

Exo 1

Écrire un script qui affiche le nombre de paramètres passé sur la ligne de commande, si aucun paramètre, alors afficher « Aucun paramètres sur la ligne de commande » sinon afficher « Il y a X sur la ligne de commande » et affiche la liste des paramètres les un a coté des autres, puis les uns en dessous des autres.

```
#!/bin/sh
if [ $# -eq 0 ]
then
echo "Aucun parametre sur la ligne de commande..."
else
echo "Il y a $# parametre(s) sur la ligne de commande"
echo "liste des parametres :"
echo $*
for i in $*
do
echo $i
done
fi
```

Exo 2

Écrire un script qui dit si le premier paramètre passé en ligne de commande est un nom de fichier ou un nom de répertoire, ou de type inconnu.

```
#!/bin/sh
if [ $# -eq 0 ]
then
echo "Aucun paramètre sur la ligne de commande..."
else
if [ -f $1 ]
then
echo « Le paramètre passé en ligne de commande est un fichier... »
else if [ -d $1 ]
then
echo « Le paramètre passé en ligne de commande est un répertoire... »
else
echo « Type inconnu ! »
fi
fi
fi
```

Exo 3

Écrire un script qui n'affiche que les fichiers du répertoire courant.

Écrire un script qui affiche la liste des fichiers d'un répertoire passé en paramètre.

```
#!/bin/sh
```

```
a=$(ls)

for i in $a
do
if [ -f $i ]
then
echo "fichier : " $i
fi
done
```

Correction 2

```
#!/bin/sh

if [ $# -eq 0 ]
then
echo "Aucun paramètre sur la ligne de commande..."
else

a=$(ls $1)

for i in $a
do
if [ -f $1/"$i ]
then
echo "fichier : " $i
fi
done
cd ..
fi
```

Exo 4

Écrire un script qui n'affiche que les répertoires du répertoire courant.

Écrire un script qui affiche la liste des répertoires d'un répertoire passé en paramètre.

```
#!/bin/sh

a=$(ls)

for i in $a
do
if [ -d $i ]
then
echo "Répertoire : " $i
fi
done
```

Correction 2 :

```
#!/bin/sh

if [ $# -eq 0 ]
then
echo "Aucun paramètre sur la ligne de commande..."
```

```
else

a=$(`ls $1`)

for i in $a
do
if [ -d $1/"$i ]
then
echo "fichier : " $i
fi
done
fi
```

Exo 5

Écrire un script qui compte le nombre de fichiers et de répertoires dans le répertoire courant et affiche le résultat sous forme :

Fichiers : 12

Répertoires : 9

```
#!/bin/sh

a=$( `ls` )
nbd=0
nbf=0

for i in $a
do
if [ -f $i ]
then
nbf=$((nbf+1))
fi

if [ -d $i ]
then
nbd=$((nbd+1))
fi
done

echo "nb fichiers : " $nbf
echo "nb répertoires : " $nbd
```

Exo 6

Écrire un script qui affiche la taille d'un fichier passé en paramètre sous la forme :

Nom du fichier : test.sh ----- Taille 1562 octets

En utilisant la commande wc (word count).

```
#!/bin/sh

if [ $# -eq 0 ]
then
echo "Aucun paramètre sur la ligne de commande..."
else
wc -c $1>tmp
read a b <tmp

echo "Nom du fichier : $1 ----- Taille " $a " octets."

rm tmp
fi
```

Exo 7**Question Unix**

Écrire un script qui affiche le menu suivant en continu :

**1 – Windows?
2 – BeOs?
3 – Linux?
4 – Unix?
9- Quitter**

Réponse ?

Si vous répondez 1 alors le programme affiche « Dommage! », 2 il affiche « Peut mieux faire! », 3 « Pas mal! », 4 « Super! », Q pour quitter le menu et revenir a la ligne de cmd.

```
#!/bin/sh

while :
do
echo "1 - Windows?"
echo "2 - BeOs?"
echo "3 - Linux?"
echo "4 - Unix?"
Echo "Q - Quitter"

read -p "Réponse ?" -n 1 rep

case $rep in
1) echo "Dommage!";;
2) echo "Peut mieux faire!";;
3) echo "Pas mal!";;
4) echo "Super!";;
9) exit 0;;
*) echo "Sans opinion apparemment!!!";;
esac
done
```

Exo 8

Soit un fichier contenant les valeurs suivantes dans cet ordre :

9 10
20 30
10 20

Écrire un script qui permet de faire la somme des 2 valeurs figurant sur une ligne et affiche à l'écran le résultat sous la forme $X+Y=Z$.

```
#!/bin/sh

a=$(cat bidon.txt)

nb=0
for i in $a
do

if [ $nb -eq 1 ]
then
v2=$i
somme=$(( $v1+$v2 ))
echo $v1 '+' $v2 ' = '$somme
let nb=2
fi

if [ $nb -eq 0 ]
then
v1=$i
let nb++
fi

if [ $nb -eq 2 ]
then
let nb=0
fi
done
```

Exo 9

Idem à l'exercice 8 en passant en paramètre le fichier contenant les valeurs et le type d'opération à effectuer (mul, div, add, sou)

Exemple :

calculx.sh nombre.txt mul
calculx.sh nombre.txt sou

```
#!/bin/sh

if [ $# -lt 2 ]
then
echo "Aucun paramètre sur la ligne de commande..."
else

a=$(cat $1)

nb=0
for i in $a
do

if [ $nb -eq 1 ]
then
v2=$i

case $2 in
add)calc=$(( $v1+$v2 ));;
sou)calc=$(( $v1-$v2 ));;
mul)calc=$(( $v1*$v2 ));;
div)calc=$(( $v1/$v2 ));;
esac

echo $v1 '+' $v2 ' = '$calc
let nb=2
fi

if [ $nb -eq 0 ]
then
v1=$i
let nb++
fi

if [ $nb -eq 2 ]
then
let nb=0
fi
done

fi
```

