DECOUVERTE DU BASH

I/ Variables et Arguments

1. ø Créer un script bonjour.sh qui affiche "Bonjour *nom*"

```
#!/bin/bash
echo "Bonjour $1"
```

2. ø Créer un script whoami.sh qui affiche :

- l'utilisateur courant
- le répertoire courant
- le nombre d'arguments passés au script

```
#!/bin/bash
echo "$(whoami)"
echo "$(pwd)"
echo "$#"
```

3. ø Créer un script somme.sh qui prend 2 nombres en argument et affiche leur somme.

```
#!/bin/bash
echo $(($1+$2))
```

4. ø Créer un script affiche args.sh qui affiche tous les arguments passés au script.

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

II/ Conditions

1. ø Écrire un script testfichier.sh qui teste si un fichier existe :

- s'il existe → afficher « Fichier trouvé »
- sinon → afficher « Fichier absent »

```
#!/bin/bash
if [ -f NOM_FICHIER ]
then
        echo "Fichier trouvé"
```

```
else
    echo "Fichier absent"
fi
```

2. ø Modifier le script testfichier.sh pour que le nom du fichier soit passé en argument

```
#!/bin/bash
if [ -f $1 ]
then
    echo "Fichier trouvé"
else
    echo "Fichier absent"
fi
```

3. ø Écrire un script pairimpair.sh qui prend un nombre en argument et affiche s'il est pair ou impair.

```
#!/bin/bash
if [ $(($1 % 2)) -eq 0 ]
then
    echo "$1 est pair"
else
    echo "$1 est impair"
fi
```

4. ø Écrire un script connecte.sh qui prend un nom d'utilisateur en argument et affiche s'il est connecté.

```
#!/bin/bash
if who | grep -wq $1
then
    echo "$1 est connecté"
else
    echo "$1 n'est pas connecté"
fi
```

5. ø Écrire un script menu.sh qui affiche un menu avec 3 choix :

- 1 → afficher la date du jour
- 2 → afficher les utilisateurs connectés
- 3 → quitter

```
#!/bin/bash
echo "1 -> afficher la date du jour"
echo "2 -> afficher les utilisateurs connectés"
echo "3 -> quitter"
```

```
echo "Votre choix : "
read choix
if [ $choix -eq 1 ]
then
    echo "$(date)"
elif [ $choix -eq 2 ]
then
    echo "$(users)"
else
    echo "bye"
fi
```

III/ Boucles

1. ø Écrire un script listeargs.sh qui affiche tous les arguments passés au script un par un à l'aide d'une boucle for.

```
#!/bin/bash
for i in $@
do
    echo $i
done
```

2. ø Écrire un script table.sh qui demande un nombre à l'utilisateur puis affiche sa table de multiplication de 1 à 10. Il faudra vérifier que l'utilisateur ait entré un nombre.

```
#!/bin/bash
echo "Entrez un nombre : "
read nb
if [[ $nb =~ ^[0-9]+$ ]]
then
    echo "Table de multiplication de $nb"
    for i in {1..10}
    do
        echo "$(($nb * $i))"
    done
fi
```

3. ø Écrire un script pair.sh qui affiche les nombres pairs entre 1 et 20 avec une boucle for.

```
#!/bin/bash
for i in {1..20}
do
    if [ $(($i % 2)) -eq 0 ]
    then
        echo $i
```

```
    fi
done
```

4. ø Reprendre et modifier le script somme.sh. Cette nouvelle version devra demander à l'utili□sateur de taper des nombres un par un. La saisie s'arrête quand l'utilisateur entre 0. Le script affiche alors la somme de tous les nombres saisis. Il faudra vérifier que la valeur entrée est bien un nombre, si ce n'est pas le cas, le programme devra s'arrêter en indiquant une erreur

```
#!/bin/bash
echo "Entrez un nombre : "
read nb

somme=0
if ! [[ $nb =~ ^[0-9]+$ ]]
then
    exit 1
fi
while [ $nb -ne 0 ]
do
    somme=$(($somme + $nb))
    echo "Entrez un nombre : "
    read nb

    somme=0
    if ! [[ $nb =~ ^[0-9]+$ ]]
    then
        exit 1
    fi
done

echo "La somme totale est $somme"
```

5. ø Écrire un script taille.sh qui affiche la taille (en octets) de chaque fichier du répertoire courant. La commande stat permet d'obtenir des informations sur un fichier

```
#!/bin/bash
for fichier in $(ls)
do
    echo "$fichier a une taille de $(stat -c%s $fichier)"
done
```

IV/ Scripts avancés

1. ø Écrire un script sauvegarde.sh qui :

- prend en argument un répertoire
- crée une archive nommée **sauvegarde-AAAAMMJJ.tar.gz**, contenant le répertoire donné

```
#!/bin/bash
jour=$(date +"%d")
mois=$(date +"%m")
annee=$(date +"%Y")
tar -cvzf "sauvegarde-$annee$mois$jour.tar.gz" $1
```

2. ø Écrire un script recherche.sh qui :

- prend en argument un mot et un fichier
- affiche combien de fois le mot apparaît dans le fichier

```
#!/bin/bash
if ! [ -f $2 ]
then
    echo "Le fichier $2 n'existe pas"
    exit 1
fi
echo "$(grep -o $1 $2 | wc -l)"
```

3. ø Écrire un script nettoie.sh qui supprime tous les fichiers temporaires *.tmp d'un dossier donné en argument.

```
#!/bin/bash
for fichier in $(ls)
do
    if [[ $fichier = *.tmp ]]
    then
        echo "Suppression de $fichier"
        rm $fichier
    fi
done
```

4. ø Écrire un script rapport.sh qui prend un dossier en argument et affiche :

- le nombre total de fichiers
- le nombre total de répertoires
- la taille totale du dossier

```
#!/bin/bash
if ! [ -d $1 ]
then
    echo "Le dossier $1 n'existe pas"
    exit 1
fi

echo "Le dossier $1 contient $(find $1 -type f | wc -l) fichier(s)"
```

```
  echo "Le dossier $1 contient $(ls -d $1 | wc -l) dossier(s)"
  echo "Le dossier $1 a une taille de $(stat -c%s $1)"
```