TP 3 : Redirections et Scripts Shell

1 Les redirections

1. Concaténer les deux fichiers `/etc/passwd` et `/etc/group` dans le fichier `/tmp/comptes`

   Réponse :

   ```
   $ cat /etc/passwd /etc/group > /tmp/comptes
   ```

2. Afficher la liste de tous les processus actifs et sauvegarder le résultat de la commande dans le fichier nommé `psliste`.

   Réponse :

   ```
   $ ps aux > psliste
   ```

3. Chercher tous les fichiers de nom contenant "`passwd`" depuis le répertoire /. Sauvegarder le résultat de votre commande (sans les messages d'erreurs) dans le fichier `listepasswd`. Tous les messages d'erreur doivent être redirigés dans le fichier spécial `/dev/null`.

   Réponse :

   ```
   $ ls -R / 2>/dev/null | grep passwd > listepasswd
   ```

4. Trier le fichier `/etc/passwd` par ordre alphabétique et enregistrer le résultat dans le fichier `TriPasswd`.

   Réponse :

   ```
   $ sort /etc/passwd > TriPasswd
   ```

5. Trier le fichier `/etc/passwd` suivant l'UID et sauvegarder le résultat à la suite du fichier `TriPasswd` de la question précédente.

   Réponse :

   ```
   $ sort -t: -k3 -n /etc/passwd >> TriPasswd
   ```

6. Remplacer la séquence de commandes suivantes par une seule commande :

   ```
   cd /etc
   ls > /tmp/liste.txt
   cat /tmp/liste.txt
   wc -l < /tmp/liste.txt
   ```

   Réponse : Ces commandes permettent de compter le nombre de fichiers et répertoires du répertoire `/etc`.

   ```
   $ ls /etc | wc -l
   ```

7. Afficher la liste de tous les processus de l'utilisateur `root`

   Réponse : Il faut chercher, dans le résultat de la commande `ps aux`, les lignes qui commencent par la chaîne de caractères `root`

   ```
   $ ps aux | grep ^root
   ```

1. Ecriver le script suivant nommé `Parametres` :

```
#!/bin/bash

# Parametres : Paramètre de position
echo "Nom du script en cours d'exécution : $0"
echo "Nombre de paramètres transmis à un script : $#"
echo "Numéro du processus du script en cours d'exécution $0 est : $$"

echo "Paramètre 1 du script $0 : $1"
echo "Paramètre 2 du script $0 : $2"
echo "Paramètre 3 du script $0 : $3"
echo "Liste de tous les paramètres du script $0 : $*"
```

Une fois le script rendu exécutable, lancez-le de différentes manières :

- Sans paramètre.

  Réponse :

```
$ chmod u+x Parametres.sh
$ ./Parametres.sh
Nom du script en cours d'exécution : Parametres.sh
Nombre de paramètres transmis à un script : 0
Numéro du processus du script en cours d'exécution Parametres.sh est : 1236
Paramètre 1 du script Parametres.sh :
Paramètre 2 du script Parametres.sh :
Paramètre 3 du script Parametres.sh :
Liste de tous les paramètres du script Parametres.sh :
```

- Avec un seul paramètre.

  Réponse :

```
$ ./Parametres.sh exemple
Nom du script en cours d'exécution : Parametres.sh
Nombre de paramètres transmis à un script : 1
Numéro du processus du script en cours d'exécution Parametres.sh est : 1240
Paramètre 1 du script Parametres.sh : exemple
Paramètre 2 du script Parametres.sh :
Paramètre 3 du script Parametres.sh :
Liste de tous les paramètres du script Parametres.sh : exemple
```

- Avec deux paramètres séparés par un espace.

  Réponse :

```
$ ./Parametres.sh exemple Hello
Nom du script en cours d'exécution : Parametres.sh
Nombre de paramètres transmis à un script : 2
Numéro du processus du script en cours d'exécution Parametres.sh est : 1242
Paramètre 1 du script Parametres.sh : exemple
Paramètre 2 du script Parametres.sh : Hello
Paramètre 3 du script Parametres.sh :
Liste de tous les paramètres du script Parametres.sh : exemple Hello
```

- Avec trois paramètres séparés par des espaces.

