Initiation à l'environnement Unix : TP sur `sed`

décembre 2024 — Pierre Rousselin

Exercice 1 : La mygale et la souris

1. Télécharger « La cigale et la fourmi » de La Fontaine en entrant la commande

```
$ wget https://www.math.univ-paris13.fr/~rousselin/unix/cigale.txt
```

Vous pouvez copier coller dans votre terminal cette commande (pour coller dans le terminal utiliser ctrl+shift+V).

2. Le programme `sed` est utilisé principalement (mais pas seulement) pour faire des substitutions, ligne par ligne. Par exemple, la commande

```
sed 's/toto/titi/' mon_fichier
```

remplace la première occurrence de `toto` par `titi` dans chaque ligne de `mon_fichier`. Remarque importante : le fichier de départ n'est pas modifié, le résultat est envoyé sur la sortie standard.
En utilisant `sed`, afficher la fable en changeant `fourmi` par `souris` dans le corps de la fable (pour le titre on verra plus tard).

3. Changer le verbe `chanter` par le verbe `coder`, chaque fois qu'il apparaît (y compris dans des formes conjuguées).

4. Dans la première partie d'une commande de substitution, on peut utiliser une expression rationnelle, comme avec `grep`. En particulier, les ancres `^` et `$` représentent, respectivement, le début et la fin de la ligne et l'expression `.` représente un caractère quelconque. Par exemple, la commande `sed 's/^.//'` supprime le premier caractère de chaque ligne.

a) Insérer, en début de chaque ligne, 4 espaces.

b) Insérer, en début de chaque ligne, une tabulation (pour entrer le caractère de tabulation dans le terminal, taper ctrl+V, puis une tabulation).

5. On peut enchaîner les commandes `sed` en utilisant la syntaxe suivante :

```
sed -e commande1 -e commande2 ... -e commandeN mon_fichier
```

(si `mon_fichier` n'est pas précisé, `sed` prend son entrée sur l'entrée standard).
Changer `cigale` par `mygale` partout ou elle apparaît.

6. On peut préciser sur quelles lignes on veut travailler en faisant précéder une commande `sed` d'une ou deux adresses de lignes, de la façon suivante :

```
sed '5 s/titi/toto/' mon_fichier # seulement sur la ligne 5
sed '3,6 s/titi/toto/' mon_fichier # lignes 3 à 6
sed '/tata/ s/titi/toto/' mon_fichier # lignes contenant tata
sed '/tata/,/tutu/ s/titi/toto'
```

La dernière substitution s'opère de la première ligne contenant `tata` jusqu'à (inclus) la prochaine ligne contenant `tutu`.

Ajouter la chaîne `!!!` en fin de ligne dans les cas suivants :

a) seulement la ligne 10 ;

b) seulement les lignes 4 à 9 ;

c) seulement les lignes commençant par `J` ;

d) depuis « Elle alla » jusqu'à « principal ».

7. Par défaut, chaque ligne (après substitution éventuelle) est imprimée sur la sortie standard. L'option `-n` permet de changer ce comportement. De plus, la commande `p` (pour *print*) permet d'imprimer des lignes, par exemple

```
sed -n '10,12 p'
```

permet de n'imprimer que les lignes entre 10 et 12.
En se servant de cette fonctionnalité, afficher :

a) les 8 première lignes ;

b) les lignes contenant un point ;
c) les lignes contenant un accent circonflexe ;
d) les lignes de la première occurence de « fourmi » à la deuxième.

8. Revenons sur les commandes de sustitution elles sont de la forme

`s/motif/remplacement/[drapeaux]`

Dans le remplacement, on peut avec `&`, faire référence à la chaîne qui a correspondu au motif. Par exemple, la commande

```
sed 's/[0-9][0-9]*/(&)/' monfichier
```

permet de mettre entre parenthèse le premier nombre décimal de chaque ligne de `monfichier`. Le métacaractère `*` représente la présence 0, 1, 2, ... fois de l'expression qui le précède. On peut aussi écrire avec les expressions rationnelles étendues :

```
sed -E 's/[0-9]+/(&)/' monfichier
```

Le métacaractère `+` dans une expression étendue représente la répétition 1, 2, ..., fois de l'expression qui le précède. Le drapeau peut comporter entre autres :

