Redirections en C-shell

D. Bogdaniuk – P. Portejoie

Samuel Le Berre	**7/11/16**
Groupe A2	

Les commandes UNIX utilisent 3 fichiers standards pour leurs entrées-sorties :

- **stdin** : le fichier d'entrée standard, canal 0,
- **stdout** : le fichier de sortie standard, canal 1.

Quand une commande s'exécute, les résultats sont normalement envoyés sur cette sortie.

- **stderr** : le fichier de sortie d'erreur standard, canal 2.

Quand une commande produit une erreur, le message d'erreur est envoyé sur cette sortie.

Cependant, il est possible de récupérer le contenu des sorties standards dans des fichiers, ainsi que de prendre le contenu d'un fichier comme entrée standard. Ces opérations sont appelées des redirections.

- ***<fichier*** : redirection de l'entrée standard à partir de ***fichier***
- ***>fichier*** : redirection de la sortie standard vers ***fichier***
- ***>>fichier*** : redirection de la sortie standard vers ***fichier***, en ajout
- ***>&fichier*** : redirection de la sortie standard et de la sortie d'erreur vers ***fichier***
- ***>>&fichier*** : redirection de la sortie standard et de la sortie d'erreur vers ***fichier***, en ajout

De plus, il est possible de rediriger séparément les sorties par `(commande > fs) >& erfs` qui redirige respectivement la sortie standard vers le fichier `fs` et la sortie d'erreur vers `erfs`.

Souvent, le résultat d'une commande doit être passé en entrée d'une autre pour obtenir le résultat final. Une solution consiste à récupérer le résultat de la première commande dans un fichier et de l'utiliser en entrée de la suivante. Mais une solution plus souple et qui n'engendre aucune création de fichier intermédiaire consiste à placer un tube (pipe) `|` entre les deux commandes.

Le tube permet de récupérer la sortie standard pour la placer en entrée de la commande suivante.

Par exemple pour compter le nombre de fichiers d'un répertoire, sous réserve que les noms de fichiers ne comportent pas d'espace, il suffit d'utiliser un tube qui relie la sortie de la commande *`ls`* à l'entrée de la commande *`wc`* : `ls | wc -w`

TD – TP

Exercice 1

Dans chacun des cas ci-dessous, identifiez les redirections selon leur type en marquant d'un X la ou les cases appropriées (n'hésitez pas à questionner votre enseignant) :

Commande	entrée standard	sortie standard	sortie d'erreur
`ls > liste`		X	
`wc -w < liste`	X		
`wc -w < liste > res`	X	X	
`cat liste res > global`		X	
`(ls f1 f2 liste res > present) >& absent`		X	X
`(ls f1 f2 liste res) >& absent`		X	X
`ypcat passwd | wc -l`	X	X	

Exercice 2

Créez un répertoire exo2 ; faites-en votre répertoire courant et créez-y les fichiers suivants :

```
essai.c
essai
essai.o
script.pl
script.res
```

```
[e1604902@ens-iutva-0392 e1604902]$ mkdir exo2
[e1604902@ens-iutva-0392 e1604902]$ cd exo2
[e1604902@ens-iutva-0392 exo2]$ touch essai.c essai essai.o script.pl script.res
```

Pour chacune des instructions données ci-après, indiquez ce que contiennent les fichiers de sortie. A chaque fois vous donnerez une copie du résultat que vous analyserez et justifierez.

1/ `ls > f1`

```
[e1604902@ens-iutva-0392 exo2]$ ls > f1
[e1604902@ens-iutva-0392 exo2]$ cat f1
essai
essai.c
essai.o
f1
script.pl
```

```
script.res
```

on crée un fichier f1 où l'on stocke le résultat de ls

2/ `ls >> f1`

```
[e1604902@ens-iutva-0392 exo2]$ ls >> f1
[e1604902@ens-iutva-0392 exo2]$ cat f1
essai
essai.c
essai.o
f1
script.pl
script.res
essai
essai.c
essai.o
f1
script.pl
script.res
```

on ajoute le résultat de ls à f1

3/ `wc -l < f1 > f2`

```
[e1604902@ens-iutva-0392 exo2]$ wc -l < f1 > f2
[e1604902@ens-iutva-0392 exo2]$ cat f1
essai
essai.c
essai.o
f1
script.pl
script.res
essai
essai.c
essai.o
f1
script.pl
script.res
essai
essai.c
essai.o
f1
script.pl
script.res
[e1604902@ens-iutva-0392 exo2]$ cat f2
12
```

