TD2 : Traitement de flux de type texte avec des filtres

*Recherche dans du texte – **grep***

Le commande **grep** permet de rechercher une chaîne dans du texte. Elle peut s'utiliser sur un fichier avec la syntaxe **grep ch fich**, cela affiche toutes les lignes du fichier **fich** où se trouve la chaîne **ch**.

Son utilisation la plus courante est via **un pipe**, pour rechercher dans la sortie d'une commande : **commande arg | grep ch** : affiche toutes les lignes produites par commande arg où se trouve la chaîne ch.

Exercice 1 :

1. Sur votre répertoire d'accueil, exécutez la commande :
   `ls /usr/bin>commandes_unix.txt`
   Que se passe-t-il ?
2. Cherchez dans le fichier le mot **who**.
3. Quelle option de la commande **grep** permet d'afficher le numéro de la ligne de la chaîne recherchée ?
4. Testez cette option.
5. Quelle option permet d'afficher le nombre de lignes comportant la chaîne recherchée ?
6. Retrouver le même résultat en utilisant un enchaînement de la commande **grep** avec la commande **wc**.

Exercice 2 :

1. Copier le fichier /etc/passwd en un fichier passwd sous votre répertoire de connexion.
2. Afficher les 5 premières lignes et les 5 dernières lignes du fichier passwd
3. Afficher le fichier passwd en remplaçant le caractère : par /
4. Afficher le résultat précédent page par page
5. Afficher les lignes du fichier passwd commençant par des noms de login de 3 ou 4 caractères.
6. Triez passwd sur le nom
7. Extraire les noms de login et UID puis triez suivant les UID, le tout en une seule commande, vous redirigerez le tout vers un fichier
8. Trouver le nombre des utilisateurs travaillant sur le système, ainsi que leurs identités.

*Recherche de fichiers - **find***

La commande **find** permet de chercher un ou des fichiers dans une arborescence. Elle peut à la fois effectuer la recherche et appliquer une action sur chaque fichier trouvé. De plus, les critères de recherche peuvent porter sur d'autres métadonnées que le nom de fichier.

Syntaxe :

```
find chemin expression(s) [action]
```

chemin : endroit où chercher ; "." si on veut chercher à partir du dossier courant.

Expression (s)	Critères de recherche
-name nom	Nom du fichier
-iname nom	Nom du fichier, sans tenir compte de la casse
-user nom	Le propriétaire du fichier est **nom**
-type [d|f|l …]	Fichier de type répertoire « **d** », normal « **f** », lien **l** …
-perm nummode	Fichier ayant les permissions **nummode**

Action	Description
-print	Affiche le nom complet du fichier
-exec commande {} \;	Exécute la commande spécifiée sur chaque fichier ; **{}** représente successivement chaque fichier trouvé **\;** indique que l'action **-exec** est finie

Exemples :

- ✓ Recherche de tous les fichiers **pdf** dans le dossier **/home/bidule** et ses sous-dossiers :

```
find /home/bidule -name "*.pdf"
```

- ✓ Recherche des fichiers **pdf** dans le dossier courant et ses sous dossiers, copie vers **/tmp** :

```
find . -name "*.pdf" -exec cp '{}' /tmp \;
```

Exercice 3 (commandes find et grep):

I. Cherchez dans toute l'arborescence les fichiers :
1. dont le nom se termine par .c
2. Commençant par X ou x.
3. Dont les noms ne contiennent pas de chiffre.
4. Commence par «a» et dont la deuxième lettre est «s» ou «t»
5. D'extension .doc
6. Est constitué de deux lettres exactement
7. Commence et finit par un chiffre
8. Dont vous êtes propriétaire.

II. Donner la commande qui permet de rechercher depuis votre répertoire d'accueil tous les dossiers comportant la chaîne « rep »

En utilisant la partie action, copier tous les fichiers trouvés précédemment dans le répertoire backup (à créer) se trouvant au niveau de votre répertoire d'accueil.

Exercice 4 :

1. Ecrire un fichier « liste » contenant une liste de prénoms, noms et dates de naissance sous la forme suivante : prénom nom yyyy/mm/jj , dont les champs sont séparés par quatre caractères d'espace
2. Trier le fichier liste selon l'ordre alphabétique

3. Trier le fichier liste selon leur date de naissance et afficher le résultat sur la sortie standard et dans le fichier « date_naissance »
4. Diminuer l'ensemble des espaces entre les différents champs à un seul espace, de changer le caractère '/' par le caractère '-' et rediriger le résultat vers le fichier « date_naissance_2 »
5. Trier le fichier date_naissance_2 selon le numéro de jour (par exemple la ligne prénom1 nom1 1977/04/05 vient avant la ligne prénom2 nom2 1965/02/20)

Exercice 5:

Soit le fichier index.html suivant :

```
<table class="contact">
        <tr>
                <!—ceci est un commentaire -->
          <th>Region</th>
          <th>Départements</th>
        </tr>
        <tr>
          <td rowspan={{ depts|count }}>{{ depts[0].nom_region }}</td>
          <td>{{ depts[0].nom_dept }}</td>
        </tr>
        {% for row in depts[1:] %}
        <tr>
          <td><a href="/dept/{{ row.id_dept }}">{{ row.nom_dept }}</a></td>
        </tr>
        {% endfor %}
    </table>
```

1. Remplacer les tirets "_" par des espaces "-".
2. Afficher les six premières lignes du fichier.
3. Ajouter « - » au début de chaque ligne.
4. Remplacer les fins de lignes par « : »
5. Afficher les lignes d'un fichier comprises entre une ligne contenant «for» et une autre contenant « endfor ».
6. Supprimer les commentaires du fichier