LISV Laboratoire d'ingénierie des systèmes de Versailles
UVSQ université PARIS-SACLAY

Linux : TP 2

Halim Djerroud

révision 1.0

1 Préambule :

1.1 Téléchargement du fichier de données

Dans ce TP, nous allons manipuler un fichier texte contenant un petit **dictionnaire bilingue** (Français–Anglais), enrichi d'une catégorie pour chaque mot.

1. Téléchargez le fichier depuis le serveur avec la commande suivante :

   ```
   wget https://perso.halim.info/iut_25_26/RT/R108_GnuLinux/dictionnaire.txt
   ```

   Si le lien ne fonctionne pas vous pouvez créer vous-même le fichier `dictionnaire.txt` à l'aide de l'éditeur `nano` avec le contenu suivant :

   ```
   Pomme;Apple;Fruit
   Chien;Dog;Animal
   Chat;Cat;Animal
   Livre;Book;Objet
   Maison;House;Lieu
   École;School;Lieu
   Voiture;Car;Objet
   Arbre;Tree;Nature
   Fleur;Flower;Nature
   Rouge;Red;Couleur
   Bleu;Blue;Couleur
   Vert;Green;Couleur
   Soleil;Sun;Astro
   Lune;Moon;Astro
   Étoile;Star;Astro
   Mer;Sea;Nature
   Montagne;Mountain;Nature
   Ville;City;Lieu
   Ami;Friend;Relation
   Amour;Love;Concept
   ```

2. Vérifiez que le fichier est présent dans votre répertoire courant :

   ```
   ls -l dictionnaire.txt
   ```

3. Affichez les premières lignes pour vous familiariser avec son contenu :

   ```
   head dictionnaire.txt
   ```

1.2 Structure du fichier

Le fichier est organisé en trois colonnes séparées par un point-virgule (`;`).

`MotFrançais ; MotAnglais ; Catégorie`

Exemple :

```
Pomme;Apple;Fruit
Chien;Dog;Animal
Maison;House;Lieu
...
```

Exercice 1 : Compter avec `wc`

Utilisez la commande `wc` (*word count*) sur le fichier `dictionnaire.txt`.

1. Utilisez la commande `wc` sans option pour afficher le nombre de lignes, de mots et de caractères.
2. Consultez le manuel de la commande `wc`. Quelles options permettent d'obtenir uniquement :
   — le nombre de lignes,
   — le nombre de mots,
   — le nombre d'octets (caractères).
3. Comparez les résultats obtenus avec les options `-l`, `-w` et `-c`. Que remarquez-vous ?

Exercice 2 : Extraire des colonnes avec `cut`

La commande `cut` permet d'extraire des parties de chaque ligne d'un fichier (selon un séparateur et un numéro de colonne). À l'aide de cette commande :

1. Affichez uniquement la colonne des mots français :
2. Affichez uniquement la colonne des mots anglais.
3. Affichez uniquement la colonne des catégories.

Exercice 3 : Trier et repérer les doublons

Nous allons maintenant utiliser les commandes `sort` et `uniq` pour organiser et analyser le fichier `dictionnaire.txt`.

1. Triez le fichier complet par ordre alphabétique des lignes :

   ```
   sort dictionnaire.txt
   ```

2. Affichez la colonne des catégories à l'aide de la commande `cut` puis enregistrer le résultat dans un fichier temporaire `ex. tmp.txt`, puis triez les résultats du nouveau fichier. Vous pouvez utiliser `copier/coller` et `nano` pour copier le résultat dans un nouveau fichier.
3. Avec le même procédé, supprimez les doublons pour obtenir la liste unique des catégories.
4. Comptez le nombre de fois que chaque catégorie apparaît.

Exercice 4 : Rechercher avec `grep`

La commande `grep` permet de rechercher des motifs (chaînes de caractères) dans un fichier.

1. Affichez toutes les lignes du fichier contenant le mot `Animal`.
2. Recherchez toutes les lignes contenant le mot `Nature`.
3. Recherchez les lignes contenant le mot `Amour`, sans tenir compte de la casse.
4. Comptez le nombre de lignes contenant le mot `Astro`.
5. Recherchez toutes les lignes qui **ne contiennent pas** le mot `Couleur`.

Exercice 5 : Comparer des fichiers avec `diff` et `cmp`

La comparaison de fichiers est utile pour détecter des différences entre deux versions d'un même document.

1. Copiez le fichier `dictionnaire.txt` dans un nouveau fichier `dictionnaire_v2.txt`.
2. Éditez `dictionnaire_v2.txt` et modifiez quelques lignes (par exemple, changez une traduction ou ajoutez une nouvelle entrée).
3. Comparez les deux fichiers ligne par ligne avec la commande :

   ```
   diff dictionnaire.txt dictionnaire_v2.txt
   ```

   Que remarquez-vous ?
4. Utilisez la commande `cmp` pour comparer les fichiers au niveau binaire. Quelle différence avec `diff` ?