On met le nombre d'élément dans f1 dans f2

4/ `(ls  *.java > f1) >& f2`

```
[e1604902@ens-iutva-0392 exo2]$ (ls  *.java > f1) >& f2
[e1604902@ens-iutva-0392 exo2]$ cat f2
ls: impossible d'accéder à '*.java': Aucun fichier ou dossier de ce type
```

On renvoie dans f2 le résultat de sortie et de sortie d'erreur. Pour cette exemple il n'y a pas de fichier finissant avec .java donc on a la sortie d'erreur dans f2.

5/ `(ls -l essai vide > f1) >& f2`

```
[e1604902@ens-iutva-0351 exo2]$ (ls -l essai vide > f1) >& f2
[e1604902@ens-iutva-0351 exo2]$ cat f1
-rw-r--r--. 1 e1604902 etud 0 18 nov.  16:13 essai
[e1604902@ens-iutva-0351 exo2]$ cat f2
ls: impossible d'accéder à 'vide': Aucun fichier ou dossier de ce type
```

6/ `(ls -l essai.c > f1) >& f2`

```
[e1604902@ens-iutva-0351 exo2]$ (ls -l essai.c > f1) >& f2
[e1604902@ens-iutva-0351 exo2]$ cat f1
-rw-r--r--. 1 e1604902 etud 0 18 nov.  16:13 essai.c
[e1604902@ens-iutva-0351 exo2]$ cat f2
```

7/ `(ls -l essai vide) >& f2`

```
[e1604902@ens-iutva-0351 exo2]$ (ls -l essai vide) >& f2
[e1604902@ens-iutva-0351 exo2]$ cat f1
-rw-r--r--. 1 e1604902 etud 0 18 nov.  16:13 essai.c
[e1604902@ens-iutva-0351 exo2]$ cat f2
ls: impossible d'accéder à 'vide': Aucun fichier ou dossier de ce type
-rw-r--r--. 1 e1604902 etud 0 18 nov.  16:13 essai
```

8/ `ls -l | wc -l > f1`

```
[e1604902@ens-iutva-0351 exo2]$ ls -l | wc -l > f1
[e1604902@ens-iutva-0351 exo2]$ cat f1
8
```

Exercice 3

La commande *cat* lit les données sur l'entrée standard jusqu'à la rencontre d'une marque de fin de fichier (saisie de *ctrl-D*) et les affiche sur la sortie standard.

La commande *sort* est sensiblement identique : elle lit les données sur l'entrée standard jusqu'à la rencontre de *ctrl-D* puis elle les affiche sur la sortie standard mais en les triant .

Remarques : - *cat f1* est équivalent à *cat < f1* (les 2 notations sont permises)
- de la même façon *sort f1* est équivalent à *sort < f1*

Vous devez répondre aux questions suivantes en utilisant ces deux commandes et les redirections. Vous devrez dans chacun des cas donner une copie d'écran justificative du résultat.

Question préliminaire :

Lancez la commande *cat* seule (ie sans arguments) puis saisissez des lignes de caractères (utilisez la touche *Entrée* à la fin de chaque ligne et *Ctrl-D* pour terminer la saisie).

Vous observez que chaque ligne saisie est doublée. L'explication est simple : tout caractère entré au clavier est systématiquement envoyé sur la sortie standard pour y être affiché, puis à chaque fin de ligne (écho généré par la touche *Entrée*) la commande *cat* envoie à son tour la ligne reçue sur la sortie standard.

