Les commandes de base UNIX

TP 4 find et grep

Comme pour le TP3, ces questions sont des questions de cours, les réponses sont donc dans le cours, l'aide mémoire et surtout, il est nécessaire de consulter le **man** pour plus de détails (commande **less** pour se déplacer dans les pages de man).

SERIE 3 : Exercices sur find et grep, étude de leurs options

Testez les commandes sur des fichiers et répertoires d'essai pour vous faire la main et comprendre ce qui se passe.

Le fichier /usr/local/public/SYS/germinal.txt est à utiliser pour tester.

Les options de *find*

Voir en particulier *–name, -mtime, newer, -type, -size*

1. Comment chercher ***tous les fichiers commençant par un «a» majuscule ou une minuscule, suivi d'éventuellement quelques lettres ou chiffres, par un chiffre entre 3 et 6*** ?

   Voir option -name qui permet de spécifier le nom du ou des fichiers recherchés. On peut indiquer le nom d'un fichier complet (fichier.txt), ou utiliser des expressions régulières (celles du shell, pas celles de grep...) :

   - L'étoile (*) désigne «un ou plusieurs caractères»;
   - Le point d'interrogation (?) remplace un (et un seul) caractère quelconque;
   - Les crochets permettent de désigner une série de caractères au choix.

   Dans notre cas, le premier caractère est un «a» ou un «A» ([aA]), suivi de quelque chose (*) et terminé par un chiffre entre 3 et 6 ([3456] ou [3-6]).

   Ecrire la commande

2. Comment fait-on pour indiquer que ***le fichier recherché a été modifié il y a plus de 30 jours ? Il y a 30 jours ? Il y a moins de 30 jours?***
3. Comment faire pour dire que le fichier a été ***modifié plus récemment*** qu'un autre fichier donné ?
4. Comment fait-on pour spécifier que le ***fichier recherché est un répertoire*** ?
5. Comment indiquer que le fichier recherché ***à une taille supérieure à une taille donnée*** ?

Les options de *grep*

Voir en particulier : –num, -A, -B, -C, -n, -c, -i, -l, -v, -L

- *-num* : le *numéro* indique le nombre de lignes de contexte que l'on veut voir figurer avant et après la ligne où figure le mot recherché. Par exemple, si on veut trois lignes de contexte, avant et après le mot (soit sept lignes au total), on tape : grep -3 ...
- -A *num* (*after*) : le *numéro* indique le nombre de lignes qui doivent suivre la ligne où figure le mot.
- -B *num* (*before*) : le *numéro* indique le nombre de lignes qui doivent précéder la ligne où figure le mot.
- -C (*context*), qui donne deux lignes de contexte avant et après.
- Etc….

Les métacaractères \ (\)

Pour une sous chaîne, on utilise la syntaxe **\ (expression régulière)**, cette sous chaîne sera identifié par un chiffre compris par 1 et 9 (suivant l'ordre de définition). La structure \(mot1\|mot2\) permet de chercher plusieurs mots. Le symbole pipe (|) signifie ici «ou». Avec grep, il faut « backslasher » le pipe (\|) car c'est un caractère spécial.

Exercices

1. Quelles sont les options de grep qui permettent d'obtenir des lignes de contexte (qui précèdent et/ou suivent la ligne où figure le mot) ? Donnez quelques exemples, tapez ces commandes
2. Comment faire apparaître le numéro de la ligne où figure le mot recherché ? Que se passe-t-il quand on demande également des lignes de contexte ? (-nC)
3. Comment faire pour afficher le nombre d'occurrences du mot recherché ?
4. Comment faire pour que grep ignore la casse des caractères (différences entre majuscules et minuscules) dans sa recherche ?
5. Comment faire pour faire apparaître non pas les lignes où figurent le mot, mais les noms des fichiers ?
6. Comment faire apparaître les lignes où ne figure pas le mot recherché ?
7. Comment faire apparaître les noms des fichiers ne contenant pas le mot recherché ?
8. Comment faire pour que grep ne recherche que les lignes où figure le mot tel quel, et non pas ses variantes ? Par exemple : on cherche le mot «travail», mais pas «travailleur» ou «travailler».
9. Comment faire pour chercher plusieurs mots à la fois en faisant apparaître les numéros des lignes

Introduction aux expressions régulières ou rationnelles

grep recherche des chaînes de caractères, qui peuvent être un mot complet («terre»), une suite de lettres («tre»), ou une expression régulière. Les expressions régulières sont des formules qui représentent des chaînes de caractères. On cherche alors non pas un mot précis, mais des suites de caractères correspondant aux critères demandés. Elles sont d'un usage fréquent avec ***grep***, mais aussi avec des commandes comme ***less***, ***awk***, ou encore au sein d'un éditeur, ***ed***, ***sed*** et ***vi***.