— la lettre `g` : dans ce cas toutes les occurences de chaque ligne sont remplacées (à retenir) ;
— le chiffre `n` : dans ce cas seule l'occurrence `n` est remplacée ;
— la lettre `p` : pour imprimer le résultat (utile surtout avec l'option `-n`).

a) Remplacer dans chaque ligne toute les occurence de la lettre `l` (minuscule ou majuscule) par la chaîne `{\ell}` (il faut « échapper » la contre-oblique donc entrer `\\` à la place de `\`). Comparer avec le résultat sans drapeau.
b) Faire de même avec seulement la deuxième occurence d'une lettre `l`.
c) Enlever les lettre `e` (accentuées ou non) dans tout le texte.
d) Mettre les mots « fourmi » et « cigale » entre accolades.

9. On peut, dans le motif ou le remplacement, utiliser les références des expressions rationnelles. Pour former un groupe, on utilise `\(` et `\)` et pour faire référence à un groupe, on utilise `\1` pour le premier, `\2` pour le suivant, etc. Par exemple, la commande

```
sed 's/\([A-Z][A-Z]*\) \([a-z][a-z]*\)/\2 \1/'
```

cherche dans chaque ligne, un mot entièrement en majuscules suivi d'un mot entièrement en minuscules, puis les échange.

Si on préfère les expressions rationnelles étendues (ERE), la commande avec l'option `-E`,

```
sed -E 's/([A-Z]+) ([a-z]+)/\2 \1/'
```

fait la même chose. L'opérateur `+` signifie que l'expression qui précède est répétée 1, 2, 3, ... fois.

a) Échanger dans chaque ligne le premier mot avec le deuxième (disons pour simplifier que les mots sont séparés par un espace).
b) Échanger dans chaque ligne le premier mot avec le dernier.

10. Le texte présente un défaut typographique. En français, les caractères `;` `:` `?` et `!` doivent être précédés d'un espace. Écrire une commande `sed` qui repère ce défaut et le corrige.

11. Réfléchir à l'usage que vous pourriez faire de `sed` lorsque vous programmez, par exemple en C.

--- * ---

Exercice 2 : Toutes les fables

Grâce au merveilleux site `https://www.gutenberg.org/` nous allons télécharger toutes les fables de La Fontaine en un fichier texte.

```
$ wget https://www.gutenberg.org/files/56327/56327-0.txt
```

Le problème est qu'il peut avoir été écrit sous Windows. Petit détail qui peut être ennuyant :

— Sur les systèmes Unix, les fins de lignes sont signalées simplement par le caractère de fin de ligne (caractère numéro 10 de la table ASCII ou encore `\n`) ;

— Sur Windows, les fins de lignes sont signalées par les deux caractères à la suite : retour chariot (caractère numéro 13, `\r`) puis nouvelle ligne (caractère numéro 10).

Entrer

```
file 56327-0.txt
```

et dire ce qu'il en est pour notre fichier. On va donc changer les fins de lignes en supprimant tous les retours chariot avec la commande `tr` (pour *transliterate*). L'option `-d` permet de supprimer (*delete*) des caractères.

```
tr -d '\r' <56327-0.txt >fables.txt
rm 56327-0.txt
file fables.txt # pour vérifier
```

1. Afficher avec `less` le contenu du fichier `fables.txt`.
2. Donner la taille du fichier `fables.txt`, ainsi que son nombre de lignes et de mots.
3. Afficher, avec `grep`, le numéro de la ligne correspondant au début de « La cigale et la fourmi ».
4. Proposer une méthode permettant de compter le nombre de fables dans le fichier et la mettre en œuvre.
5. Générer le fichier `cigale.txt` contenant uniquement la fable « La cigale et la fourmi ».
6. Combien de fables ont le mot `loup` dans leur titre ?
7. Compter le nombre de lignes vides dans le fichier.
8. Enlever les lignes contenant la chaîne `[illustration]`. Indice : pour appliquer une commande `sed` à des lignes ne correspondant pas au motif `motif` on peut utiliser :
   `sed '/motif/ !cmd'`
9. Changer toutes les majuscules en minuscules dans le fichier (voir la page de manuel et/ou la page `info` concernant `tr`).

--- * ---