CHALENCON Yoan

19,5/20

Utilisation du shell Unix
6 - Commandes de traitement de textes

Rappel : vous devez simplement effectuer le travail demandé et noter les commandes utilisées et les réponses aux questions dans l'espace laissé libre. **Attention :** l'objectif est que vous soyez capables de refaire SEUL les manipulations décrites.

Pré-requis

Vous devez avoir parcouru le chapitre « Les processus » dans les diapos de cours :
R1.04-IntroSysteme/Unix.pdf

Commandes utiles (liste non exhaustive) : `grep`, `wc`, `sort`, `cut`, `sed`.

Créez un répertoire dédié à ce TP et placez-vous dans ce répertoire.

Copiez le fichier `/users/but/info/Public/Liste1A.txt` dans ce répertoire. Ouvrez ce fichier avec votre éditeur favori et observez son contenu. Il s'agit d'une table donnant la liste des étudiants avec le *login*, le *nom*, le *prénom*, le *courriel* et le *groupe de TP*. Chaque ligne contient les informations pour un étudiant, chaque information est séparée de la suivante par un caractère ':'.

1 Recherches simples

La commande `grep` permet de chercher une chaîne de caractères dans un ou plusieurs fichiers :
`grep` *`chaîneÀChercher fichier1 fichier2 ...`*

1. Faites afficher la liste de tous les étudiants de votre groupe de TP.

```
grep TP1B Liste1A.txt
```

2. Faites afficher la liste de tous les « Clement », puis la liste de tous les « Nathan ».

```
grep Clement Liste1A.txt
grep Nathan Liste1A.txt
```

3. Faites afficher la liste de tous les « Leo ». Comment faire pour éviter d'afficher aussi « Leo-Paul » et « Leonard » ? **NB :** on peut le faire en utilisant une option de `grep` (faire `man grep`) ou bien en spécifiant correctement la chaîne à chercher. Donnez les deux commandes.

```
grep -w Leo Liste1A.txt
ou
grep Leo: Liste1A.txt
```

4. Faites afficher la liste de tous les étudiants du groupe 2 (`TP2C` et `TP2D`).

```
grep TP_ Liste1A.txt
```

2 Projections simples

La commande `grep` permet de *sélectionner* un ensemble de lignes parmi les lignes du fichier.

Placez-vous dans le répertoire `TPMots` du répertoire `shell`. Il contient les schémas de programme C d'un TP que vous réaliserez en fin d'année. On souhaite renommer la procédure `av_car` en `avancerCar` dans tous les fichiers où elle est utilisée.

19. Avec `grep`, cherchez dans tous les fichiers `.c` ou `.h` la chaîne `av_car`.

```
grep av_car *.[ch]
```

20. Pour renommer la procédure, il faut intervenir (éditer) chacun des fichiers où elle apparaît. La commande `grep` dispose d'une option pour n'afficher que le nom des fichiers où la chaîne recherchée apparaît. Ajoutez cette option à la commande précédente.

```
grep -l av_car *.[ch]
```

21. On sait maintenant obtenir la liste des fichiers pertinents. Avec votre éditeur favori (`gvim`, `gedit`, `vscode` ou `geany`), ouvrez en une seule commande tous les fichiers résultants de la recherche précédente (sans utiliser de copier-coller).

```
code $(grep -l av_car *.[ch])
```

22. Utilisez la commande *rechercher/remplacer* de votre éditeur pour remplacer en une seule fois toutes les occurrences de `av_car` par `avancerCar` :
    - dans Gedit vous ne pouvez pas rechercher/remplacer sur tous les fichiers ouverts... Opter pour un autre éditeur ?
    - dans Geany (raccourci Ctrl-H), puis dans le dialogue de recherche, déplier `Remplacer tout` et choisissez `Dans la session` ;
    - dans VSCode (raccourci Ctrl-Shift-H), puis cliquez sur l'icône à droite de la chaîne de remplacement (***`Replace All`***) ;
    - dans GVim il faut passer par le mode *ligne* avec la commande `:bufdo` qui applique une commande à tous les fichiers ouverts :
      `:bufdo %s/av_car/avancerCar/g`
      `%` indique que la commande `s` s'applique à toutes les lignes du fichiers et `g` à la fin précise qu'il faut remplacer toutes les occurrences sur une ligne donnée.
      Pour tout enregistrer et quitter :
      `:wqa`

Compte-rendu

Vous rendrez à l'enseignant un exemplaire du sujet complété à la fin du TP. Si vous n'avez pas fini, utilisez l'exemplaire qui vous reste pour terminer pendant votre temps libre.

On souhaite maintenant *projeter* certaines colonnes, par exemple pour ne garder que le login ou que le nom. On peut utiliser pour ce faire la commande `cut`. Cette commande possède deux modes :

(i) projection de colonnes de caractères identifiées par leur position sur la ligne ;

(ii) projection de *champs de caractères* délimités par un caractère spécial.