«***Expression régulière***» (*Regular expression* en anglais) se traduit en bon français par «expression rationnelle (voir wikipédia) », mais l'usage est de dire «régulières». Elles sont :

☛ '.' : un joker, qui représente un caractère unique quelconque (sauf caractères de contrôle et fin de ligne).

☛ '[ ]' : des classes de caractères entre crochets [ ].

☛ '^' : le chapeau (*caret* en anglais), au début d'une classe de caractères entre crochets, signifie qu'on considère le complément de cette classe (l'ensemble des caractères qui ne sont pas dans la classe).

☛ '^' et '$' : représentent respectivement un début et une fin de ligne

L'expression régulière **^chaine$** identifie les lignes qui contiennent strictement la chaîne **chaine**

L'expression régulière **^$** identifie une ligne vide

❖ **Il faut toutefois prendre un soin particulier lorsque l'on utilise les caractères spéciaux $, *, [,], ^, |, (,) et \ dans l'expression régulière car ces caractères ont une signification particulière pour le shell. Il vaut mieux alors mettre l'expression régulière entre apostrophes, comme dans l'exemple ci-dessus.**

Exemples avec grep :

Expression	Valeur
abc	Cherche la chaîne abc n'importe où dans la ligne.
^abc	Cherche la chaîne abc en début de ligne.
abc$	Cherche la chaîne abc en fin de ligne.
^abc$	Cherche les lignes ne contenant que abc (commençant et se terminant par abc).
st[a-z][a-z]ic	Cherche n'importe quelle chaîne de caractères commençant par st, suivie de deux lettres minuscules et se terminant par ic (i.e static).
^int * main	Cherche n'importe quelle chaîne de caractères en début de ligne commençant par int, suivie de n'importe quoi et se terminant par argc (i.e int main (int argc, char **argv)).
X[0-2][0-9]	Cherche n'importe quelle chaîne de caractères commençant par X, suivie de 0, 1, ou 2 et se terminant par un chiffre (i.e X11).
.ac	cherche les chaînes de 3 caractères qui se terminent par « ac »
[a-z]	cherche n'importe quelle lettre minuscule (non-accentuée)
[^a-z]	cherche n'importe quel caractère qui n'est pas une lettre minuscule non-accentuée
[st]ac	représente entre autres « sac » et « tac »
[^f]ac	représente les mots de trois lettres qui se terminent par « ac » et ne commencent pas par « f »
^[st]ac	représente les mots « sac » et « tac » en début de ligne
[st]ac$	représente les mots « sac » et « tac » en fin de ligne
^trax$	représente le mot « trax » seul sur une ligne

grep avec expressions régulières

Ainsi, nous pourrions demander à **grep** de nous afficher toutes les lignes qui commencent par int (^int en paramètre) de tous les fichiers se terminant par .c, avec les numéros de ligne (option **-n**) :

grep -n '^int' *.c
resultat par exemple
creer.c:118:int main
modifier.c:125:int main (int argc, char **argv)

Exercices

1. Chercher toutes les lignes commençant par «a» ou «A».
2. Chercher toutes les lignes finissant par «rs».
3. Chercher toutes les lignes contenant au moins un chiffre.
4. Chercher toutes les lignes commençant par une majuscule.
5. Chercher toutes les lignes commençant par «B», «E» ou «Q».
6. Chercher toutes les lignes finissant par un point d'exclamation.
7. que donnent les expressions :

   bof[A-D]
   12[2-5]2
   12[2-56] 2 ([2-56] intervalle de 2 à 5 et 6 (et non pas 56))
   z[a-dA-D]y
   [1-3-]3
   A quoi correspond l'intervalle : [a-cI-K1-3]

   Vérifiez les avec grep

Questions subsidiaires facultatives

8. Chercher toutes les lignes ne finissant pas par un signe de ponctuation (point, virgule, point-virgule, deux-points, point d'interrogation, point d'exclamation)
9. Comment chercher tous les mots contenant un «r» précédé de n'importe quelle lettre majuscule ou minuscule ?
10. Chercher tous les mots dont la seconde lettre est un «r».