# TP 5 - `grep`, `sed` et expressions régulières

**Consignes de rendu**

1. Créez le dossier `~/Linux/TP05` et placez-vous dedans.
2. Créez-y un fichier `reponses.txt` pour noter vos observations.
3. Le TP est à rendre par e-mail en suivant le format habituel (voir slides du premier cours). Pour obtenir l'archive à envoyer depuis le répertoire `~/Linux/`, utilisez :

```
tar jcvf numéro_d'étudiant-prénom français-TP05.tar.bz2 TP05
```

**Pour tester vos expressions :** commencez dans le terminal avec `grep -E`. Si vous avez accès à Internet, RegExr peut vous aider à comprendre les correspondances, mais **il n'est pas nécessaire** pour ce TP. Attention : les moteurs d'expressions régulières n'ont pas toujours la même syntaxe.

## 1 Rechercher du texte avec `grep`

**Exercice 1 (Premières recherches)**

Dans `~/Linux/TP05`, créez un fichier `animaux.txt` contenant exactement ces lignes (une par ligne) : `chat`, `Chien`, `chat noir`, `CHAT`, `oiseau`, `chat et chien`.

(a) Affichez les lignes contenant `chat`, en respectant la casse.

(b) Affichez les lignes contenant `chat`, sans distinguer majuscules et minuscules.

(c) Comptez les lignes contenant `chat`, sans distinguer la casse.

(d) Affichez les lignes contenant `chat` avec leur numéro de ligne.

## 2 Expressions régulières

**Exercice 2 (Construire un motif)**

Créez `tests.txt` avec les lignes suivantes : `7`, `42`, `007`, `abc`, `A12`, `123abc`, `2026-10-28`, `Bonjour`.

(a) Affichez les lignes qui contiennent au moins un chiffre.

(b) Affichez les lignes qui commencent par une majuscule ASCII (A à Z).

(c) Affichez les lignes constituées uniquement d'un ou plusieurs chiffres.

(d) Un étudiant utilise `grep -E '[0-9]+' tests.txt` pour la question précédente. Pourquoi est-ce incorrect ? Corrigez sa commande.

(e) Affichez les lignes contenant exactement trois chiffres consécutifs, même si d'autres caractères apparaissent sur la ligne.

# 3 Chercher dans un vrai texte

**Exercice 3 (Notre-Dame de Paris)**

Téléchargez le texte de *Notre-Dame de Paris* avec la commande suivante :

```
wget https://monge.univ-eiffel.fr/~alabarre/teaching/shnu/linux/vh-nddp-clean.txt
```

Si le téléchargement échoue, demandez le fichier à l'enseignant. Dans les commandes ci-dessous, utilisez le nom `vh-nddp-clean.txt`.

(a) Trouvez les lignes contenant `Quasimodo`, sans distinguer la casse. Affichez les numéros de ligne.

(b) Affichez les lignes contenant deux occurrences de `Quasimodo` (sur une même ligne).

(c) L'option `-C 2` de `grep` affiche aussi deux lignes avant et après chaque correspondance. Utilisez-la pour retrouver le contexte de `Quasimodo`.

(d) Expliquez en une phrase l'intérêt de `-C` lorsque vous recherchez un mot dans un journal système ou un long texte.

# 4 Transformer du texte avec `sed`

**Exercice 4 (Remplacements et sauvegarde)**

Reprenez le fichier `animaux.txt` du premier exercice.

(a) Sans modifier le fichier, remplacez la première occurrence de `chat` par `chien` sur chaque ligne.

(b) Effectuez ce remplacement pour toutes les occurrences de chaque ligne.

(c) Enregistrez le résultat dans `animaux-modifies.txt`, sans modifier `animaux.txt`.

(d) Modifiez maintenant `animaux.txt` directement, mais conservez une copie de sauvegarde `animaux.txt.bak`.

(e) Vérifiez le contenu des deux fichiers. Lequel contient la version d'origine ?

**Astuce**

`sed` permet d'effectuer plusieurs substitutions en une seule commande, en les séparant par un point-virgule. Par exemple :

```
sed 's/chat/chien/; s/oiseau/poisson/' fichier.txt
```

Les substitutions sont effectuées dans l'ordre, sur chaque ligne.

**Exercice 5 (Changer le format d'une date)**

Dans les questions suivantes, on suppose que la ligne ne contient *qu'une date*, correctement écrite. Les caractères chinois sont des caractères littéraux de la commande.

(a) Avec `echo` et `sed`, transformez `2026-10-01` en 2026年10月01 (sans le dernier caractère chinois).

(b) Transformez 2026年10月01日 en `2026-10-01`.

(c) Avec la commande `y` de `sed`, transformez les chiffres `0123456789` en leurs équivalents chinois. Rappel : les deux chaînes de `y` doivent avoir le même nombre de caractères.

# 5 Groupes de capture

**Astuce**

Les parenthèses permettent de regrouper des parties d'une expression régulière et de mémoriser le texte correspondant à chacune d'elles.
Par exemple, dans l'expression régulière `([0-9]+)-([0-9]+)`, le premier groupe capture les chiffres avant le tiret, et le second ceux qui se trouvent après.
Avec `sed -E`, on peut réutiliser les groupes capturés dans une substitution : `\1` désigne le premier groupe, `\2` le deuxième, etc.

Exemple :

```
echo '12-34' | sed -E 's/([0-9]+)-([0-9]+)/\2-\1/'
```

Résultat : `34-12`.

**Exercice 6 (Réorganiser des informations)**

(a) Dans l'expression régulière `([0-9]{4})-([0-9]{2})-([0-9]{2})`, combien y a-t-il de groupes ? Que contiennent-ils lorsqu'on l'applique à la date `2026-10-28` ?

(b) À l'aide de `sed -E` et des groupes de capture, transformez la date `2026-10-28` en `28/10/2026`.

(c) Un fichier `personnes.txt` contient des lignes de la forme :

Dupont :Alice
Martin :Bob
Durand :Claire

Écrivez une commande qui affiche les prénoms avant les noms, séparés par un espace.

# Défis facultatifs

**Exercice 7 (Pour aller plus loin)**

(a) Trouvez une expression régulière reconnaissant une ligne contenant exactement un prénom simple ou deux prénoms reliés par un tiret (lettres ASCII, première lettre majuscule). Testez-la sur un fichier `prenoms.txt` contenant les lignes `Alice`, `Jean-Pierre`, `alice` et `Jean-Pierre`.

(b) À l'aide de `locate -r`, cherchez les chemins se terminant par `.conf`. Pourquoi les résultats peuvent-ils différer de ceux de `find` ?

(c) En une seule commande, trouvez les lignes de `tests.txt` contenant exactement trois chiffres, remplacez les chiffres `0` par `X`, et enregistrez le résultat dans `resultat.txt`.