TD 2 : Commandes et script Shell sous Unix

Exercice 1[1] : Liens physiques et droits d'accès

1. Supposons que les numéros d'inœuds s'incrémentent par un à chaque création d'un fichier. Le prochain numéro d'inœud libre est 345. Nous exécutons les commandes suivantes :

Commandes

```
$ cd ~
$ echo "Exercices inoeuds" > ~/toto
$ cp toto t2
$ cp toto t3
$ mv t2 t4
$ ln t3 t5
$ ln t4 /etc/t6
```

Donner les numéros d'inœud des fichiers toto, t2, t3, t4, t5 et t6.
Réponse :

Nom du fichier	Numéro i-noeud	Remarque
toto	345	nouveau fichier
t2	346	nouveau fichier
t3	347	nouveau fichier
t4	346	même i-noeud que t2
t5	347	même i-noeud que t3 puisque t5 est un lien physique sur t3
t6	346	même i-noeud que t4 puisque t6 est un lien physique sur t4

2. Donner les droits d'accès (en mode symbolique puis en mode octal) du fichier toto après l'exécution successive de toutes les commandes suivantes :

Commandes

```
$ touch toto
$ chmod 354 toto
$ chmod a-x toto
$ chmod ug+r toto
$ chmod o-w toto
$ chmod g-w toto
```

Réponse :

commande	mode symbolique	mode octal
chmod 354 toto	-wxr-xr--	354
chmod a-x toto	-w-r--r--	244
chmod ug+r toto	rw-r--r--	644
chmod o-w toto	rw-r--r--	644
chmod g-w toto	rw-r--r--	644

[1]Source : Contrôle 2014 —2015

3. Considérons les commandes suivantes :

Commandes

```
$ ls -l foo
-rw-rw-rw- 1 compte1 compte1 13 Jan 2015 14:30 foo
$ umask 024
$ cat bar >> foo
$ mkdir exam
$ cat foo > exam/foobar
```

Quelles sont les permissions sur les fichiers foo, foobar et exam en supposant que bar est un fichier du répertoire courant ? Justifiez votre réponse.
Réponse :

Nom du fichier	mode symbolique	mode octal	Remarque
`foo`	`rw-rw-rw-`	666	droits donnés par `ls -l`
`exam`	`rwxr-x-wx`	753	répertoire nouvellement crée (droits d'accès calculés à partir du masque 024 (777 - 024 = 753)
`foobar`	`rw-r---w-`	642	droits d'accès déduit à partir des droits sur un répertoire nouvellement crée (`exam` par exemple) puis suppression du droit d'exécution ==> 642

Exercice 2 : Variable d'environnement, redirection et contrôle de taches

1. Comment régler la variable `PATH` avec les chemins suivants?

Chemins

```
/usr/bin
/export/home/
/bin
```

Réponse : `PATH=/usr/bin:/export/home/:/bin`

2. Quelle est la différence entre :

Commandes

```
cmd1 & ; cmd2 & ; cmd3
cmd1 & ; cmd2 & ; cmd3 &
```

Réponse : La seule différence est dans la façon par laquelle la commande `cmd3` est lancée : elle est lancée en premier plan dans `cmd1 & ; cmd2 & ; cmd3` et en arrière plan dans `cmd1 & ; cmd2 & ; cmd3 &`

3. Quelle est la différence entre :

Commandes

```
cmd1 1> sortie.txt 2>&1
cmd1 2>&1 1> sortie.txt
```

Réponse : Dans `cmd1 1> sortie.txt 2>&1`, la sortie standard est redirigée vers le fichier `sortie.txt` puis l'erreur standard est redirigée vers la sortie standard qui a été déjà redirigée vers le fichier `sortie.txt`. En conclusion, la sortie standart et l'erreur standard sont redirigées vers le même fichier `sortie.txt`. Par contre dans `cmd1 2>&1 1> sortie.txt`, toute erreur sera affichée sur l'écran et la sortie standard est redirigée vers le fichier `sortie.txt`.

Exercice 3 : Variables

1. Parmi les syntaxes suivantes, laquelle est correcte :

```
$ variable = foo
$ variable= foo
$ variable =foo
$ variable=foo
```

Réponse : `variable=foo` est la syntaxe correcte : il ne faut pas mettre d'espace ni avant ni après le signe d'affectation.

2. Donner une explication possible de l'obtention du résultat suivant :

Commandes

```
$ var = x
= x
$ var + x
+ x
$ var - x
- x
```

Réponse : Une explication possible est que `var` est le nom d'un script shell où son code est :

var

```
#!/bin/bash
echo $*
```

et ce script appartient à un répertoire qui figure dans la variable d'environnement `PATH`.

Exercice 4 : La commande `read`

1. Considérons les deux cas suivants :

Cas 1

```
$ cat datafile
the quick brown fox
$ read champ1 champ2 champ3 < datafile
```

Cas 2

```
$ echo the quick > datafile
$ read champ1 champ2 champ3 < datafile
```

Donner le résultat d'exécution des commandes suivantes pour chaque cas :

Commandes

```
$ echo Champ numéro un est $champ1
$ echo Champ numéro deux est $champ2
$ echo Champ numéro trois est $champ3
```

Réponse :

Cas	Résultat
cas 1	Champ numéro un est the Champ numéro deux est quick Champ numéro trois est brown fox
cas 2	Champ numéro un est the Champ numéro deux est quick Champ numéro trois est

2. Supposons avoir le résultat suivant :

Commandes

```
$ echo $PATH; echo $PATH > /tmp/path.txt
/usr/local/bin:/usr/bin:/bin:/usr/sbin:/sbin
```

Donner le résultat d'exécution du script Shell suivant :

Script

```
#!/bin/bash
read var1 var2 var3 var4 var5 var6 var7 < /tmp/path.txt
echo "Avant de modifier IFS"
echo $var1, $var2, $var3, $var4, $var5, $var6, $var7
IFS=:
read var1 var2 var3 var4 var5 var6 var7 < /tmp/path.txt
echo "Pour IFS=:"
echo $var1, $var2, $var3, $var4, $var5, $var6, $var7
```

Réponse :

Résultat

```
Avant de modifier IFS
/usr/local/bin:/usr/bin:/bin:/usr/sbin:/sbin,,,,,,
Pour IFS=:
/usr/local/bin,/usr/bin,/bin,/usr/sbin,/sbin,,
```