Les commandes de base UNIX

TP 4 : corrigé find et grep

Les options de **FIND**

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

   C'est l'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]). On écrit donc :

   find . -name '[aA]*[3-6]' -print

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?**

   C'est l'option -mtime qui permet de donner une indication en jours. La syntaxe varie selon ce que l'on souhaite :

   - ***-mtime 30*** : le fichier a été modifié il y a 30 jours;
   - ***-mtime +30*** : le fichier a été modifié il y a 30 jours ou plus;
   - ***-mtime -30*** : le fichier a été modifié il y a 30 jours ou moins.

3. **Comment faire pour dire que le fichier a été modifié plus récemment qu'un autre fichier donné ?**

   On utilise l'option ***-newer*** («plus récent»). Par exemple, on cherche un fichier .tex modifié plus récemment que fichier.txt :

   find . -newer fichier.txt -name '*.txt'

   On peut raffiner la demande, en combinant cette option avec l'option -mtime : cherchons les fichiers modifiés plus récemment que fichier.txt mais il y a plus de 5 jours :

   find . -newer fichier.txt -mtime +5 -name '*.txt'

4. **Comment fait-on pour spécifier que le fichier recherché est un répertoire ?**

   On utilise l'option ***-type*** pour spécifier le genre de fichier recherché : les principaux sont f (*file*) pour un fichier normal, et d (*directory*) pour un répertoire. On tape donc :

   find . -type d

5. **Comment indiquer que le fichier recherché à une taille supérieure à une taille donnée ?**

   On utilise l'option ***-size,*** suivie d'un nombre et d'une lettre indiquant l'unité de mesure (c : octets, k : kilo-octets). Comme pour -mtime, on utilise +, - ou [rien] pour indiquer que la taille est, respectivement, supérieure, inférieure ou égale à la valeur donnée.

   Par exemple, on recherche un fichier modifié il y a moins de 12 jours et dont la taille soit supérieure à 30 K :

   find . -type f -size +30k -mtime -12 -print

Les options de GREP

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) ?**

   Il y en a plusieurs, qui se recoupent :

   - ***-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 la 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. Si on en veut quatre, on tapera :

     grep -A 4 ...

   - **-B *num*** (*before*) : le *numéro* indique le nombre de lignes qui doivent précéder la ligne où figure le mot. Si on en veut dix, on tape :

     grep -B 10 ...

   - **-C (*context*),** qui donne deux lignes de contexte avant et après. En fait, les trois lignes suivantes sont strictement équivalentes :

     grep -2 ...
     grep -C ...
     grep -A 2 -B 2 ...

2. **Comment faire apparaître le numéro de la ligne où figure le mot recherché ?**

   Avecl'option **-n** (*number*); le numéro figure tout au début de la ligne, suivi d'un deux-points (:) et du texte. Par exemple :

$ ***grep -n violon verlaine.tex***

12:des violons de l'automne

Quand on fait une recherche dans plusieurs fichiers, le nom du fichier figure d'abord, puis le numéro de la ligne, et enfin le texte, le tout séparé par des deux-points. Par exemple :

$ ***grep –n violon ****

verlaine.tex:12:des violons de l'automne
orchestre:45:Cordes : contrebasse, violoncelle, alto, violons.

Que se passe-t-il quand on demande également des lignes de contexte ?

La disposition générale ne change pas, par contre, le signe utilisé pour séparer la ligne de son numéro est un tiret (-) quand il s'agit des lignes de contexte, et un deux-points quand il s'agit de la ligne voulue. Par exemple :

$" ***grep -nC violon verlaine.tex***

10-
11-Les sanglots longs
12:des violons de l'automne
13-bercent mon coeur
14-d'une langueur monotone

3. **Comment faire pour afficher le nombre d'occurrences du mot recherché ?**

   On utilise l'option -c (*count*) :

   $ grep -c violon *
   verlaine.tex:1
   orchestre:1

4. **Comment faire pour que grep ignore la casse des caractères (différence entre majuscules et minuscules) dans sa recherche ?**

   Par défaut, grep fait la différence entre les majuscules et les minuscules; pour invalider ce comportement, on utilise l'option -i (*ignor ecasse*).

5. **Comment faire pour faire apparaître non pas les lignes où figurent le mot, mais les noms des fichiers ?**

   C'est l'option -l qui permet de faire cela : afficher les noms des fichiers où figure au moins une fois la chaîne de caractères recherchée → grep- l 'la chaine'

6. **Comment faire apparaître les lignes où ne figure pas le mot recherché ?**

   On veut en fait inverser le sens de la recherche : c'est l'option -v qui fait cela.

7. **Comment faire apparaître les noms des fichiers ne contenant pas le mot recherché ?**

   On utilise l'option -L, qui affiche les noms de fichiers où ne figure pas la chaîne de caractères recherchée. Il ne faut bien sûr pas confondre les options -l et -L...

