Module 2
420-127-LG – Script 2 série 2
Solution
Quelques unes des solutions proviennent de quelques élèves de la classe.
Si vous reconnaissez votre code, je vous en remercie d'avance.

1. Faites un script appelé gestion qui affichera un menu simple comme celui-ci:
    1. Créer un groupe
    2. Renommer un groupe
    3. Détruire un groupe
    4. Créer un usager

    Le script doit ensuite saisir le choix de l'usager et faire afficher le choix.

```
#!/bin/bash

clear
echo "1. Créer un groupe "
echo "2. Renommer un groupe"
echo "3. Détruire un groupe"
echo "4. Créer un usager"
echo "----------------------"
read -p "Entrez votre numéro: " Choix

case $Choix in
    1)
      echo "Vous avez choisi le No 1"
      ;;
    2)
      echo "Vous avez choisi le No 2"
      ;;
    3)
      echo "Vous avez choisi le No 3"
      ;;
    4)
      echo "Vous avez choisi le No 4"
      ;;
    *)
      echo "Ce choix est invalide"
      ;;
esac

exit 0
```

2. Créez un script appelé "nombres" qui affiche les nombres entiers d'une borne à une autre. On appellera votre script ainsi:

   ```
   nombres 1 15
   ```

   Votre script devra dans ce cas afficher les nombres de 1 à 15. Toutefois, on pourra également l'appeler ainsi:

   ```
   nombres 15 1
   ```

   Dans ce cas, votre script devra afficher les nombres de 15 à 1, à rebours. Utilisez une boucle while pour afficher en ordre croissant et une boucle until pour afficher en ordre décroissant.

   Prenez pour acquis que l'usager appelle toujours votre script correctement, inutile de valider le nombre de paramètres (à moins que vous ayez envie de le faire, dans ce cas, laissez-vous aller).

```
#!/bin/bash

clear
if [ $# -ne 2 ]; then  # vérifie qu'il n'y a que deux paramètres
  echo "La commande doit comporter 2 nombres en paramètre."
  echo "Exemple de commande :  nombres 1  15"
  echo
  exit 1
fi

Nombre1=$1
Nombre2=$2

if [ $Nombre2 -ge $Nombre1 ]; then
  while [ $Nombre1 -le $Nombre2 ] ; do
     echo $Nombre1
     (( Nombre1++ ))
  done
else
  while [ $Nombre1 -ge $Nombre2 ] ; do
     echo $Nombre1
     (( Nombre1-- ))
  done
fi

exit 0
```

3. Créez un script appelé "verify" qui accepte en paramètre des chemins et noms de fichiers (ou de répertoires) complets (autant que désiré) et qui nous dira si ces fichiers (ou répertoires) existent sur le système.

   Par exemple, l'usager pourrait entrer:

   ```
   verify /usr/sbin/groupadd /usr/bin/georges /etc /dev/null
   ```

   Le script lui dirait alors:

   ```
   Le fichier /usr/sbin/groupadd existe
   /usr/bin/georges n'existe pas
   Le répertoire /etc existe
   /dev/null existe mais n'est ni un fichier ni un repertoire
   ```

   Évidemment l'usager pourra entrer autant de noms de fichiers (ou de répertoires) qu'il voudra. Vous aurez besoin d'imbriquer des if. Rappelez-vous des switchs -d, -f et -e pour vérifier les existances de fichiers ou de répertoires.

```
#!/bin/bash

for Param in $*;  do
  if [ -e $Param ]; then    # teste si l'entrée existe
        if [ -d $Param ]; then # teste si c'est un répertoire
           echo -e "Le répertoire $Param existe"
           echo
        else
          if [ -f $Param ]; then  # teste si c'est un fichier
              echo -e "Le fichier $Param existe"
              echo
          else
            echo -e "$Param existe mais n'est ni un fichier ni un repertoire"
            echo
          fi
      fi
   else
        echo -e "$Param n'existe pas"
        echo
   fi

done

exit 0
```

4. Faire un **script** qui permet de compter le nombre de fichier que vous pouvez exécuter dans un répertoire donné.

```
#!/bin/bash
        cpt=0
        for i in `ls $1`
          do
            if [ -x $i -a  -f $i ]; then
                  (( cpt++ ))
            fi
         done
        echo $cpt
exit 0
```

5. Réaliser une nouvelle commande que vous nommerez "**list**" et qui en fera un peu plus que la commande « ls ». Voici ce que votre script doit réaliser :
    - faire afficher la liste du contenu du répertoire à l'écran et à la fin,
    - faire afficher le nombre de répertoire total,
    - faire afficher le nombre de fichier total.

```
#!/bin/bash

NbreRepertoire=0
NbreFichier=0
NomDossier=""  #Sert à ajouter le répertoire à partir duquel on veut les
statistiques

if [ $# -eq 1 -a -d $1 ]; then
  Commande=`ls $1`
  NomDossier="$1/"
else
  Commande=`ls`

fi

for Entree in $Commande;  do
  if [ -d $NomDossier$Entree ]; then
    (( NbreRepertoire++ ))
  else
    if [ -f $NomDossier$Entree ]; then
      ((NbreFichier++))
    fi
  fi
done
ls -al $NomDossier
echo "-----------------------------------------------"
echo "Nombre de repertoire: $NbreRepertoire"
echo "Nombre de fichier: $NbreFichier"

exit 0
```

6. Créer un script nommé « creergroupe » qui permettra de créer un groupe dans le système avec le nom du groupe qui lui sera passé en paramètre. Voici des exemples :

```
creergroupe  andromede
creergroupe  voielactee
```

Solution sans vérification préalable :

```
#!/bin/bash

groupadd $1

exit 0
```

Solution avec vérification :

```
#!/bin/bash

if [ $# -eq 1 ]; then
  if [ $(getent group $1) ]; then
    echo "Le groupe $1 existe déjà"
  else
    groupadd $1
    echo "Le groupe $1 a été ajouté dans le système"
  fi
fi
exit 0
```

getent group nom_groupe vérifie si le groupe nommé « nom_groupe » existe dans le système.
Si le groupe existe, la commande retourne la ligne qui définie le groupe. Dans le cas contraire, la commande retourne du vide.

Vous pouvez aussi vérifier de la même façon si un usager existe dans le système avec la commande suivante : getent passwd nom_usager