Exercice 6 : Explorer les informations système

Dans cet exercice, nous allons découvrir quelques commandes permettant d'obtenir des informations sur le système Linux.

1. Affichez la version du noyau et le nom de la machine :

```
uname -a
```

2. Vérifiez l'espace disque disponible sur les partitions montées :

```
df -h
```

3. Affichez la taille occupée par votre répertoire personnel :

```
du -sh ~
```

4. Consultez l'état de la mémoire (RAM) de votre machine :

```
free -h
```

5. Depuis combien de temps votre machine fonctionne-t-elle ?

Exercice 7 : Archivage et compression

Dans cet exercice, nous allons apprendre à regrouper des fichiers dans une archive et à les compresser.

1. Créez un dossier de test et ajoutez-y quelques fichiers :

```
mkdir archive_test
cd archive_test
nano fichier1.txt  #  contenu : Bonjour"
nano fichier2.txt  #  contenu : Linux TP
nano fichier3.txt  #  contenu : Archive
cd ..
```

2. Créez une archive nommée `test.tar` contenant le dossier `archive_test` :

```
tar -cvf test.tar archive_test
```

3. Vérifiez le contenu de l'archive sans l'extraire :

```
tar -tf test.tar
```

4. Compressez l'archive avec `gzip` puis décompressez-la :

```
gzip test.tar
ls -l test.tar.gz
gunzip test.tar.gz
```

5. Créez une archive compressée directement en une seule étape :

```
tar -czvf test.tar.gz archive_test
```

6. Créez également un fichier ZIP et testez l'extraction :

```
zip -r test.zip archive_test
unzip test.zip
```

Exercice 8 : Commandes pratiques et personnalisation

Dans cet exercice, nous allons découvrir quelques commandes utiles pour travailler plus efficacement dans le terminal.

1. Affichez l'historique de vos commandes :

```
history
```

   Quelle est la différence entre les numéros affichés et les lignes de commandes ?

2. Ré-exécutez une ancienne commande de l'historique en utilisant son numéro :

1.2 Structure du fichier

```
!25
```

(Remplacez `25` par le numéro d'une commande existante dans votre historique.)

3. Créez un alias pour la commande `ls -l` nommé `ll` :

```
alias ll="ls -l"
```

Testez-le ensuite avec :

```
ll
```

4. Nettoyez l'écran du terminal :

```
clear
```

Mini-projet : Analyse avancée d'une bibliothèque de livres

Dans ce mini-projet, vous allez organiser vos fichiers dans une petite arborescence, créer un fichier CSV, puis écrire un script Bash qui réalise automatiquement plusieurs analyses avec les commandes étudiées.

1. Créez l'arborescence suivante dans votre répertoire personnel :

```
projet_biblio/
 data/
 scripts/
 results/
```

2. Créez un fichier nommé `livres.csv` dans le dossier `data` :

```
nano data/livres.csv
```

3. Recopiez le contenu suivant dans le fichier, puis enregistrez et quittez :

```
Titre;Auteur;Année;Genre
1984;George Orwell;1949;Roman
Le Petit Prince;Antoine de Saint-Exupéry;1943;Conte
L'Étranger;Albert Camus;1942;Roman
Les Misérables;Victor Hugo;1862;Roman
La Peste;Albert Camus;1947;Roman
Fahrenheit 451;Ray Bradbury;1953;Science-Fiction
Dune;Frank Herbert;1965;Science-Fiction
Fondation;Isaac Asimov;1951;Science-Fiction
Harry Potter;J.K. Rowling;1997;Fantasy
Le Seigneur des Anneaux;J.R.R. Tolkien;1954;Fantasy
```

4. Dans le dossier `scripts`, créez un script nommé `analyse_livres.sh` avec `nano`. Le script doit afficher :
   - Le nombre total de livres.
   - La liste des genres présents (`cut`, `sort`, `uniq`).
   - Le nombre de livres de type `Roman`.
   - Le nombre de livres de type `Science-Fiction`.
   - Tous les livres écrits par `Camus`.
   - Les titres qui commencent par la lettre `L` (regex avec `grep "^L"`).
   - Les titres qui se terminent par la lettre `e` (`grep "e$"`).
   - Les livres publiés entre 1900 et 1999 (`grep "19[0-9][0-9]"`).
   - Un tri par année (colonne 3) en ordre croissant.
   - Un tri par genre (colonne 4), puis par titre (colonne 1).

5. Rendez le script exécutable et exécutez-le :

```
chmod +x scripts/analyse_livres.sh
./scripts/analyse_livres.sh
```