TD n°2 : Programmation shell

Objectif : Pratiquer les bases de l'écriture de scripts shell

Exercice 1 – Exemple de programme shell

1) Que fait le programme shell suivant, dont le nom est `mystere` ?

```
#!/bin/sh
if [ $# -ne 1 ]; then
    echo "Fournir un nom en parametre"
    exit 1
fi
if  [ -d "$1" ]; then
    exit 0
else
    "$0" `dirname "$1"`
    mkdir "$1"
fi
```

2) Proposez des exemples d'appels du programme `mystere`.

Exercice 2 – La boucle while

Écrire un programme shell qui affiche les arguments du programme, dans l'ordre d'apparition (1er argument en premier). Si le programme n'a aucun argument, afficher « *sans argument* ».

Indication  **:** Utilisez la commande `shift`.

Exercice 3 – La commande read

Écrire un programme shell qui affiche ligne par ligne le contenu d'un fichier dont le nom est fourni en paramètre.

Exercice 4 – La boucle for

Écrire un programme shell qui affiche tous les sous-répertoires du répertoire courant, *en utilisant une boucle*.

Exercice 5 – Opérateurs sur les chaînes

Que fait le programme shell suivant (remplacez les '. . .' par le texte adéquat) ?

```
#!/bin/sh
w=`who | grep $1`
if [ -z "$w" ]; then echo "$1 n'est pas ..."; fi
```

Exercice 6 – Les conditionnelles imbriquées

1) Écrire un programme shell qui accepte 2 paramètres. Le premier paramètre est `+r`, `-r`, `+w` ou `-w`, et le deuxième paramètre spécifie une extension de nom de fichiers. En fonction de la valeur du premier paramètre, le programme modifiera les droits du groupe de tous les fichiers du répertoire courant dont l'extension est égale au deuxième paramètre. Pour contrôle, avant chaque modification des droits sur un fichier, le programme affichera le nom du fichier.

Exemple d'utilisation (le script s'appelle `droitsfichiers`) :

```
droitsfichiers +r .html
```

2) Proposez une nouvelle version de ce programme capable d'accepter trois paramètres. Les trois paramètres spécifient alors :

1. le répertoire dans lequel sont contenus les fichiers dont les droits seront modifiés,
2. les modifications des droits pour le groupe,
3. l'extension des fichiers concernés.

Exemples d'utilisation (le script s'appelle `droitsfichiers`) :

```
droitsfichiers .. -w .dat
droitsfichiers perso -r .txt
```

Exercice 7 – L'instruction case

Écrire un programme shell qui efface un fichier après avoir demandé confirmation à l'utilisateur. Le programme doit recevoir en paramètre le ou les noms du/des fichier(s) à effacer. Pour chaque fichier, il demande alors à l'utilisateur : « *Voulez-vous réellement effacer le fichier xxx ?* ». Si la réponse est *oui*, le programme affiche « *suppression confirmée »*, et efface le fichier. Si la réponse est *non* le programme affiche « *suppression abandonnée »*. Dans les autres cas, le programme affiche « *réponse invalide »* et repose la question à l'utilisateur.

Exercice 8 – La commande basename

On voudrait écrire un script permettant de facilement changer de façon systématique l'extension d'une série de fichiers. On souhaite par exemple renommer tous les fichiers `.htm` du répertoire courant en `.html`

La syntaxe de notre script `rename` serait alors la suivante : `rename .htm .html`

On vérifiera que le nombre d'arguments reçu est correct, que toutes les opérations sont licites et correctement effectuées

Indication  **:** utilisez la commande `basename`

Exercice 9 – Guillemets, quotes ou back quotes ?

Écrire un programme shell qui commence par afficher « *Entrer le nom d'un répertoire :* » puis lit le nom (relatif ou absolu) d'un répertoire (commande `read`). Le programme affichera ensuite le texte « *Le répertoire xxx contient les fichiers suivants* : » suivi de la liste des fichiers contenus dans le répertoire.

1) Ecrire une première version dans laquelle *xxx* est le nom du répertoire reçu en argument.

2) Ecrire une seconde version dans laquelle *xxx* correspond au chemin absolu du répertoire entré par l'utilisateur, même si celui-ci est donné en relatif. On suppose toutefois que le répertoire donné en paramètre se trouve dans le répertoire courant (inutile d'effectuer une recherche dans toute l'arborescence).
Le script devra en plus afficher un message d'erreur adéquat dans les cas suivants :

- si le script n'est pas appelé avec le bon nombre d'arguments,
- si le nom entré par l'utilisateur n'est pas un répertoire,
- si c'est un répertoire non lisible.

Exercice 10 – Les expressions régulières

Dans une fichier :

1) Chercher toutes les lignes commençant par '`a`' ou '`A`'.

2) Chercher toutes les lignes finissant par '`rs`'.

3) Chercher toutes les lignes contenant au moins un chiffre.

4) Chercher toutes les lignes commençant par une majuscule.

5) Chercher toutes les lignes commençant par '`B`', '`E`' ou '`Q`'.

6) Chercher toutes les lignes finissant par un point d'exclamation.

7) Cherchez toutes les lignes se terminant par un point.

8) Chercher toutes les lignes ne finissant pas par un signe de ponctuation (point, virgule, point-virgule, deux-points, point d'interrogation, point d'exclamation).

9) Comment chercher tous les mots contenant un '`r`' précédé de n'importe quelle lettre majuscule ou minuscule ?

10) Chercher tous les mots dont la seconde lettre est un '`r`'.