TD2 – Langages de script

Abdallah Ammar

13 janvier 2026

Récupération des fichiers de données

Dans ce TD, vous allez travailler sur des fichiers texte fournis par l'enseignant. Ces fichiers contiennent des données fictives destinées à l'apprentissage des commandes de filtrage et d'analyse de texte.

Commencez par créer un répertoire `data`, puis téléchargez les fichiers à l'aide des commandes suivantes :

```
$ mkdir data
$ cd data
$ wget https://raw.githubusercontent.com/AbdAmmar/LDS/main/TD/TD2/data/users.txt
$ wget https://raw.githubusercontent.com/AbdAmmar/LDS/main/TD/TD2/data/logs.txt
$ wget https://raw.githubusercontent.com/AbdAmmar/LDS/main/TD/TD2/data/notes.txt
```

Vérifiez que les fichiers ont bien été téléchargés. Si la commande `wget` n'est pas disponible, installez-la avec :

```
$ sudo apt update
$ sudo apt install wget
```

1 Lire et compter

Travail à faire

1. Affichez le contenu du fichier `users.txt`.
2. Comptez le nombre de lignes du fichier.
3. Affichez le contenu du fichier `logs.txt`.
4. Comptez le nombre de lignes du fichier `notes.txt`.

Commandes utiles

```
$ cat users.txt
$ wc -l users.txt
```

2 Introduction aux pipes

Un *pipe* permet d'envoyer la sortie d'une commande en entrée d'une autre.

Travail à faire

1. Comptez le nombre de lignes de `users.txt` sans afficher le contenu du fichier.
2. Expliquez le rôle du symbole `|`.

Commande clé

```
$ cat users.txt | wc -l
```

3 Rechercher des informations avec `grep`

Travail à faire

À partir du fichier `users.txt` :

1. Affichez les lignes contenant `bash`.
2. Comptez le nombre d'utilisateurs utilisant `bash`.
3. Affichez les lignes ne contenant pas `bash`.

Commandes utiles

```
$ grep bash users.txt
$ grep -c bash users.txt
$ grep -v bash users.txt
```

4 Extraire des informations

Le fichier `users.txt` est structuré : les champs sont séparés par le caractère `:`.

Travail à faire

1. Affichez uniquement la colonne des noms d'utilisateurs.
2. Affichez uniquement la colonne des shells.

Commandes utiles

```
$ cut -d: -f1 users.txt
$ cut -d: -f2 users.txt
```

5 Trier et compter

Travail à faire

1. Affichez la liste des shells utilisés.
2. Triez cette liste.
3. Comptez combien de fois chaque shell apparaît.

Pipeline attendu

```
$ cut -d: -f2 users.txt | sort | uniq -c
```

6 Analyse de fichiers de logs

Travail à faire

À partir du fichier `logs.txt` :

1. Affichez uniquement les lignes contenant `ERROR`.
2. Comptez le nombre d'erreurs.
3. Affichez les noms des utilisateurs ayant généré une erreur.

Commandes utiles

```
$ grep ERROR logs.txt
$ grep -c ERROR logs.txt
$ grep ERROR logs.txt | cut -d= -f2
```

7 Mini-analyse

Travail à faire

Sans utiliser d'éditeur de texte :
— Quel est le shell le plus utilisé ?
— Quel utilisateur apparaît le plus souvent dans les logs ?
Justifiez vos réponses à l'aide de commandes Linux.

Pour aller plus loin – Exercice ludique (optionnel)

Installez les programmes `cowsay` et `fortune`. Testez ensuite l'utilisation d'un pipe entre les deux commandes.

```
$ sudo apt install cowsay fortune
$ fortune
$ fortune | cowsay
```

8 Les flux standards

Ce TD a pour objectif d'introduire la gestion des sorties et des erreurs sous Linux. Sous Linux, un programme communique avec l'extérieur à l'aide de flux :
— l'entrée standard (stdin) ;
— la sortie standard (stdout) ;
— la sortie d'erreur (stderr).
Par défaut, la sortie standard et la sortie d'erreur sont affichées à l'écran.

9 Rediriger la sortie standard

Travail à faire

1. Listez le contenu de votre répertoire personnel.
2. Redirigez cette sortie dans un fichier `liste.txt`.
3. Vérifiez le contenu du fichier.

Commandes

```
$ ls
$ ls > liste.txt
$ cat liste.txt
```

10 Générer et observer une erreur

Travail à faire

1. Essayez d'afficher un fichier qui n'existe pas.
2. Observez le message affiché.

```
$ cat fichier_inexistant.txt
```

Expliquez pourquoi le message n'est pas redirigé avec `>`.

11 Rediriger la sortie d'erreur

La sortie d'erreur peut être redirigée séparément.

Travail à faire

1. Redirigez l'erreur précédente dans un fichier `erreur.txt`.
2. Vérifiez le contenu du fichier.

```
$ cat fichier_inexistant.txt 2> erreur.txt
$ cat erreur.txt
```

12 Rediriger sortie et erreur

Il est possible de rediriger à la fois la sortie standard et la sortie d'erreur.

Travail à faire

1. Lancez une commande produisant à la fois une sortie et une erreur.
2. Redirigez les deux flux dans un même fichier.

```
$ ls fichier_inexistant > sortie.txt 2> erreur.txt
```

Ou en une seule commande :

```
$ ls fichier_inexistant > tout.txt 2>&1
```