8. **Comment faire pour que grep ne recherche que les lignes où figure le mot tel quel, et non pas ses variantes ?**

   C'est l'option -w (comme *word*) qui sert à cela : un mot complet est délimité comme suit :

   - Début : la chaîne de caractères est placée au début d'une ligne, ou précédée d'un blanc, d'une tabulation ou d'une ponctuation.
   - Fin : la chaîne de caractère est placée en fin de ligne, ou suivie d'un blanc, d'une tabulation ou d'une ponctuation.

   Si donc on veut chercher «travail» et aucune forme dérivée de ce mot, on écrit :

   grep -w travail mon-fichier

9. **Comment faire pour chercher plusieurs mots à la fois en faisant apparaître les numéros des lignes ?**

   On veut chercher toutes les occurrences des mots «terre» et «ciel» dans les deux premiers chapitres de la première partie de *Germinal*, avec les numéros des lignes. On propose deux solutions, la première utilisant les ressources de la syntaxe de grep, la seconde utilisant l'option -f avec un fichier.

   1. **Syntaxe de grep** : La structure \(mot1\|mot2\) permet de chercher plusieurs mots. Ici, on tape la ligne suivante :

      ***grep '\(ciel\|terre\)'* fichier**

   On met des apostrophes de part et d'autre de l'expression pour la protéger contre le shell, c'est-à-dire pour que le shell ne cherche pas à interpréter l'expression.

   2. **Option «-f *fichier*»** : dans un fichier quelconque, que nous appellerons liste, on indique les mots que l'on recherche : «ciel» et «terre». Chaque ligne correspond à un mot recherché. Il ne faut donc pas mettre de ligne comme

      terre ciel

   car le programme chercherait la chaîne de caractères «terre ciel», qui est assez improbable en français. Il ne faut pas non plus laisser de ligne blanche : le programme afficherait l'ensemble du texte.

   Quelle que soit la solution retenue, on veut ensuite afficher le numéro des lignes (option -n); d'autre part, pour que la recherche soit exhaustive, il vaut mieux que grep ne fasse pas de différence entre les majuscules et les minuscules, avec l'option -i (*ignore case*, ignorer la casse des caractères). Il faut aussi décider si on cherche les mots tels quels, sans leurs variantes (comme «terre» au pluriel), ou si on accepte ces variantes. Si on ne veut que le mot sans ses dérivés, on utilise l'option -w.

   Pour désigner les deux fichiers où faire la recherche, on peut les écrire littéralement :

   zola1.txt zola2.txt

   ou, mieux, utiliser les joker du shell :

zola[12].txt

[12] signifie «le caractère 1 ou le caractère 2».

Finalement, on peut taper, au choix :

grep -inw -f liste zola1.txt zola2.txt
grep -inw -f liste zola[12].txt
grep -inw '\(ciel\|terre\)' zola1.txt zola2.txt
grep -inw '\(ciel\|terre\)' zola[12].txt

Et on obtient :

zola1.txt:13:ciel, le pavé se déroulait avec la rectitude d'une jetée, au milieu de
zola1.txt:36:brûlaient si haut dans le ciel mort, pareils à des lunes fumeuses. Mais, au
zola1.txt:50:besogne. Les ouvriers de la coupe à terre avaient dû travailler tar d, on
zola1.txt:124:terre, lorsqu'un accès de toux annonça le retour du charretier. Le ntement,
zola1.txt:191:bleues en plein ciel, comme des torches géantes. C'était d'une tristesse
zola1.txt:207: Le manoeuvre, après avoir vidé les berlines, s'était assis à terre,
zola1.txt:222:fois avec tout le poil roussi, une autre avec de la terre jusque dans le

(...)

Le résultat est un peu différent quand on n'utilise pas l'option -w.

Introduction aux EXPRESSIONS REGULIERES

1. **Chercher toutes les lignes commençant par «a» ou «A»**.

Il faut indiquer que l'on veut le début de la ligne, avec le chapeau (*caret* en anglais). Ensuite, on veut préciser que la ligne commence par un «a» minuscule ou majuscule. Il y a deux façons de le faire :

- Utiliser l'option -i qui fait ignorer la différence entre les majuscules et les minuscules.
- Dire que l'on cherche un «a» ou un «A». C'est à cela que servent les crochets : [abc] signifie «a ou b ou c». Ici, ce sera [aA].

Enfin, il faut protéger les signes contre le shell, pour qu'il ne les interprète pas; on met donc l'expression entre apostrophes. Remarque : la protection des expressions régulières contre le shell est une question complexe....

Il faut donc écrire :

grep -i '^a' fichier
ou

grep '^[aA]' fichier

2. **Chercher toutes les lignes finissant par «rs»**

   C'est le dollar ($) qui représente la fin de la ligne. Il faut donc écrire :

   grep 'rs$' fichier

3. **Chercher toutes les lignes contenant au moins un chiffre**

   Pour désigner un chiffre, on peut en indiquer une liste entre crochets : [0123456789]. Il est plus simple d'utiliser une classe de caractères : [0-9] qui désigne, comme la solution précédente, n'importe quel chiffre de zéro à neuf.

