

Initiation USI, TP6

Important :

- Pendant la séance prenez des notes que vous enverrez par courrier électronique à votre enseignant.
- Vous devez constituer un fichier de notes dans lequel vous listerez toutes les commandes effectuées ainsi que les éventuels résultats, et tout commentaires que vous jugerez nécessaire pour la bonne compréhension de ce document (que ce soit pour votre enseignant ou vos futures relectures et révisions).
- Vous travaillerez dans une copie du répertoire `/net/Bibliothèque/ASR1/Semaine6/` à l'intérieure du répertoire dédié au module.
- Au besoin, vous terminerez la feuille pendant la semaine.

1 Codage

1. La commande `file` permet d'obtenir des informations sur les fichiers. Parmi elles figure le codage utilisé pour les fichiers indiqués en paramètres. À l'aide du manuel de cette commande, isolez l'option permettant d'afficher cette information. Lancez la commande sur les fichiers du répertoire `codage`. Quel codage est utilisé pour chaque fichier ?
2. À partir du fichier ASCII de ce répertoire (que vous pourrez modifier au besoin)¹ et de la commande `od` retrouvez les valeurs décimales ASCII des caractères : `0 1 9 a b A B . * entrée`. Détaillez les commandes et opérations effectuées.
3. Même question en ISO-Latin-1 et en UTF-8
4. Comparez les résultats en ASCII et en UTF-8. Que remarquez vous ? Pourquoi ?
5. Pour les codages ISO-Latin-1 et UTF-8, ajoutez des caractères accentués et comparez les résultats.
6. Faites une copie de `fichier1.txt` et nommez-la `fichier4.txt`. à l'aide de la commande `echo` et d'une redirection, ajoutez une ligne de texte supplémentaire contenant un caractère accentué. Quel est l'effet sur le contenu et le codage du fichier ?
7. Convertissez `fichier3.txt` en ISO-8859-1 à l'aide de la commande `iconv` (nommez le fichier sortie `fichier5.txt`). Que se passe t-il ? Comprenez et expliquez.
8. Ouvrez le `fichier3.txt` avec `gedit`. Qu'observez-vous ?
9. Si vous ne l'avez pas déjà fait durant la séance de TD, récupérez la page d'accueil du LaBRI (<http://www.labri.fr>) avec la commande `wget` et enregistrez là sous `labri.txt`.
10. Relevez le codage de `labri.txt`
11. Editez là sous `emacs`, et notez la séquence présente en fin de ligne.
12. A l'aide de la commande `od` (que vous redigerez vers la commande `head` pour afficher seulement le codage des 10 premières lignes), repérez le codage de la fin de ligne.
13. Enlevez les séquences `^M` du fichier à l'aide de la commande `emacs query-replace`². Sauvegardez.
14. Relevez le codage de `labri.txt`
15. Regardez le résultat dans `od`.
16. Cherchez sur Internet la signification de CR-LF et concluez.

1. Vérifiez bien cependant qu'aucun changement de codage n'est effectué

2. ! pour appliquer à toutes les occurrences

2 Filtres et Redirection

Objectifs : Le but de cette partie est de vous familiariser par l'expérience avec les méthodes de redirection et les commandes (filtres) UNIX permettant de réaliser des traitements sur des gros fichiers texte. Seules quelques options offertes par ces commandes seront utilisées : l'aide en ligne (commande `man`) et la pratique vous permettront de les utiliser efficacement.

Pour chaque commande vous devrez noter dans votre compte rendu les effets, des exemples d'utilisation et tous les commentaires et indications que vous jugerez utiles ou pertinents.

Quelques commandes

Voici un certain nombre de filtres classiques qui agissent sur des textes :

- `more`, `less` : listent le contenu d'un fichier page par page
- `cat` : liste les contenus des fichiers mentionnés sur la sortie standard
- `find` : recherche un fichier à partir dans un arborescence de répertoires
- `wc` : compte le nombre de lignes, de mots et d'octets d'un fichier
- `tr` : remplace ou supprime des caractères
- `sort` : trie des lignes d'un fichier
- `head/tail` : affiche les premières/dernières lignes d'un fichier
- `grep` : affiche les lignes du fichier contenant la chaîne de caractères spécifiée
- `cut` : affiche ou supprime des zones spécifiques d'un fichier (manipulation de colonnes)

Ces commandes disposent bien sûr d'un grand nombre d'options et il vous appartient donc de vous familiariser à leur emploi. Pour cela, n'hésitez pas à consulter leur manuel d'utilisation (`man` commande).

