SE - TP 6

Exercice 1 : Introduction au script shell (bash)

1. Placez les commandes qui suivent dans le fichier */tmp/hello*, et executer le script :

```
#!/bin/bash
#ceci est un commentaire

if test $# -eq 2; then
   echo "Bonjour $2 $1 "
else
   echo "Erreur de syntaxe : $0 nom prenom"
fi
```

2. Faîtes un man test pour comprendre la structure de l'instruction `test`.
3. La variable $? contient la valeur de sortie de la dernière commande. Par convention, la valeur 0 signifie qu'il n'y a pas d'erreur. Cette variable est modifiée selon que la condition de la commande `test` est vraie ou non. Essayez les commandes suivantes : `test 0 -ne 1; echo $?` puis `test 1 -ne 1; echo $?`.
4. Ecrivez un script qui affiche "fichier present" si l'argument de la commande correspond à un fichier qui existe.
5. Idem pour un répertoire.
6. Ecrivez un script qui lance un processus en tâche de fond (utilisation du &), puis affiche le PID de ce fils en utilisant la variable $!. Utilisez la commande `wait $!` pour attendre la fin de ce fils.

Exercice 2 : Ecriture de scripts shells simples

1. Ecrivez un script qui affiche le nombre d'arguments sur la ligne de commande ainsi que le premier de ces arguments (s'il y en a au moins un).
2. Que fait le script suivant :

```
#!/bin/bash
((test $1 -lt $2) && (echo '$1 < $2')) || (echo '$2 < $1')
```

3. Faîtes de même en remplacant les apostrophes par des guillemets.
4. Que fait le script suivant ?

```
#!/bin/bash
ls $1 2> /dev/null
if test $? -eq 0; then
   echo "fichier existe"
else
   echo "fichier inexistant"
fi
```

5. Réécrivez ce script en utilisant l'option -e de `test` et en supprimant l'appel à la commande `ls`.
6. Ecrivez un script *coupe* qui prend 3 arguments : un nom de fichier, et 2 entiers n et p. Ce script affiche les lignes de n à p.

Exercice 3 : Etude des scripts avancés

1. Un processus shell possède des variables auxquelles sont affectées certaines valeurs. Pour affecter une valeur à une variable on utilise la syntaxe `VARIABLE="valeur"`. La variable n'est accessible que depuis le shell où elle a été définie. Pour utiliser cette variable dans des processus fils, il faut exporter la variable : `export VARIABLE`. La liste des variables est accessible avec la commande `env`. Pour affecter le résultat d'une expression arithmétique on utilise la syntaxe suivante : `let VAR2=$VAR1 + 1`.

   (a) listez et étudiez les variables d'environnement.

   (b) Ecrivez un script permettant de calculer la somme ou le produit de 2 entiers. Le script comportera 3 paramètres : l'opérateur et les 2 entiers.

2. Examinez puis modifiez le script suivant :

```
#!/bin/bash
prog=`basename $0`
if [ -z $1 ]; then
   cat << EOF
   Usage : $prog repertoire
   EOF
   exit 0
fi
cd $1
for file in *; do
   echo -n "affiche : ${file} (o|n|q): "
   read reponse
   echo "$reponse"
done
```

   Modifiez le script pour que lorsque l'utilisateur répond "o" :

   (a) si $file est un répertoire, on affiche le contenu de ce répertoire

   (b) si $file est un fichier de type lien, on affiche ce lien

   (c) si $file est un fichier non executable, on affiche le fichier

Exercice 4 : Cryptage

1. Ecrivez un script permettant le cryptage d'un fichier dont le nom est passé en paramétre.
2. Ecrivez un script permettant le décryptage d'un fichier dont le nom est entré par l'utilisateur à la demande du programme.
3. Ecrivez un script permettant le cryptage et le décryptage d'un fichier passé en paramétre ou tapé par l'utiliateur si celui-ci n'est pas fourni en paramétre.