TP

Programmation Shell

Exercice 1 :

Ecrire un script shell permettant de créer 10 fichiers vides nommés ainsi : file1, file2, …, file10 sous un répertoire nommé **boucle** se trouvant dans votre répertoire personnel. Ce script affichera par la suite le nombre total des fichiers créés et vous demanderas une confirmation avant de supprimer le répertoire boucle.

Correction :

```
# ! /bin/bash
if [ ! –a $HOME/boucle ] ; then
    mkdir   $HOME/boucle
fi
for  var in `seq 1 10`
do
  touch   $HOME/boucle/file$var
  echo "fichiers créés"
done
nbre=`ls $HOME/boucle|wc –l`
echo "le nombre de fichier sous le répertoire boucle est $nbre"
rm –Ri  $HOME/boucle
```

Exercice 2 :

Ecrire un script shell **verifpassword** permettant de demander de saisir au clavier un mot de passe tout en vérifiant s'il concorde avec le vrai mot de passe passé en argument, à chaque fois vous introduisez un mot de passe erroné, il vous donne la main encore une fois pour réessayer. Vous aurez droit à 10 tentatives maximum. Le résultat du test sera affiché à l'utilisateur par la suite.

Correction :

```
# ! /bin/bash
for  i in `seq 1 10`
do
   echo "Saisissez un mot de passe : "
   read password
if [ ''$password'' = $1 ] ; then
  echo  "Mot de passe correct"
  exit 0
else
   echo ''mot de passé erroné''
fi
let i=i+1
done
```

Exercice 3 :

Ecrire un script shell connexion permettant d'afficher le nombre de connexion de chaque utilisateur envoyé en argument ainsi que les détails de ses 3 dernières connexions. Ce script peut accepter plusieurs paramètres et permet d'afficher la date et l'heure des 3 dernières connexions.

Indication : *Utilisez la commande* **last**

Correction:

```
# ! /bin/bash
for  user  in $*
do
nbre_cx=`last  $user| wc –l `

echo l'utilisateur $user s'est connecté au système $nbre_cx fois

echo ''Les trois dernières connexions étaient le:''

last -3 $user| cut –c 40-55

done
```

Exercice 4 :

Ecrire un script shell **age** qui permet les actions suivantes :

- Demander le nom de l'utilisateur et le saisir au clavier
- Vérifier si le fichier contenant l'âge de l'utilisateur ''username_age'' existe sous son répertoire personnel.

Si le fichier existe alors nous allons afficher son âge.

Sinon, le script demandera à l'utilisateur d'introduire son âge, crée le fichier ''username_age'' et insère l'âge de l'utilisateur dedans.

Si la valeur saisie au clavier est inférieure à 16, alors le script affichera le message « vous êtes trop jeune »

Si la valeur entrée est supérieure à 60, alors le script indiquera que « vous êtes trop vieux ».

Correction :

```
# ! /bin/bash
echo Veuillez introduire votre nom :
read username


if [ test –f $HOME/$username_age ] ; then
   age=`cat $HOME/$username_age `
   echo votre age est $age
else
  echo Veuillez introduire votre age
   read age
if [  ''$age''  -lt  16  ] ; then
  echo ''vous êtes trop jeune''
elif [  ''$age''  -gt  60 ] ; then
   echo '' vous etes trop vieux''
else
   echo $age > $HOME/$username_age
fi
fi
```

Exercice 5 :

Ecrivez un script SHELL appelé **permiss** qui reçoit plusieurs paramètres (des noms de fichiers):

Pour chaque fichier reçu, il s'agit d'effectuer les tâches suivantes :

- Si le fichier est ordinaire, vous lui ajoutez le droit de lecture pour le propriétaire, s'il ne l'a pas déjà.
- Si le fichier est un répertoire, vous lui ajoutez le droit d'exécution.
- Sinon vous réinitialisez toute la chaîne de permission relative au propriétaire à « rwx »

Si aucun nom de fichier n'a été introduit en paramètre, un message d'erreur sera redirigé vers le fichier /home/Desktop/error

Correction :

```
for i in $*
do
  if [ -z $i ] ; then
    echo "Vous n'avez pas introduit de paramèters" > /home/Desktop/error
    exit 0
  fi
  if [ -f $i ] ; then
     car=`ls –l $i|cut –c3`
     if [ $car = "r" ] ; then
        echo le fichier $i possède déjà le droit le lecture
    else
      echo ajout du droit de lecture
      chmod u+r $i
      ls –l $i
```

```
    fi
 elif [ -d $i ]; then
   echo ajout du droit d'exécution
   chmod u+x $i
   ls –l $i
else
   echo Changement des droits d'accès pour le propriétaire
   chmod u=rwx $i
fi
Fi
done
```

Exercice 7 :

Ecrivez un script appelé **Choix** permettant de vous ouvrir un menu interactif afin de donner votre avis à propos de l'euthanasie en indiquant si vous êtes pour ou contre ce phénomène.

Ce script permettra également de vérifier l'intégrité des valeurs entrées.

Correction :

```
# ! /bin/bash
echo "Êtes-vous pour ou contre  l'euthanasie ?"
select opinion in Pour Contre
 do
   case $opinion in
    "Pour"|"Contre")
                echo merci pour votre choix !
                exit 0 ;;
        "*")
        echo Votre réponse n'a aucune signification
        exit 0 ;;
    esac
   done
```

Exercice 8 :

Ecrire un script qui :

- demande à un utilisateur de saisir une commande
- exécute la commande.
- répète cette opération tant que l'utilisateur le désire, le script se ferme lorsque l'utilisateur tape **q** ou **Q**

Correction :

```
# ! /bin/bash
until [ "$rep" = "q" ] || ["$rep" = "Q" ]
do
   echo Veuillez introduire une commande :
   read cmd
   echo `
$cmd`
   echo Voulez vous continuer ?
   read rep
done
```

Exercice 8 :

Ecrire un script shell qui affiche une liste de choix de 1 à 4 comme présenté ci-dessous et demande à l'utilisateur d'introduire deux valeurs ainsi qu'un parmi les 4 choix disponibles, suite à la saisie de l'utilisateur ce script permet de calculer l'opération choisie et affiche le résultat sur le terminal.

```
1- Addition
2- Soustraction
3- Multiplication
4- Division

Entrez A :
Entrez B :
Entrez votre choix :
```

5 == 6	if [5 -eq 6]
5 != 6	if [5 -ne 6]
5 < 6	if [5 -lt 6]
5 <= 6	if [5 -le 6]
5 > 6	if [5 -gt 6]
5 >= 6	if [5 -ge 6]

Remarque :
Le choix doit être supérieur ou égal à 1 et inférieur ou égal à 4
Pas d'opération avec des valeurs négatives (inférieures à zéro).

```
echo 1-Addition
echo 2-Soustraction
echo 3-Multiplication
echo 4-Division
echo "entrer A"
read A
while [[ $A -lt 0 ]]
do
        echo "A doit être supérieur à 0"
        read A
done
echo "entrer B"
read B
while [[ $B -le 0 ]]
```

```
do
        echo "B doit être supérieur à 0"
        read B
done
echo " donner votre choix"
read op
while [[ $op -le 0 || $op -ge 5 ]]
do
        echo "choix  incorrect "
        echo "donner votre choix "
        read op
done
case  $op in
 1)     pp=+
        echo $(( $A$pp$B)) ;;
 2)     pp=-
        echo $(( $A$pp$B)) ;;
 3)     pp=*
        echo $(( $A$pp$B)) ;;
 4)     pp=/
        echo $(( $A$pp$B)) ;;
esac
```