2.1 Redirections simples

Essayez, observez et commentez :

- `find ~` (effet ?)
- `find ~ > liste-fichiers` (où sont les résultats ?)
- `cat -n liste-fichiers` (rôle de l'option ?)
- `wc liste-fichiers`
- `find ~ | more`
- `find ~ | wc` (option pour voir seulement le nombre de lignes ?)
- `echo abcd | wc -c` (comptez)
- `echo "Bienvenue à l'IUT" > message`
- `date >> message` (différence entre `>` et `>>` ?)
- `tr "[a-z]" "[A-Z]" < message`

Décortiquez le fonctionnement de la commande

```
tr ' ' '\n' < memoires.txt | sort -u > mots
```

3 grep (début)

Essayez, observez et commentez les commandes :

- `grep roi memoires.txt`
- `grep -n roi memoires.txt` (rôle de l'option ?)
- `grep ^roi memoires.txt`
- `grep duchesse$ memoires.txt` (signification de `^` et `$` ?)

Faites afficher les lignes contenant un mot (par exemple `duchesse`).

En une seule ligne, faire afficher les lignes contenant à la fois "roi" et "duc".

4 head et tail

Essayez, observez et commentez :

```
head -n 20 liste-fichiers
head -c 20 liste-fichiers
tail -n 20 liste-fichiers
tail -c 20 liste-fichiers
find ~ | head -15
```

Donnez une combinaison de `head` et `tail` qui produise en sortie, les lignes 100 à 120 des données fournies sur son entrée standard.

Conseil : Vérifiez avec `cat -n memoires.txt` qui fournit des lignes numérotées.

5 Grep (suite)

Essayez, observez, comprenez, commentez :

```
grep licence memoires.txt
grep licen.e memoires.txt
grep -i licen.e memoires.txt

grep 'roi.*ministre' memoires.txt
grep 'ministre.*roi' memoires.txt

grep -v e memoires.txt
grep -v " " memoires.txt | grep -v ^$

grep '^[éèê]' memoires.txt
grep '[x-z]$$' memoires.txt
```

Effectuez les recherches suivantes dans le fichier `mots` que vous avez obtenu plus tôt.

- Mots contenant `assi`
- mots se terminant par `oi`
- mots commençant par une voyelle et se terminant par `z`
- mots de 5 lettres commençant par `m` et se terminant par `e`.
- nombre de mots commençant par une lettre de l'intervalle `[a-m]`

6 Les filtres cut et tr

Dans le fichier `liste-comptes`, vous devez voir un ensemble de lignes composées de champs. Le format de chaque ligne est le suivant :

```
login:Prénom NOM (groupe):rép.:shell
```

Essayez, comprenez et commentez :

```
tr ":" " " <liste-comptes
cut -c1-8 liste-comptes
cut -c1-2,5-20 liste-comptes
cut -f 1 liste-comptes -d ":"
cut -f 1,3-4 liste-comptes -d ":" | tr ":" " "
```

À partir du fichier `liste-comptes`, affichez une liste des utilisateurs avec les trois champs suivants (séparés par des espaces)

- “login” utilisateur,

- Prénom NOM (groupe),
 - répertoire d'accueil de l'utilisateur.
- Stockez cette liste dans un fichier `ma-liste` à l'aide d'une redirection.

À partir de ce fichier `ma-liste`, affichez les utilisateurs de votre promotion et leur nombre.

À l'aide du filtre `sort`, affichez une liste triée par ordre alphabétique de tous les "NOMs" d'utilisateurs de votre promotion.

7 La commande find

La commande `find` permet d'afficher la liste des fichiers ou sous-répertoires d'un répertoire donnés satisfaisant certains critères. Exemple :

```
find /net/exemples -name '*.cc'
find /net/exemples -ctime -3
```

Avec la commande `find`, écrire une ligne de commande permettant d'afficher le nombre de fichiers portant le suffixe `.log` dans le répertoire `/var/log`

La commande `xargs` applique une autre commande à chacun des noms qui lui sont transmis sur l'entrée standard.

Étudiez ces exemples

```
find ~ -name SECRET | xargs chmod og-w
find /net/exemples -ctime -3 -type f | xargs ls -l
```

Faites afficher les premières lignes de tous les fichiers sources C++ que vous avez modifiés dans la semaine écoulée.

8 Le filtre sed

Le filtre `sed` (*stream editor*) applique des transformations (substitutions, destructions, etc.) à un texte qu'il reçoit en entrée.

Essayez et commentez :

```
sed -e 's/hello/bonjour/' < hello.cc
sed -e 's/^ *//' < hello.cc
```

Les transformations peuvent être appliquées à des parties du texte

```
sed -e '2,$s/hello/bonjour/' <hello.cc
sed -e '4,$d' < hello.cc
sed -e '1,/include/d' < hello.cc
```

Avec cette commande, comment créer un fichier `g` constitué des `n-1` premières lignes d'un fichier `f` de `n` lignes ?