour un répertoire (*directory*), «l» pour un lien (*link*); par exemple :

   drwxr-xr-x 2 emilia users 1024 mai 10 02:27 www/

   Pour obtenir la liste des fichiers du répertoire courant qui sont des liens symboliques, il suffit donc d'afficher cette liste longue et de ne retenir que les lignes qui commencent par un «l», grâce à la commande grep. On utilise une expression régulière pour cela : le chapeau (^) désigne le début de la ligne, et l'ensemble de l'expression est placée entre apostrophes pour qu'elle ne soit pas interprétée par le shell. On tape donc :

   ls -l | grep '^l'

5. **Combien de lignes contiennent le mot «*file*» dans la page de man de less ?**

   Il faut lancer le man de less, et ensuite chercher dedans le nombre de lignes contenant le mot file («fichier» en anglais), avec grep. C'est l'option -c (*count*) de grep qui va servir.

   Finalement, on écrit : man less | grep -c file

   Et on obtient 205. C'est le nombre de lignes contenant au moins une fois la chaîne de caractères *file*, mais pas le nombre d'occurences du mot dans le fichier : il pourrait y avoir plusieurs fois *file* sur la même ligne. On atteint là encore la limite de ce que sait faire grep.

6. **Quels sont les dix plus gros fichiers de /usr/bin/ ?**

Il faut d'abord faire une liste des fichiers avec leurs tailles (option -l de ls. Ensuite, il faut trier ces lignes en fonction de la taille, avec la commande sort. Une ligne est de la forme suivante :

-rwxr-xr-x 1 root staff 5872 Jan 21 1994 /usr/bin/rm*

C'est le cinquième champ, celui de la taille, qui nous intéresse. La numérotation commençant à 0 (comme les étages), pour indiquer le cinquième champ on écrit +4. Ensuite, on veut que le tri se fasse selon un critère numérique (option -n) et à l'envers, en commençant par les plus grandes valeurs (option -r, *reverse*).

Pour finir, on veut les dix premières lignes du résultat. On les sélectionne avec la commande head; par défaut, elle prend les dix premières lignes d'un fichier, il n'est donc pas nécessaire de préciser le nombre de lignes voulues. Finalement, on écrit :

ls -l /usr/bin | sort -nr +4 | head

Questions subsidiaires

7. **Combien de fichiers de configuration avez-vous ?**

La réponse n'est pas simple, car vous avez également des répertoires de configuration, comme .netscape/ qui contient entre autres vos marque-pages (bookmarks.html), vos préférences (preferences), etc. On va compter le nombre de fichiers commençant par un point situé dans votre répertoire principal, en éliminant les copies de sauvegarde (qui finissent par un tilde). Il y a plusieurs manières de procéder.

a. **ls, grep, tr, cut**

On peut penser à une solution complexe et bien lourde utilisant toutes ces commandes, ce qui permet de les apprendre plus qu'autre chose... On affiche une liste longue avec l'ensemble des fichiers (ls -la) puis qui élimine tous les répertoires et les liens avec grep en ne gardant que les lignes qui commencent par un tiret :

```
ls -al | grep '^-'
```

Ensuite, on utilise tr pour ne conserver qu'un seul espace entre chaque champ de la liste, en utilisant l'option -s (squeeze) qui remplace par une seule occurence la répétition d'un caractère spécifié; on met des apostrophes autour de l'espace pour qu'il soit compris comme tel et non comme un simple espace :

```
tr -s ' '
```

On utilise cut pour conserver le dernier champ avec les noms de fichiers (en l'occurence le neuvième champ), en indiquant que l'espace sert de délimiteur de champ :

```
cut -d' ' -f9
```

Enfin, on demande à grep de ne retenir que les fichiers commençant par un point, et ne finissant pas par un tilde, et on les comptel'option -c de grep. Cela donne pour finir :

```
ls -al | grep '^-' | tr -s ' ' | cut -d' ' -f 9 | grep -c '^\..*[^~]$'
```

Mais que de tourments pour si peu... :-)

b. **ls et grep**

Il y a plus simple... On demande à ls une liste des fichiers commençant par un point (avec l'option -d pour ne pas descendre dans les répertoires de configuration), on ne garde que les lignes ne finissant pas par un / ou un tilde, et on compte :

```
ls -ad .* | grep -vc '\(~\|/\)$'
```

c. **find**

On demande à find de chercher dans le répertoire courant (.), sans descendre dans l'arborescence (-maxdepth 1), tous les fichiers normaux (-type f : *file*), dont le nom commence par un point (-name '.*'). On les compte ensuite avec wc :

```
find . -maxdepth 1 -name '.*' -type f | wc -l
```

Cette solution est préférable car elle ne lance que deux processus et non pas au moins cinq comme la précédente.

