TP3 correction

Exercice1 :
Créer un script qui effectue à la suite les opérations suivantes :

1. crée un répertoire "Essai_Script"
2. crée un fichier vide "toto.txt" dans ce répertoire"
3. crée une copie de "toto" nommée "toto2", toujours dans ce répertoire
4. stocke une liste des fichiers de votre répertoire d'accueil, toujours dans "Essai_Script"
5. affiche le texte "voila, c'est fait !"

```
cd ~
mkdir Essai_Script
touch Essai_Script/toto.txt
cp Essai_Script/toto.txt  Essai_Script/toto2.txt
ls >  Essai_Script/liste.txt
echo " voilà, c'est fait "
```

Exercice2 :
Réaliser un script qui affiche un message de bienvenue, qui lit une phrase de quatre mots au clavier et qui l'affiche à l'envers.

```
echo " bienvenue "
read a b c d
echo $d $c $b $a
```

Exercice3 :
Refaire l'exercice précédent sans utiliser la commande read.

```
echo $4 $3 $2 $1
```

Exercice4 :
Écrire un script qui compte à rebours de 5 à 1 et qui affiche ensuite "décollage".

```
for i in 5 4 3 2 1 decollage
do
        echo $i
done
```

Exercice5 :
Écrire un script qui crée 5 fichiers "fic1.txt" à "fic5.txt" dans un répertoire passé en paramètre.

```
for i in 1 2 3 4 5
do
        touch $1/fic${i}.txt
done
```

Exercice6 :
1. Écrire un script qui affiche la liste des fichiers présents dans un répertoire passé en paramètre en ajoutant un "a" au début du nom de tous les fichiers.
2. Ce script peut-il être réutilisé par un administrateur système qui voudrait renommer tous les fichiers d'un répertoire donné ? Si oui, expliquer comment.

```
for nom in `ls $1`
do
        echo ./a$nom
done
```

Exercice7 :
En utilisant le fichier "annuaire.txt" du TP précédent, écrire un script "existcopain" permettant de déterminer si le nom passé en paramètre à la commande correspond à une personne de l'annuaire.

```
if (egrep -q $1 annuaire.txt)
then
        echo “utilisateur trouvé !”
else
        echo “utilisateur non trouvé...”
fi
```

Exercice8 :

Écrire un script qui traduit en anglais un chiffre passé en paramètre. Les chiffres sont compris entre 0 et 5. Le script répondra "inconnu" en cas d’erreur.

```
case $1 in
0) echo zero;;
1) echo one ;;
2) echo two ;;
3) echo three ;;
4) echo four ;;
5) echo five ;;
*) echo Inconnu!;;
esac
```

Exercice9 :

Écrire un script qui déplace l’utilisateur dans le répertoire passé en paramètre et y crée un fichier vide uniquement si ce répertoire existe et que l’utilisateur est autorisé à se déplacer dedans. Un message d’erreur devra être affiché le cas échéant.

```
if test -d $1 -a -x $1
then
        cd $1
        touch fichier_vide
else
        echo "erreur, le repertoire n’existe pas."
fi
```

Exercice10 :

Écrire un script shell qui affiche le nom de tous les fichiers du répertoire "/usr/include" dont le nom se termine par ".h" et ayant plus de 100 lignes.

```
for nom in /usr/include/*.h
do
        nb=`wc -l $nom| cut –f1 –d" "`
        if (test $nb -gt 100)
        then
                echo $nom
        fi
done
```

Exercice11 :

Écrire un programme permettant de changer facilement l’extension d’une série de fichiers.
Exemple : "renomme htm html ~/mydir" renomme tous les fichiers ".htm" du répertoire "mydir" en ".html".

```
for nom in `ls $3/*.$1`
do
        nom_sans_ext=`basename $nom  .$1`
        mv $nom $nom_sans_ext.$2
done
```

Exercice12 :

Écrire un script qui affiche un menu donnant le choix entre 3 commandes :

1. Affichage de la date
2. Addition de deux nombres
3. Quitter

```
choix=0
while test $choix != 3
do
        echo "1 - Date "
        echo "2 - Addition"
        echo "3 - Quitter"
        read choix
        case $choix in
        1) date ;;
        2) echo "saisir deux nombres entiers"; read a b; expr $a + $b ;;
        3) echo Fin;;
        *) echo "Nombre entre 1 et 3";;
        esac
done
```

Exercice13 :

1. Écrire un script "trash" qui déplace dans un répertoire "poubelle" les fichiers dont les noms sont passés en paramètres.
2. Ajouter une option "-c" à la commande permettant de connaître la taille de la poubelle (indice : utiliser la commande du).
3. Ajouter une option "-e" permettant de vider la poubelle
4. Ajouter une option "-h" permettant d'afficher l'utilisation de la commande

```
vide_poubelle()
{
rm poubelle/*
}


display_aide()
{
echo "envoie la liste des fichiers passes en arguments dans le repertoire ./poubelle"
echo "-c donne la taille de la poubelle"
echo "-e vide la poubelle"
echo "-h affiche cette aide"
exit
}


case $1 in
"-c") du poubelle;;
"-e") vide_poubelle ;;
"-h") display_aide ;;
esac

if !(test -d poubelle)
then
        mkdir poubelle
fi
```

```
for nom_fichier
do
        if test -f $nom_fichier
        then
                mv $nom_fichier poubelle
        else
                echo "$nom_fichier inexistant"
        fi
done
```

Exercice14 :

Écrire une commande shell avec 3 arguments. Elle indiquera si les 3 chaînes sont identiques ou non par le code de retour suivant :

1. 0 si les 3 chaînes sont égales
2. 1, 2 ou 3 si la chaîne différente des autres est à la (1/2/3)ième position
3. 4 si les trois chaînes sont différentes
4. 5 si le nombre de paramètres est incorrect

```
if (test $# -eq 3)
then
        if (test $1 = $2)
        then
                if (test $1 = $3)
                then
                        exit 0
                else
                        exit 3
                fi
        fi

        if (test $3 = $2)
        then
                exit 1
        fi

        if (test $1 = $3)
        then
                exit 2
        fi
exit 4
else
        echo "nombre d arguments incorrects"
fi
exit 5
```