On utilisera ici le mode *(ii)* puisque les informations sont délimitées par `':'`. La commande :

```
cut -d ':' -f 1,4 Liste1A.txt
```

affiche les champs 1 et 4 de toutes les lignes du fichier. Testez cette commande.

5. Faites afficher la liste des logins de tous les étudiants.

```
cut -d ':' -f 1 liste1A.txt
```

6. Faites afficher la liste des noms et prénoms de tous les étudiants.

```
cut -d ':' -f 2,3 liste1A.txt
```

3 Tri du fichier selon divers critères

La commande `sort` permet de trier les lignes d'un fichier texte. L'ordre des caractères peut varier selon la langue utilisée (français pour nous). Essayez la commande :

```
sort Liste1A.txt
```

Comme `cut`, `sort` permet de sélectionner les champs sur lesquels on veut trier : l'option `-t` sert à définir le caractère séparateur de champs et l'option `-k` définit le champ sur lequel on trie. Notez que pour trier sur plusieurs critères il faut répéter l'option `-k`.

7. Faites afficher la liste des étudiants triée sur le nom. Pour vérifier, il est recommandé de renvoyer la sortie du tri dans un tube vers la commande `less`.

```
sort -t ':' -k 2 Liste1A.txt | less
```

8. Faites afficher la liste des étudiants triée sur le groupe puis sur le nom.

```
sort -t ':' -k 5 -k 2 Liste1A.txt | less
```

4 Combinaison de commandes

Les commandes précédentes permettent de faire des choses utiles, mais prennent tout leur intérêt lorsqu'on les combine avec des tubes.

9. En combinant `grep` et `cut`, faites afficher le nom, le prénom et le courriel de tous les étudiants de votre groupe de TP.

```
grep TP1B Liste1A.txt | cut -d ':' -f 2,3,4
```

10. Modifiez la commande précédente pour que le résultat soit trié sur le nom des étudiants.

```
grep TP1B Liste1A.txt | cut -d ':' -f 2,3,4 | sort -t ':' -k 1
```

11. Faites afficher uniquement le nom de tous les « Clement » du groupe 1 (`TP1A` et `TP1B`).

```
grep TP1 Liste1A.txt | grep Clement | cut -d ':' -f 2
```


12. En combinant `grep` et `wc`, donnez la commande qui affiche le nombre d'étudiants de votre groupe de TP.

```
grep TP1B liste1A.txt | wc -l          → 14
```

13. Il existe une option de la commande `grep` donnant le même résultat sans utiliser `wc` (faire `man grep`).

```
grep -c TP1B liste1A.txt          → 14
```

14. L'option `-u` de `sort` permet d'éliminer les doublons lors d'un tri. Faites afficher la liste des groupes présents dans le fichier (sans répétition).

```
sort -t ':' -k 5 -u liste1A.txt | cut -d ':' -f 5
```

15. La commande `uniq` a un peu le même effet que l'option `-u` de `sort` : elle élimine les lignes dupliquées (consécutives, il faut donc que le fichier soit préalablement trié). Elle offre de plus une option pour compter le nombre de répétitions de chaque ligne.
Donnez la commande qui affiche la liste des groupes avec le nombre d'étudiants dans chaque groupe :

```
sort -t ':' -k 5 liste1A.txt | cut -d ':' -f 5 | uniq -c
```

(14 pour tous les groupes)

5 Modification des résultats

On peut souhaiter modifier les résultats produits par les commandes. On voudrait par exemple obtenir la liste des noms et prénoms des étudiants d'un groupe donné. On peut aisément combiner `grep`, `cut` (et éventuellement `sort`) pour obtenir cette liste, mais on aura dans le résultat un `':'` entre le prénom et le nom. La commande `sed` (*stream editor*) peut nous aider à résoudre ce problème. Cette commande permet (entre autres) d'appliquer des commandes de *recherche et remplacement* sur toutes les lignes d'un fichier.
La forme utile ici est la suivante :

```
sed -e 's/chaîneARemplacer/chaîneDeRemplacement/'
```

16. Faites afficher le nom et le prénom de tous les étudiants de votre groupe de TP, en triant **sur le prénom** et en remplaçant le `':'` par un espace.

```
grep TP1B liste1A.txt | cut -d ':' -f 2,3 | sort -t ':' -k 2 | sed -e 's/:/ /'
```

17. Faites afficher le nom, le prénom et le groupe de TP de tous les étudiants de votre groupe de TD, en remplaçant le groupe de TP par le groupe de TD correspondant, par exemple `TP2C` ⇒ `TD2` et `TP2D` ⇒ `TD2`.

```
grep TP1 liste1A.txt | cut -d ':' -f 2,3,5 | sed -e 's/TP1[AB]/TD1/'
```

6 Retour sur les substitutions : résultat d'une commande

18. Donnez la commande qui affiche par exemple (avec `echo`) :
`Le groupe TP3F comporte XX étudiants`
où bien sûr `XX` est le nombre d'étudiant du groupe TP3F.

```
echo "Le groupe TP3F comporte $(grep TP3F liste1A.txt | wc -l) étudiants"
```