Faites la même chose en lançant cette fois-ci la commande *cat > fs*. Puis affichez le contenu du fichier *fs*. Qu'observez-vous ? Expliquez.

```
[e1604902@ens-iutva-0351 exo2]$ cat > preliminaire
test
de
la
question
preliminaire
[e1604902@ens-iutva-0351 exo2]$ cat preliminaire
test
de
la
question
preliminaire
[e1604902@ens-iutva-0351 exo2]$
```

Cat a renvoyer notre saisie dans le fichier

1/ Comment créer un fichier en saisissant les lignes qu'il contiendra au clavier ?

```
[e1604902@ens-iutva-0351 TP7]$ cat > ex1
contenu de
la question 1
[e1604902@ens-iutva-0351 TP7]$ cat ex1
contenu de
la question 1
```

2/ Comment copier un fichier sans utiliser *cp* ?

```
[e1604902@ens-iutva-0351 TP7]$ cat ex1 > ex2
[e1604902@ens-iutva-0351 TP7]$ cat ex2
contenu de
la question 1
```

3/ Comment fusionner deux fichiers existants (f1 et f2) vers un troisième qui sera crée (f3), en 2 commandes consécutives ?

4/ Comment fusionner deux fichiers existants (f1 et f2) vers un troisième qui sera crée (f3), en une seule commande ?

5/ Reprenez la question 1/ (question préliminaire) cette fois-ci avec *sort*

```
[e1604902@ens-iutva-0351 TP7]$ sort > f1
test
de
la
question1
avec
sort
[e1604902@ens-iutva-0351 TP7]$ cat f1
avec
de
la
question1
 sort
test
```

sort fait la même chose que cat mais range par ordre alphabétique le contenu entré

6/ En une seule commande, comment afficher, triées, des données saisies au clavier ?

```
[e1604902@ens-iutva-0351 TP7]$ sort
a
f
g
h
j
v
a
f
g
h
j
v
```

Sort permet de trié par ordre alphabétique les données saisies.

7/ En une seule commande, comment trier des données d'un fichier vers un autre ?

```
[e1604902@ens-iutva-0351 TP7]$ cat > f1
a
g
j
f
d
[e1604902@ens-iutva-0351 TP7]$ sort f1 > f2
[e1604902@ens-iutva-0351 TP7]$ cat f2
a
d
f
g
j
```

Sort permet également de trié des éléments d'un fichier par ordre alphabétique.

8/ En une seule commande, comment fusionner deux fichiers puis afficher le résultat sous forme triée ?

```
[e1604902@ens-iutva-0351 TP7]$ cat > f1
a
h
[e1604902@ens-iutva-0351 TP7]$ cat > f2
y
f
```

```
[e1604902@ens-iutva-0351 TP7]$ sort f1 f2
a
f
h
y
[e1604902@ens-iutva-0351 TP7]$ sort f1 > f3 f2 > f3
```

Sort peut également ranger par ordre alphabétique 2 fichiers

9/ En une seule commande, comment fusionner deux fichiers dans un troisième en triant leurs données ?

```
[e1604902@ens-iutva-0351 TP7]$ cat > f1
a
h
[e1604902@ens-iutva-0351 TP7]$ cat > f2
y
f
[e1604902@ens-iutva-0351 TP7]$ cat f3
a
f
h
y
```

Exercice 4

Petit retour sur les scripts et avertissement de sécurité

Soit le script *concat.csh* que vous trouverez sur le forum habituel.

1/ Copiez le fichier dans votre environnement et lancez son exécution (préalablement, pensez aux droits) en lui fournissant comme jeu de données : **cle** pour la première chaîne demandée et **ar** pour la seconde. Que fait-il ?

2/ Relancez-le en saisissant **`cle** pour la première chaîne demandée et **ar`** pour la seconde (rappel : ` est la backquote, caractère qui s'obtient par la combinaison de touches altgr 7). Qu'observez-vous ?

3/ Editez le contenu du script et observez-en l'algorithme. Que se passerait-il si l'utilisateur saisissait *`rm -rf *`* **(ne le faites surtout pas!)** ? Que pouvez-vous en conclure en matière de sécurité ? Proposez une solution d'ordre général (sa mise en œuvre n'est pas demandée dans le cadre de cet exercice)