TP4 – Shell – filtrage

OBJECTIFS :

- Manipuler les commandes de recherche, de redirection, et de filtrage
- Pouvoir combiner ces commandes pour réaliser un travail donné

RESSOURCES : document de TD intitulé « **Filtrages et parcours récursifs** »

1- Flux et redirections

La plupart des commandes « shell » génèrent un résultat sous forme d'un flux de données qui peut être exploité de 3 façons :

- **Affiché** dans dans **terminal** (par défaut)
- **Dirigé** dans un **fichier** avec les signes **>** ou **>>**
  - ls *.c **>** mes_fichiers_c : enregistre les noms de tous les fichiers « .c » du dossier courant dans le fichier « mes_fichiers_c »
  - ls *.h **>>** mes_fichiers_c : ajoute les noms de tous les fichiers « .h » du dossier courant à la fin du fichier « mes_fichiers_c »
- **Dirigé** vers l'entrée d'une **deuxième commande** au travers d'un pipe (tube) avec le signe « **|** »
  - cat mes_fichiers_c **|** wc : le flux résultat de la commande « cat » (contenu du fichier « mes_fichiers_c ») est pris comme entrée de la commande « wc » qui compte le nombre de lignes, de mots et de caractères et affiche le résultat dans le terminal.

Tester ces commandes dans un dossier qui contient des fichiers .c et des fichiers .h
Chaque terminal shell ouvert porte un numéro (0, 1, …), et est accessible avec le nom /dev/pts/n (où n est le numéro du terminal). On peut donc y diriger le résultat d'une commande comme pour un fichier.
Exécuter la commande echo bonjour de manière à ce que son résultat soit affiché dans un terminal différent du terminal de lancement de la commande.

2- Recherche dans une arborescence

La commande **find** <chemin> … <expression> [<action>] réalise un parcours de l'arborescence indiquée en recherchant les objets qui vérifient l'expression indiquée, et peut exécuter dessus une ou plusieurs actions (voir, dans le document de TD, les différentes options).

- find . –name '*.c' –print : effectue une recherche dans l'arborescence du dossier courant de tous les objets portant un nom se terminant par « .c » et affiche le nom dans le terminal
- find . –name '*.c' –print | wc –l : exécuter cette ligne de commande et vérifier le résultat
- find . –name '*.c' –exec wc –l {} \ ; : exécuter cette ligne de commande et vérifier le résultat. Les {} indiquent que les arguments du wc proviennent du résultat du find et le ; (précédé par \; pour éviter son interprétation) délimite la fin du exec.

Expliquer la différence entre les résultats des deux dernières lignes de commande

3- Filtrage

3.1 Filtrage selon motif : La commande « **grep** » permet de filtrer un flux de donnée (entrée clavier, fichier, résultat d'une commande) selon un motif et d'afficher les lignes contenant ce motif :

- grep main *.c permet d'afficher toutes les lignes contenant le mot « main » dans tous les fichiers « .c » du dossier courant.
- ps –ef **|** grep *nom_login* : le flux résultat de la commande ps –ef (liste complète des processus) est dirigé vers l'entrée de la commande grep qui ne garde que les lignes contenant le motif indiqué en argument (*nom_login* : votre nom de login)

Exercice : afficher le nombre de processus vous appartenant.

Utilisation des expressions régulières : des méta-caractères peuvent être utilisés et interprétés pour caractériser une ligne ou une partie de ligne. En voir les exemples donnés dans le document de TD et les tester sur le fichier « Fichier_test.txt » fourni.

Exercices :

- Ecrire et tester la commande qui permet de supprimer tous les fichiers .o présents dans l'arborescence du répertoire courant. Conseil : les afficher dans un premier temps avant de les supprimer.
- Ecrire et tester la commande qui permet de calculer la somme du nombre de caractères présents dans tous les fichiers ordinaires de l'arborescence du répertoire courant.
- Ecrire et tester la commande qui permet de compter le nombre de « return » dans tous les fichiers .c de l'arborescence du répertoire courant.

3.2 Tri : voir, dans le document de TD, le fonctionnement de la commande « **sort** » (page 8) et tester les exemples donnés sur le fichier « Fichier_test.txt » fourni.

3.3 Filtrage de lignes par leur position : voir, dans le document de TD, le fonctionnement des commandes **« head** » et « **tail** », et tester les exemples donnés sur le fichier « Fichier_test.txt » fourni.

Exercice : Ecrire et tester la commande qui permet de lister les caractéristiques complètes des 3 fichiers les plus récents du répertoire courant.

3.4 Sélection de zones d'une ligne : voir, dans le document de TD, le fonctionnement de la commande **« cut** » et tester les exemples donnés sur le fichier « Fichier_test.txt » fourni.

Avec l'option -d on peut indiquer le séparateur selon lequel le découpage de la ligne est réalisé. Par exemple : cut -d\; -f2 Fichier_notes.csv sélectionne la deuxième zone de chaque ligne de Fichier_notes en utilisant le ; comme séparateur (précédé ici par \ pour éviter qu'il ne soit interprété). Fichier_notes.csv contient un tableau simple en .csv (avec ; comme séparateur) qui fournit, sur chaque ligne, le groupe, le nom de l'étudiant, une suite de note, la moyenne, et la décision du jury (ADM pour admis, AJ pour ajourné).

Exercices :

- Ecrire et tester la commande qui permet de lister les étudiants admis (nom et moyenne) dans le fichier Fichier_notes.csv.
- Ecrire et tester la commande qui permet de lister les étudiants ajournés du groupe B

3.5 Filtrage avec substitution de caractères : voir, dans le document de TD, le fonctionnement de la commande **« tr** » et tester les exemples donnés sur le fichier « Fichier_test.txt » fourni.

Exercices :

- Ecrire et tester la commande qui permet de compter le nombre de voyelles dans Fichier_test.txt
- Ecrire et tester la commande qui permet de lister tous les répertoires du répertoire courant en donnant le nom et la date de dernière modification.

Un peu plus : Vérifier le fonctionnement de la commande sed (commande man ou internet).
L'utiliser pour calculer la longueur de la plus longue ligne d'un fichier donné.
Un idée ! Penser au tri de lignes composées du même caractère et de longueurs différentes.