8. **Combien de répertoires de configuration avez-vous ?**

Il y a deux solutions, l'une faisant appel à ls et grep, l'autre faisant appel à find.

a. **ls et grep**

Le principe est le même que dans l'exercice précédent. La différence est que grep va chercher les lignes qui commencent par un point **et** finissent par un slash. Cela se fait de la manière suivante :

grep '^\..*/$'. * signifie «n'importe quel caractère (point) répété zéro ou plusieurs fois (étoile)».

```
ls -ad .* | grep '^\..*/$' | wc -l
```

On part ici du principe que vous avez un alias de ls sur ls -F (qui met un slash après les noms de répertoires, une étoile après les noms d'exécutables et une arobase après les noms de liens). Si ce n'est pas le cas, ajoutez-le dans un fichier .alias ou tapez ls -lF.

b. **find**

On demande à find de chercher tous les fichiers de type «répertoire» (-type d : *directory*), et dont le nom commence par un point (-name '.*', les quotes servant à protéger du shell le contenu de l'expression), dans descendre dans l'arborescence.

On tape donc : find . -maxdepth 1 type d -name '.*' | wc -l

Il y a une inexactitude dans le résultat : find compte aussi ./ dans le résultat.

9. **Pour chaque ligne du fichier /etc/passwd, affichez :**

- Le cinquième caractère;
- Les caractères 5 à 10, et le treizième;
- Tous les caractères à partir du quinzième.

Une manière de résoudre cet exercice consiste à utiliser le programme sed qui sert à manipuler du texte. Nous allons utiliser la commande s de sed. Sa syntaxe est : s/*regexp*/*remplacement*/

où *regexp* est une expression régulière et *remplacement* une chaîne devant la remplacer.

- Pour afficher le cinquième caractère de chaque ligne, on peut faire ceci :

```
nacre ~ $ cat /etc/hosts | sed -e "s/^.\{4\}\(.\).*/\1/"
```

Le "^.\{4\}" du début sert à filtrer les 4 premiers caractères de chaque ligne. Ensuite, \(.\) filtre n'importe quel caractère, le fait de mettre des parenthèses enregistre le caractère en question dans une variable que l'on peut utiliser dans la chaîne de remplacement avec "\1". Ensuite, .* filtre le reste de la ligne.

- Sur le même principe, pour afficher les caractères 5 à 10 et le treizième :

```
nacre ~ $ cat /etc/hosts | sed -e "s/^.\{4\}\(.\{6\}\)..\(.\).*/\1 \2/"
```

- Et enfin, pour afficher tous les caractères à partir du quinzième :

```
nacre ~ $ cat /etc/hosts | sed -e "s/^.\{14\}//"
```

Exercices sur les pages jaunes

1. **Combien de personnes de la promotion ont un login commençant par «l» ?**

   Pour avoir une liste de tous les comptes, on tape

   ypcat passwd

   On veut ensuite sélectionner les gens dont le répertoire personnel dont le login commence par l, et compter le nombre de lignes (option -c de grep). On tape donc :

   ypcat passwd |grep c '^l.*' et on obtient 19.

2. **Classer les comptes de ces utilisateurs par numéro d'UID**

   Le numéro d'ID (*user identification*) identifie les utilisateurs, les numéros les plus bas correspondant au système (l'UID de root est 0). C'est le troisième champ, les champs étant séparés par des deux-points.

   On va utiliser sort pour trier, selon un critère numérique (option -n). Par défaut, avec sort les champs sont délimités par des blancs, il faut donc indiquer qu'ici ce sont des deux-points, avec l'option -t. Enfin, on trie en fonction du troisième champ (2, la numérotation commençant à 0).

   On suppose aussi que l'on veut pouvoir lire le résultat du tri avec less. On tape donc :

   ypcat passwd | grep '/users/9[0-9]' | sort -n -t: +2 | less

Questions subsidiaires

3. **Affichez le nom et le prénom des personnes ayant un compte sur nacre, et dont le nom de famille contient au moins 2 voyelles côte à côte. Compter ces personnes.**

   nacre ~ $ ypcat passwd | sed -e "s/:\/.*// s/.*://" | grep " .*[aeiou][aeiou].*"

4. **Affichez le login des personnes ayant sh comme shell de login.** On utilise grep pour ne garder que les lignes des pages jaunes correspondant aux personnes ayant sh comme shell de login et on utilise ***sed*** pour n'afficher que le login :

   nacre ~ $ ypcat passwd | grep "sh$" | sed -e "s/:.*//"

   Comme vous pourrez le constater, sur nacre, tout le monde à /bin/sh comme shell de login. Le shell (en général zsh) n'est lancé qu'à partir du script .profile. Il s'agit d'un choix de l'administrateur système.