

Aide-mémoire commandes texte Unix

Les plus utiles, avec leurs options les plus courantes.

Filtres simples

cat	copie l'entrée sur la sortie, telle quelle.
-n	numérote les lignes, à partir de 1
-b	idem sans numérotter les lignes blanches
-s	"squeeze" les lignes blanches consécutives
head	ne garde que le début (10 lignes par défaut)
-n	garde <i>n</i> lignes
tail	ne garde que la fin (10 lignes par défaut)
-n	garde <i>n</i> lignes
+n	garde à partir de la ligne <i>n</i>
sort	trie les lignes (alphabétiquement par défaut)
-r	(<i>revert</i>) : ordre inverse
-n	ordre numérique
-k <i>n</i>	trie selon le <i>n</i> -ième champ de chaque ligne
-t 'c'	spécifie le séparateur de champ
-u	(<i>unique</i>) : élimine les lignes consécutives identiques
-S <i>size</i>	spécifie la taille de la mémoire vive utilisée (ex : 128M, 1G)
-T <i>dir</i>	spécifie le répertoire temporaire utilisé (si /tmp est trop petit)
tr	"traductions" de caractères
'ch1' 'ch2'	remplace chaque caractère de <i>ch1</i> par le caractère correspondant dans <i>ch2</i>
-d 'ch'	(<i>delete</i>) : supprime tous les caractères de <i>ch</i>
-s 'ch'	"squeeze" tous les caractères consécutifs de <i>ch</i>
	caractères spéciaux : \t (tabulation), \n (<i>new line</i>), \\ (<i>backslash</i>) chaînes spéciales : a-z, A-Z, 0-9 etc.
cut	découpe les lignes en "champs"
-fn	conservé uniquement le <i>n</i> -ième champ
-fn,m	idem champs <i>n</i> et <i>m</i>
-fn-m	idem champs <i>n</i> à <i>m</i>
-d'c'	délimiteur de champs, '\t' (TAB) par défaut
uniq	"squeeze" les lignes <i>consécutives</i> identiques (on applique souvent sort au préalable).
-c	compte le nombre de lignes identiques

Recherche de motifs

Dans les commandes ci-dessous, l'*expression* recherchée peut être un simple mot ou une expression plus complexe (voir syntaxe dans la rubrique *regular expressions* de `man grep`).

Ces deux commandes fonctionnent en filtre mais `grep` peut aussi travailler avec des fichiers passés en paramètre.

<code>grep</code>	<p><i>expression</i> [<i>fichier(s)</i>] recherche les lignes contenant l'expression. Avec <i>fichier(s)</i> en paramètre, chaque ligne est précédée du nom du fichier</p> <ul style="list-style-type: none"> -c compte le nombre de lignes qui devraient être écrites -v inverse le résultat (lignes qui ne contiennent pas l'expression) -A <i>n</i> (<i>after</i>) : affiche les <i>n</i> lignes qui suivent chaque ligne sélectionnée -B <i>n</i> (<i>before</i>) : idem lignes qui précèdent -C <i>n</i> (<i>context</i>) : idem lignes autour -f <i>file</i> lit les expressions (une par ligne) dans un fichier -i (<i>ignore case</i>) -n numérote les lignes -w (<i>word</i>) : l'expression doit former un mot complet (un "mot" est constitué de lettres, chiffres ou <i>underscore</i>)
<code>sed</code>	<p>'<i>s/expression/remplacement/g</i>' remplacement d'expressions. Sans le /<i>g</i> (<i>global</i>), seule la première occurrence de chaque ligne est remplacée</p> <ul style="list-style-type: none"> -n avec /<i>p</i> (éventuellement /<i>gp</i>), seules les lignes modifiées sont recopiées -f <i>file</i> lit les commandes (une par ligne) dans un fichier

Autres

wc	(<i>word count</i>) -c caractères -w mots -l lignes
diff	<i>f1 f2</i> lignes différentes entre deux fichiers. Si l'un des deux paramètres est -, c'est l'entrée standard -b (<i>blank</i>) ignore les différences dans le nombre d'espaces -B ignore les lignes blanches -i (<i>ignore case</i>) -q (<i>quiet</i>) indique seulement si les fichiers diffèrent -y affichage côte-à-côte
paste	<i>fichiers</i> inverse de <i>cut</i> : la sortie contient une colonne pour chaque fichier. Si l'un des deux paramètres est -, c'est l'entrée standard
join	<i>f1 f2</i> jointure : fusionne les lignes de <i>f1</i> et <i>f2</i> selon une clef présente dans les deux. Les deux fichiers doivent être triés selon cette clef.
zcat <i>f</i>	décompresse un fichier <i>.gz</i> et l'écrit sur la sortie. Utile pour ne pas décompresser le fichier en entier sur le disque. Identique à <i>gzip -cd</i> et <i>gunzip -d</i>
awk	Lit un fichier ligne par ligne, découpe chaque ligne en champs et effectue une opération donnée en argument sur les champs de chaque ligne. Le premier champ est désigné par \$1, le deuxième par \$2, ... La ligne entière est désignée par \$0. NR représente le numéro de la ligne courante. Les instructions sont passées en argument, entre apostrophes. Ex : <code>awk '{if(\$1 > 2) print \$0;}'</code> Affiche toutes les lignes pour lesquelles le premier champ est supérieur à 2. Ex : <code>awk '{sum+=\$2; print sum/NR;}'</code> Affiche pour chaque ligne la somme de tous les deuxièmes champs des lignes précédentes, divisée par le nombre de lignes.