Université d'AIN TEMOUCHENT – Algérie

Faculté des Sciences et de la Technologie

Département de Mathématiques et d'Informatique

Matière: **Système d'exploitation avancé**

Chapitre : Les variables	**TP n° : 03**

1. Les variables d'environnement

Les variables d'environnement sont des variables que l'on peut utiliser dans n'importe quel programme. On parle aussi parfois de variables globales contrairement aux variables classiques qui sont utilisées uniquement dans les scritps où elles existent.

La commande **env** affiche toutes les variables d'environnements que vous avez actuellement en mémoire

Code: Console

```
$ env
ORBIT_SOCKETDIR=/tmp/orbit-info
GLADE_PIXMAP_PATH=:/usr/share/glade3/pixmaps
TERM=xterm
SHELL=/bin/bash
GTK_MODULES=canberra-gtk-module
USER=info
PATH=/home/info/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bi
n:/usr/GDM_XSERVER_LOCATION=local
PWD=/home/info/bin
EDITOR=nano
SHLVL=1
HOME=/home/info
OLDPWD=/home/info
```

Décrivons quelque uns:

- **SHELL** : indique quel type de shell est en cours d'utilisation (sh, bash, ksh...)
- **PATH** : une liste des répertoires qui contiennent des exécutables que nous pouvons lancer sans indiquer leur répertoire.
- **EDITOR** : l'éditeur de texte par défaut qui s'ouvre lorsque cela est nécessaire.
- **HOME** : la position de votre dossier home.
- **PWD** : le dossier dans lequel vous vous trouvez.
- **OLDPWD** : le dossier dans lequel vous vous trouviez auparavant.

Remarque :

Notez que les noms de ces variables sont, par convention, écrits en majuscules.

Comment utiliser ces variables dans les scripts ? Il suffit de les appeler par leur nom !

Code: Bash

```
#!/bin/bash
echo "Le répertoire en cours est $PWD"
```

Résultat

Code: Console

```
Le répertoire en cours est /home
```

Remarque :

1) *Pour définir sa propre variable d'environnement, on utilise la commande **export**.*

Code: Bash

```
#!/bin/bash
export MAVARIABLE
```

2) Note un peu technique: un script lancé comme on le fait depuis le terminal ne peut modifier une variable d'environnement pour l'ensemble du système, car c'est un processus enfant. Un processus enfant ne peut pas modifier les variables d'environnement pour ses parents.

2. Les variables des paramètres

Comme toutes les commandes, les scripts peuvent accepter des paramètres. Ainsi, on pourrait appeler notre script comme ceci :

Code: Console

```
./variables.sh param1 param2 param3
```

Comment récupérer ces paramètres dans le script. Pour cela, des variables sont automatiquement créées :

- **$#** : contient le nombre de paramètres
- **$0** : contient le nom du script exécuté (ici "./variables.sh")
- **$1** : contient le premier paramètre
- **$2** : contient le second paramètre
- ...
- **$9** : contient le 9ème paramètre

Code: Bash

```
#!/bin/bash
echo "Vous avez lancé $0, il y a $# paramètres"
echo "Le paramètre 1 est $1"
```

Résultat

Code: Console

```
$ ./variables.sh param1 param2 param3
Vous avez lancé ./variables.sh, il y a 3 paramètres
Le paramètre 1 est param1
```

Si on utilise plus de 9 paramètres, par exemple 15

Code: Console

```
./script.sh fichier1 fichier2 fichier3 fichier4 ... fichier14 fichier15
```

Pour traiter autant de paramètres, on les fera en général un par un. On peut "décaler" les paramètres dans les variables $1 $2 etc. avec la commande **shift**.

Code: Bash

```
#!/bin/bash
echo "Le paramètre 1 est $1"
shift
echo "Le paramètre 1 est maintenant $1"
```

Résultat

Code: Console

```
$ ./variables.sh param1 param2 param3
Le paramètre 1 est param1
Le paramètre 1 est maintenant param2
```

3. Les tableaux

Pour définir un tableau, on peut faire ceci :

Code : Bash

```
tableau=('valeur0' 'valeur1' 'valeur2')
```

Cela crée une variable tableau qui contient 3 valeurs (valeur0, valeur1, valeur2).

Pour accéder à une case du tableau, on doit utiliser la syntaxe suivante :

Code : Bash

```
${tableau[2]}
```

... ceci affichera le contenu de la case n°2 (donc "valeur2").

Remarque :

Les cases sont numérotées à partir de 0. Notez par ailleurs que pour afficher le contenu d'une case du tableau, vous devez entourer votre variable d'accolades comme fait dans le dernier script ${tableau[2]}.

Vous pouvez aussi manuellement définir le contenu d'une case :

Code : Bash

```
#!/bin/bash
tableau=('valeur0' 'valeur1' 'valeur2')
tableau[5]='valeur5'
echo ${tableau[1]}
```

Résultat

Code : Console

```
valeur1
```

Le tableau peut avoir autant de cases que nous le voulons. La numérotation n'a pas besoin d'être continue, nous pouvons sauter des cases sans problème. Pour afficher l'ensemble du contenu du tableau d'un seul coup, on utilise ${tableau[*]} :

Code : Bash

```
#!/bin/bash
tableau=('valeur0' 'valeur1' 'valeur2')
tableau[5]='valeur5'
echo ${tableau[*]}
```

Résultat

Code : Console

```
valeur0 valeur1 valeur2 valeur5
```

Exercice :

1) Écrire un script qui calcule la somme de 5 valeurs X1, X2, X3, X4, et X5 passées comme arguments au nom du script.

2) La même question précédente qui calcule 5 valeurs mais en utilisant deux variables seulement *Indication (utilisez la commande shift)*.

3) Est-il possible de le faire avec une seule variable ?

4) Reprenez la question 1° où les cinq valeurs sont stockées dans un tableau et calculer leur somme. Le résultat doit ainsi apparaitre comme suit:
"Ces 5 variables 1 2 3 4 5 font 15"