Exo 10

Réalisez un script qui en fonction de l'heure courante affiche « Bonjour » entre 0h et 12h, « Bon après midi » entre 12h et 17h et « Bonne soirée » entre 17h et 0h.

```
#!/bin/sh
heure=`date | awk '{print $4}' | awk -F: '{print $1}'`
if [ $heure -lt 12 ]
then
echo « Bonjour! »
```

```

elif [ $heure -lt 17 ]
then
echo « Bon après midi! »
else
echo « Bonsoir! »
fi

```

Correction 2 :

```

#!/bin/sh

date>tmp
read a b c d e<tmp
rm tmp

deb=`expr index "$d" :`

let deb=deb-1

heure=${d:0:$deb}

if [ $heure -lt 12 ]
then
echo "Bonjour!"
elif [ $heure -lt 17 ]
then
echo "Bon apres midi!"
else
echo "Bonsoir!"
fi

```

Exo 11

Écrire un script qui génère un mot de passe dont on passera la longueur en paramètre.

```

#!/bin/sh

if [ $# -ge 1 ]
then
tab="abcdefghijklmnopqrstuvwxy0123456789ABCDEFGHIJKLMNopqrstuvwxyz"
nbparam=0

while [ $nbparam -lt $1 ]
do

valeur=$(( $RANDOM % 62 ))
mdp=$mdp${tab:$valeur:1}
let nbparam=nbparam+1

done

echo "Mot de passe : " $mdp

else

echo "Syntaxe : ./mdp.sh [nbLettres]"

```

```
fi
```

Exo 12

Écrire un script qui donne les valeurs de $Y=10X+3X^2$ pour de valeurs allant de x à x' avec un incrément de z .

Les valeurs x , x' et z seront passé en paramètres sur la ligne de commande.

```
#!/bin/sh

if [ $# -eq 3 ]
then
x=$1
xp=$2
z=$3
inc=$1

while [ $inc -le $xp ]
do
let b2=3*inc*inc
let resultat=10*inc+b2
echo "10X+3X² avec X=$inc donne $resultat"
let inc=inc+z
done

else
echo "Syntaxe : $0 [x] [x'] [pas]"
fi
```

Exo 13

Écrire un script Shell qui permet de proposer à l'utilisateur de deviner un nombre entre 0 et 1000 et ceci en un maximum de 10 coups. A chaque valeur proposer par l'utilisateur, l'application annoncera

« La valeur est plus grande ! » si $val_proposer < val_a_chercher$

« La valeur est plus petite ! » si $val_proposer > val_a_chercher$

« Vous avez trouvé la valeur correcte ! » si $val_proposer = val_a_chercher$ et on arrête le script

« Vous avez perdu ! » si $nb_coup > 10$

```
#!/bin/sh

clear

echo "Cherche la valeur Version 0.1"
echo

achercher=0
Bornesup=1000
nbcop=10
num=1
```

```
ok=0
valeur=0

nombre(){
  achercher=$(( $RANDOM % $Bornesup ))
}

demande_valeur(){

valeur=1001
while [ $valeur -lt 0 ] || [ $valeur -gt 1000 ]
do
printf "Cp-$num : Indiquez une valeur entre 0 et 1000 : "
read valeur
done

}

verifier_valeur(){

if [ $valeur -lt $achercher ]
then
ok=0
echo "La valeur recherché est plus grande..."
echo
fi

if [ $valeur -gt $achercher ]
then
ok=0
echo "La valeur recherché est plus petite..."
echo
fi

if [ $valeur -eq $achercher ]
then
ok=1
echo "Vous avez trouvé la valeur correcte..."
fi
}

nombre

while [ $num -le $nbcp ] && [ $ok -ne 1 ]
do
demande_valeur
verifier_valeur
let num=num+1
done

if [ $num -gt $nbcp ]
then
echo "Vous avez perdu la partie, la valeur a trouver était $achercher, désolé!"
fi

exit 0
```

Exo 14

Vous disposez d'un répertoire contenant un ensemble de scripts dont les noms ne comportent pas d'extension (script1, script2, script3, script4...)

Tous ces scripts sont correctement écrits : ils ont donc comme première ligne l'en-tête standard (nommée "Shebang") :

```
#!/bin/interpreteur
```

Dans laquelle l'interpréteur peut être *ksh, bash, perl, php, python, etc.*

Le but du script est de rajouter à chaque fichier une extension significative de l'interpréteur associé. Le répertoire d'action est donné en premier paramètre.

Le script doit :

- **vérifier l'existence du premier argument**
- **vérifier que le premier argument désigne un répertoire qui existe**
- **trouver un moyen de récupérer de façon sûre la première ligne contenant le Shebang**
- **extraire de sa valeur le nom de l'interpréteur**
- **changer le nom du fichier en y ajoutant l'extension correspondante.**

- ```
#!/bin/bash

usage : add_extensions.sh <dirname>

if (($# == 0))
then
 echo "pas assez d'arguments"
 exit
fi

if [! -d $1]
then
 echo "l'argument ne désigne pas un répertoire"
 exit
fi

for f in $1/*
do
 FIRSTLINE=`grep "#\!.*" $f`
 echo $FIRSTLINE
 mv $f $f.${FIRSTLINE##*\}/}
done
```