  Réponse :

```
$ ./Parametres.sh Bonjour tout le monde
Nom du script en cours d'exécution : Parametres.sh
```

```
Nombre de paramètres transmis à un script : 3
Numéro du processus du script en cours d'exécution Parametres.sh est : 1243
Paramètre 1 du script Parametres.sh : Bonjour
Paramètre 2 du script Parametres.sh : tout
Paramètre 3 du script Parametres.sh : le monde
Liste de tous les paramètres du script Parametres.sh : Bonjour tout le monde
```

2. Créer un script nommé **AfficheRep** qui affiche les caractéristiques et le contenu d'un répertoire dont le nom est donné en paramètre du script.

Réponse :

```bash
#!/bin/bash
if [ $# -eq 0 ]
then
  echo "Il faut au moins un paramètre"
  exit 1
elif [ -d $1 ]
then
  echo "Les caractéristiques du répertoire $1 sont : `ls -ld $1`"
  echo "Le contenu du répertoire $1 est : `ls -l $1`"
else
  echo "$1 n'est pas un répertoire"
  exit 2
fi
```

3. Créer un script **vi2** qui prend en argument un nom de fichier. Le script réalise une sauvegarde du fichier dans le répertoire **/tmp** avant de lancer l'éditeur de textes **vi** pour afficher et/ou modifier son contenu. Le fichier de sauvegarde aura comme nom :
nom_parametre_1.numero_processus_script_en_cours.

Réponse :

```bash
#!/bin/bash
if [ $# -eq 0 ]
then
  echo "Il faut au moins un paramètre"
  exit 1
elif [ -f $1 ]
then
  cp $1 /tmp/$1.$$    # Copie de sauvegarde dans /tmp
  vi $1       # Ouverture du fichier avec vi
else
  echo "$1 n'est pas fichier"
  exit 2
fi
```

4. Lancer le script suivant que vous nommerez **CmdeSet** :

```bash
#!/bin/bash

# Utilisation de la commande set

# Première façon : set chaine_de_caractères
set Nom Prenom
echo $1
echo $2

# set `commande`
```

```
set `ls -ld /etc`
echo "Résultat : $*"
echo "Paramètre 1 de resultat : $1"
echo "Paramètre 2 de resultat : $2"
echo "Paramètre 3 de resultat : $3"
echo "Paramètre 4 de resultat : $4"
echo "Paramètre 5 de resultat : $5"
echo "Paramètre 6 de resultat : $6"
echo "Paramètre 7 de resultat : $7"
echo "Paramètre 8 de resultat : $8"
Horaire=$8
# Rôle de la variable IFS
set $Horaire
echo "Paramètre 1 de $Horaire est : $1"
echo "Paramètre 2 de $Horaire est : $2"
# On modifie le séparateur de champs
IFS=:
echo "Le séparateur de paramètre est : $IFS "
set $Horaire
echo "Paramètre 1 : $1"
echo "Paramètre 2 : $2"
```

Que permet la variable IFS ?

Réponse : *La variable d'environnement* `IFS` *définit le « séparateur par défaut », c'est à dire le(s) caractère(s) utilisé(s) pour délimiter les mots dans une chaîne de caractères.*

5. Créer un script nommé **InfoSys** qui affiche les informations suivantes du système :

   - Architecture matérielle.
   - Nom de la machine
   - Nom du système d'exploitation
   - Version du noyau.

   La commande **uname** est invoqué une seule fois dans le script. Chaque information est précédée de son libellé ci-dessus.

   Réponse :

```
#!/bin/bash

set `uname -a`

echo "Architecture matérielle : ${15}"
echo "Nom de la machine : $2"
echo "Nom du système d'exploitation : $1"
echo "Version du noyau : $3"
```

6. Créer un script nommé **InfoDate** qui affiche la date sous la forme : **nom_du_jour-mois-année** et l'heure sous la forme suivante : il est xx heures et yy minutes et zz secondes.

   Réponse :

```
#!/bin/bash

set `date`
echo "$1-$2-$6
IFS=:
set $4
echo "Il est $1 heures et $2 minutes et $3 secondes"
```