

# Systeme d'exploitation

## Editeur de flux

### (9)

- SED

- Stream EDitor

- Éditeur de flux de données : filtre l'affichage du contenu d'un fichier
  - prend ses données **ligne par ligne** depuis un fichier texte,
  - **affiche** un résultat sur la sortie standard
  
- Utilisation : automatisation de tâches d'édition répétitives
  - remplacer une chaîne dans différents fichiers/lignes (*nom de domaine*)
  - harmonisation de documents (*nettoyage*)
  - extraction de données (*de fichiers de log p.ex.*)
  
- Scripts SED : fichier regroupant des commandes sed

# SED en ligne de commande

**sed** [options] '**commande sed**' fichier

- Par défaut, chaque ligne traitée est affichée : **-n** inhibe cet affichage
- Par défaut, les commandes **affichent** le fichier traité avec **-i** elles **modifient** le fichier

fichier  
 exemple
 

un  
deux  
trois

- commande de **numérotation des lignes** : "="

- sed '=' exemple # **affiche** en insérant une ligne numéro de ligne

1  
un  
2  
deux  
3  
trois

- sed **-n** '=' exemple # n'affiche que des numéros de ligne

1  
2  
3

- sed **-i** '=' exemple # **ajoute** les lignes numéros dans le fichier exemple

# Contrôler l'exécution

- Spécifier la ou les lignes sur lesquelles sera appliquée la commande

- commande de **suppression de ligne** : **d**

- sed '**3 d**' exemple **# supprime la 3<sup>ème</sup> ligne**

un  
deux  
quatre  
cinq

un  
deux  
trois  
quatre  
cinq

- sed '**2,4 d**' exemple **# supprimer les lignes 2 à 4**

un  
cinq

- commande d'**affichage** : **p** (*utile que si l'option **-n** est précisée*)

- sed **-n '2,4 p**' exemple **# n'afficher que les lignes 2 à 5**

deux  
trois  
quatre

# Exécution

---

- **\$** : désigne la dernière ligne du fichier
  - sed '5,**\$** d' exemple # supprime à partir de la ligne 5

```
sed '3,$ d' exemple :
un
deux
```

- **!** : applique la commande aux lignes ne correspondant pas au motif
  - sed '3,5 **!** d' exemple # détruire tout sauf les lignes 3 à 5

```
sed '2,4 ! d' exemple :
deux
trois
quatre
```

# Expressions régulières

---

- Spécifier les lignes à traiter par correspondance avec une **expression régulière**
  - définie entre 2 caractères arbitraires (le "/" en gl)
    - `sed '/^#/ d ' fichier`    # supprime les lignes de commentaires
    - `supprime les lignes comprises entre debut et fin`
      - `sed '/^debut$/, /^fin$/ d ' fichier`
    - `supprime les lignes avec 2 à 4 chiffres successifs`
      - ( protection des { } par des \ )
      - `sed ' /[[[:digit:]]\{2,4\} / d ' fichier`
  - **-r** : expressions régulières **étendues** (plus de protection des { } )
    - `sed -r ' / [[[:digit:]]{2,4} / d ' fichier`

# Contrôle de la sortie

---

- **w fichier** : écrit la ligne dans le fichier

Exemple/Rappel : les fonctions sont délimitées par : `maFonction() { .... }`

- `sed -n '/() {$/ w nomsFonctions.sh' script.sh #noms de fonctions`
- `sed -n '/() {$/ ,/^}/ w fonctions.sh ' script.sh #corps de fonctions`

- **r fichier** : lit le fichier et l'affiche sur la sortie standard (après la ligne)

- insérer le contenu du fichier `entete.txt` en 2<sup>ème</sup> ligne

- `sed -i '1r entete.txt' script.sh`

(Attention : sans valeur numérique, insère le fichier avant chaque ligne)

- **i\ texte** : insère le texte `texte` avant la ligne (celle correspondant au motif)
- **a\ texte** : insère le texte `texte` après la ligne
- **c\ texte** : remplace la ligne par le texte `texte`