   Il faut donc taper :

   grep '[0-9]' fichier

4. **Chercher toutes les lignes commençant par une majuscule**

   Comme on l'a vu, c'est le chapeau qui indique le début de la ligne. Pour indiquer que l'on cherche une majuscule, on peut soit en donner une liste ([ABCDEFGHIJKLMNOPQRSTUVWXYZ]), soit utiliser une classe de caractères : [A-Z], la seconde solution étant, de loin, préférable...

   Il faut donc taper :

   grep '^[A-Z]' fichier

5. **Chercher toutes les lignes commençant par «B», «E» ou «Q»**

   Il faut indiquer entre crochets les trois lettres recherchées :

   grep '^[BEQ]' fichier

6. **Chercher toutes les lignes finissant par un point d'exclamation**

   Le point d'exclamation n'a pas de signification particulière avec grep, on peut donc le mettre tel quel :

   grep '!$' fichier

7. **que donne les expressions :**

bof[A-D] donne bofA, bofB, bofC, bofD

12[2-5]2 donne 1222, 1232, 1242, 1252

12[2-56] 2 ([2-56] intervalle de 2 à 5 et 6 (et non pas 56)) donne 1222, 1232, 1242, 1252, 1262

z[a-dA-D]y donne zay, zby, zcy, zdy, zAy, zBy, zCy, zDy

[1-3-]3 donne 13, 23, 33, -3

A quoi correspond l'intervalle : [a-cI-K1-3] intervalle de a à c, I à K et 1 à 3 (a, b, c, I, J, K, 1, 2, 3)

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)**

   Il faut donner une liste de caractères, que l'on ne veut pas voir figurer; la liste sera entre crochets, comme on l'a déjà vu, et c'est le chapeau qui signifiera, dans ce contexte, «sauf». Par exemple, si on cherche tous les «a», sauf ceux suivi de «b», «c» ou «t», on écrit :

   grep 'a[^bct]'

   Il y a une seconde difficulté, qui vient de ce que certains caractères sont spéciaux avec grep. Vous vous doutez que le chapeau est spécial quand il est placé au début de l'expression, et que le dollar l'est quand il est placé en fin d'expression. Dans notre cas :

   - Le point désigne n'importe quel caractère.
   - Le point d'interrogation signifie «le caractère qui précède apparaît 0 ou 1 fois». Avec egrep, il fonctionne tout seul, avec grep, il faut le faire précéder d'un ***backslash*** pour qu'il fonctionne; par exemple (avec grep), pour chercher «charbon» ou «vagabond», on écrit :

   grep 'ar?bo' fichier

   (chercher la suite de lettre «abo» avec un «r» facultatif entre le «a» et le «b»).

   Pour que grep interprète littéralement ces caractères, et ne les considère plus comme spéciaux, il faut les faire précéder d'un backslash (\). Si par exemple vous cherchez toutes les lignes qui se terminent par un point, il faut taper :

   grep '\.$' fichier

   Dans notre cas cependant, ces caractères sont protégés par les crochets. On peut donc écrire :

   grep '[^.,;:?!]$' fichier

   On peut aussi utiliser l'option -v, qui prend toutes les lignes où ne figure pas une chaîne de caractères donnée; dans ce cas, on tape :

   grep -v '[.,;:?!]$' fichier

9. **Comment chercher tous les mots contenant un «r» précédé de n'importe quelle lettre majuscule ou minuscule ?**

   On cherche une chaîne de caractères qui soit indifféremment au début ou au milieu d'un mot. N'importe quelle lettre, ce sont les classes de caractères [a-zA-Z] ou [:alpha:], qui sont équivalentes.

   Il y a une petite subtilité avec l'emploi de classes du second type; elles désignent un groupe de caractères, et il faut mettre une seconde paire de crochets pour dire «n'importe quel caractère de cette classe prédéfinie». On tape donc au choix :

grep '[a-zA-Z]r' fichier'
ou
grep '[[:alpha:]]r' fichier'

Attention, dans ces listes ne sont pas compris les caractères accentués...

10. **Chercher tous les mots dont la seconde lettre est un «r»**.

    C'est le symbole \< qui désigne un début de mot. La première lettre du mot est indifférente, la seconde est un «r». On écrit donc :

    grep '\<.r' fichier

    Il y a cependant un problème avec les caractères accentués, que grep considère comme des blancs. Dans ce cas, il vaut mieux procéder autrement : un mot est précédé d'un début de ligne, ou d'un blanc ou d'une tabulation. Un début de ligne, c'est le chapeau, un blanc ou une tabulation, c'est la classe de caractères [:space:].

    On va se servir du pipe (|) qui signifie «ou». Avec grep, il faut backslasher le pipe, avec egrep ce n'est pas nécessaire. On écrit donc (avec grep) :

    grep '^.r\|[[:space:]].r' fichier

    Ce n'est quand même pas si simple; les mots peuvent être précédés d'un tiret (mots composés), d'une apostrophe, de guillemets divers (``, '', «, <<), et, si l'auteur du texte n'est pas respectueux des règles de typographie, d'une ponctuation. Il y a donc bien des cas à envisager...