Redirections des entrées/sorties

Semaine 4 - séance 2

Objectifs

La *redirection des E/S* permet de remplacer l'écran et le clavier par des fichiers texte pour les commandes de l'interpréteur de commandes. Elle permet aussi de combiner plusieurs commandes pour en créer de plus puissantes.

Note sur les redirections

Lors de l'exécution d'une commande, trois fichiers texte sont ouverts par défaut et connectés à un périphérique d'entrée/sortie (voir table 1).

Nom du fichier	N° de descripteur	Périphérique d'E/S
stdin	0	clavier
stdout	1	écran
stderr	2	écran

TABLE 1 – Fichiers d'entrée/sortie

De ces périphériques vont circuler des flux d'informations. Avec le shell, il est possible de rediriger ces flux d'E/S de ou vers des fichiers. Il est aussi possible de rediriger les flux de sortie d'une commande vers l'entrée d'une autre. Cela s'effectue en utilisant l'*opérateur filtre* (via '|' le *pipe character*) et des commandes nommées *filtres*. Une ligne de commande utilisant cette technique se nomme naturellement un *pipe-line*.

Syntaxe : De façon générale la syntaxe est la suivante :
commande [opérateur_de_redirection fichier_texte]...
où opérateur_de_redirection est l'un des suivants :
- La redirection depuis le clavier est notée : 0< (ou <),
- La redirection de la sortie standard est notée : 1> (ou >) si on souhaite créer un nouveau fichier ou 1» (ou ») si on souhaite ajouter les données à la fin du fichier,
- La redirection de la sortie standard des erreurs est notée : 2> si on souhaite créer un nouveau fichier ou 2» si on souhaite ajouter les données à la fin du fichier,

Redirection des flux d'entrée-sortie

Manipulation 1
Listez le contenu d'un répertoire de votre choix mais au lieu de l'afficher, enregistrez le résultat dans un fichier de nom contenu.txt. *Affichez son contenu page par page.*

Manipulation 2
Exécutez et interprétez les commandes suivantes :

```
ls -l > liste.txt
cat liste.txt
ls >> liste.txt
more liste.txt
ls > liste.txt
cat liste.txt
```

Manipulation 3
On suppose que le fichier pasla *n'existe pas. Exécutez et interprétez les commandes suivantes :*

```
mkdir tmp
cp -v insulte.txt pasla tmp
cp -v insulte.txt pasla tmp > mess.txt
cat mess.txt
cp -v insulte.txt pasla tmp 2> mess.txt
cat mess.txt
cp -v insulte.txt pasla tmp > mess.txt 2>&1
cat mess.txt
```

Manipulation 4
Exécutez et interprétez les commandes suivantes :

```
ls -R / | more
grep "#define" /usr/include/*.h | sort | more
```

Manipulation 5
Prenez connaissance d'une des aides suivantes en tapant : man stdin, man stdout *ou* man stderr *Quel est l'intérêt d'utiliser ces noms de fichier ?*

1. Lisez et commentez le programme C suivant (il se trouve dans le fichier *TD7_saluations.c*) :

```
/* Nom du programme : salutations.c
    Fonction : attendre un nom sur l'entree standard (stdin)
    et afficher sur la sortie standard (stdout) et la sortie
    standard des erreurs (stderr)
*/

#include <stdio.h>
#include <string.h>

int main()
{
  char nom[80];

  // Affichage sur la sortie standard stdout
  printf("\n\tVotre nom SVP : ");

  // Saisie sur l'entree standard stdin
  fgets(nom, 80, stdin);

  // Affichage sur la sortie standard stdout
  printf("\n\n\t%s sur stdout.\n\n",nom);

  // Affichage sur la sortie standard des erreurs stderr
  fprintf( stderr, "\t%s sur stderr.\n", nom);

  return 0;
}
```

Manipulation 6
Compilez-le par la commande suivante :

```
gcc TD4_salutations.c -o salutations
```

Manipulation 7
Exécutez la suite de commandes suivantes et interprétez-la :

```
./salutations
echo paviot > monnom
./salutations < monnom
./salutations < monnom > fichier_stdout
./salutations < monnom > fichier_stdout 2> fichier_stderr
./salutations < monnom > fichier_stdout_stderr 2>&1
```