# Commandes de substitution

---

- **s/ancienne/nouvelle/**
  - substitue la **1<sup>ère</sup>** occurrence de ancienne par nouvelle  
sed '**s/[0-9]//**' fichier # **supprime le 1<sup>er</sup> chiffre de chaque ligne**
- **s/ancienne/nouvelle/x**
  - si **x = g** : remplace toutes les occurrences  
sed 's/[0-9]/#/g' fichier # **remplace par des # tous les chiffres**
  - **x = ###** (un nombre) : remplace la **###<sup>ème</sup>** occurrence  
sed 's/[0-9]///4' fichier # **supprime le 4<sup>ième</sup> chiffre dans chaque ligne**
  - **x = p** : affiche (uniquement si -n)
  - **x = w** fic # **écrit la ligne substituée dans le fichier fic**



# Transcription

---

- **y**/liste1/liste2/
  - remplacer chaque caractère de *liste1* par son correspondant dans *liste2* (*listes de même longueur, pas d'intervalles*)
    - sed '**y**/éèêë/eeee/' document.txt *# supprime les accents sur les e*
- **e** commande
  - évaluer une commande et insérer son résultat dans la sortie (*avant la ligne en correspondance*)
    - sed '/^date\$/ **e** date|cut -d" " -f1-3'
    - *insère une ligne avec la date quand le mot date est seul sur une ligne*

# Exécuter plusieurs commandes

---

- Grouper plusieurs commandes sed sur une même ligne: " ; " ou " -e "
  - sed '=;1r entete.txt' *fichier # numéroté et ajouter un fichier en-tête*
  - sed -e '=' -e '1r entete.txt' *fichier*
- Appliquer plusieurs commandes pour un même motif : { ... ; ... }
  - sed -n '/^e/ {=;p}' *fichier # lignes commençant par 'e' avec leur numéro*
- Regrouper les commandes sed dans un fichier : sed -f script.sed *fichier*  
 applique l'ensemble des commandes une ligne après l'autre
  - Ex : script.sed
    - s/[0-9]//g
    - s:/ /g

# Blocs dans les scripts SED

---

- Appliquer une suite de commandes à un même motif : **{...}**

**1,5{**

s:/ /g

s/ \*/ /g

**}**

*pour chaque ligne comprises entre **1 et 5***

*remplacement des : par un espace*

*réduction des suites d'espaces successifs*

**/^u/{**

i/utilisateur

**}**

*pour les lignes commençant par un **u***

*insérer avant la ligne le mot : utilisateur*

**/{\$/,/^}/{**

i/supprimé

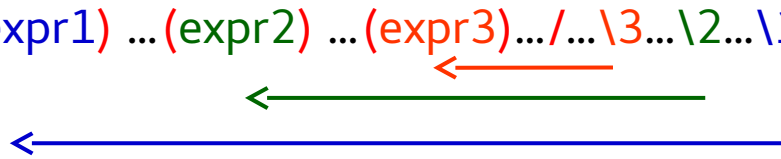
d

**}**

*pour chaque ligne du **bloc compris entre { }***

*insérer le mot supprimé à la place de la ligne*

# Références arrières

- Reprise de la correspondance avec l'**expression régulière complète** : **&**
  - reprendre pour la substitution, la chaîne mise en correspondance avec l'expression régulière (uniquement la 1<sup>ère</sup> occurrence par défaut)
  - Exemple : faire précéder de "N°" tous les nombres (42 devient N°42)
    - `sed 's/[0-9][0-9]*/N°&/g' fichier`
- Reprise de **sous-expressions** définies par des parenthèses (...):
  - pour ne pas avoir à protéger les (), utiliser l'option **-r** (régulières étendues)
  - références arrières par **\numéro**
    - `sed -r 's/... (expr1) ... (expr2) ... (expr3) ... /... \3... \2... \1/'`

- Exemple : inverse l'ordre des 3 champs et insérer un espace
  - `sed -r 's/^(.*);(.*)$ /\3 \2 \1/' fichier.csv`

# Exécution sur plusieurs fichiers

---

- Exécuter une(des) commande(s) sur plusieurs fichiers :
  - `sed -i '1r entete' *.sh` => n'insère l'entête **que dans le premier fichier**  
(la numérotation porte sur l'ensemble des fichiers)
- 3 solutions :
  - 1) option **-s** : traite indépendamment chacun des fichiers  
`sed -is '1r entete' *.sh`
  - 2) soit utiliser la commande **find** en amont  
`find / -name "*.sh" -exec sed -i '1r entete' {} \;`
  - 3) soit écrire un **script shell** appelant une commande ou un script SED  
for i  
do  
    **sed -i '1r entete' \$i**  
done