Exercices TP

Recopiez le contenu du répertoire `/Infos/lmd/2011/licence/ue/li219-2012fev/TP4` qui contient tous les fichiers nécessaires pour la réalisation de ce TP et en particulier le fichier `compte_rendu_TP4.txt`que vous devez compléter au fur et à mesure et soumettre à la fin de la séance.

Pour tester vos scripts vous pourrez avoir besoin d'ouvrir deux fenêtres "Terminal", l'une pour lancer votre script et l'autre pour lui envoyer un signal avec la commande **`kill`**.

Exercice 31 – Terminaison d'un processus

Question 1

Exécutez le script suivant. Comment pouvez-vous arrêter l'exécution ? Attention, votre solution doit provoquer la **terminaison** du processus en train de s'exécuter.

```
#! /bin/bash
# boucleinfinie.sh

echo Je suis le processus $$
trap "echo $$ ignore le signale SIGINT" SIGINT

i=0
while [ true ]
do
 echo $i
 i=`expr $i + 1`
 sleep 1
done
```

Question 2

Exécutez le script suivant, envoyez au processus le signal `SIGKILL`. Que constatez-vous ?

```
#! /bin/bash
# boucleinfinie2.sh

echo Je suis le processus $$
trap "echo $$ ignore le signale SIGKILL" SIGKILL

while [ true ]
do
 sleep 60
done
```

Question 3

Ecrivez un script `boucleinfinie3.sh` pour que, lorsque le processus exécutat le script reçoit un `ctrl-C`, il affiche un message et arrête immédiatement son exécution.

Question 4

Écrivez un script `boucleinfinie4.sh` qui, lors de la réception d'un signal `SIGINT` (2), `SIGQUIT` (3) ou `SIGTERM` (15), affiche le nom du signal reçu et se termine. Ce script doit faire une boucle infinie tant qu'aucun signal n'est reçu.

Exercice 32 – Signaux

Question 1
Écrivez un script `saisie.sh` qui demande la saisie de trois entiers. Le premier entier doit être comprise entre 1 et 31, le deuxième entre 1 et 12 et le troisième entre 1960 et 1980. Si un entier n'est pas dans l'intervalle attendu, la saisie de la valeur est répétée tant que les contraintes ne sont pas vérifiées.

Question 2
Écrivez un nouveau script `saisie1.sh` qui a le même comportement que le script `saisie.sh` mais qui assure en plus qu'un processus qui l'exécute ignore le signal `SIGINT` (2) lors de l'exécution de la deuxième boucle de saisie. Le processus a le comportement par défaut avant ou après cette boucle.

Exercice 33 – Valeur de retour d'un processus

Dans cet exercice, vous devez arrêter l'exécution des processus si le nombre de paramètres n'est pas correct.

Question 1
Écrivez un script `petit_fils.sh` qui prend un paramètre et qui affiche le texte `Je suis le processus` suivi de la valeur de ce paramètre. Voici un exemple d'exécution :

```
Prompt% ./petit_fils.sh 7
Je suis le processus 7
```

Question 2
Le script `alea.sh` (qui vous est fourni) affiche une valeur tirée aléatoirement entre `0` et `9`.

Écrivez un script `fils.sh` qui prend un paramètre (son identité). Le script récupère une valeur tirée aléatoirement (en utilisant le script `alea.sh`) et crée un nombre égal à cette valeur de processus `petit_fils.sh`. Avant de créer ses petits-fils, il affiche son identité et le nombre de petits-fils à créer.

Chaque processus `petit_fils.sh` reçoit un paramètre composé de son ordre de création concaténé à l'identité de son père. Voici un exemple d'exécution :

```
Prompt% ./fils.sh 3
Je suis le fils 3 et je dois creer 4 processus
Je suis le processus 3.1
Je suis le processus 3.2
Je suis le processus 3.3
Je suis le processus 3.4
```

Question 3
Écrivez un processus `pere.sh` qui prend un paramètre et qui crée un nombre de processus `fils.sh` égal à la valeur de ce paramètre. Voici un exemple d'exécution :

```
Prompt% ./pere.sh 3
Je suis le fils 1 et je dois creer 2 processus
Je suis le processus 1.1
Je suis le processus 1.2
Je suis le fils 2 et je dois creer 4 processus
Je suis le processus 2.1
Je suis le processus 2.2
Je suis le processus 2.3
Je suis le processus 2.4
Je suis le fils 3 et je dois creer 1 processus
Je suis le processus 3.1
```

Question 4

Ecrivez une nouvelle version des scripts pour que le script `pere.sh` affiche le nombre total de processus créés (le nombre de processus `fils.sh` + le nombre de processus `petit_fils.sh`). Votre solution **ne doit pas utiliser** de fichier. Voici une exécution :

```
Prompt% ./pere.sh 3
Je suis le fils 1 et je dois creer 2 processus
Je suis le processus 1.1
Je suis le processus 1.2
Je suis le fils 2 et je dois creer 4 processus
Je suis le processus 2.1
Je suis le processus 2.2
Je suis le processus 2.3
Je suis le processus 2.4
Je suis le fils 3 et je dois creer 1 processus
Je suis le processus 3.1
10 processus ont ete crees
```

Exercice 34 – Valeur retournée et/ou valeur affichée ?

Soit le script `repertoire.sh` suivant :

```
#! /bin/bash
# repertoire.sh

for rep in "$@"; do
  if [ -d "$rep" ]; then
    echo $rep est un repertoire
  else
    echo $rep n est pas un repertoire
  fi
done
```

Question 1

Écrivez un script `test.sh` qui prend en paramètre une liste de noms, qui appelle **une seule fois** le script `repertoire.sh` et qui affiche, en fonction des cas :

- `Tous les parametres sont des repertoires`
- `1 parametre ne correspond pas a un repertoire`
- `x parametres ne correspondent pas a un repertoire` (où `x` correspond au nombre de paramètres qui ne sont pas des répertoires)

Le script `test.sh` s'occupe exclusivement de l'affichage et le script `repertoire.sh` s'occupe du comptage. Vous modifierez donc aussi le script `repertoire.sh` qui ne devra contenir aucune instruction d'affichage.