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 conditions	**TP n° : 06 (3/3)**

1. Effectuer plusieurs tests à la fois

Dans un if, il est possible de faire plusieurs tests à la fois. En général, on vérifie :

- Si un test est vrai ET qu'un autre test est vrai
- Si un test est vrai OU qu'un autre test est vrai

Les 2 symboles à connaître sont :

- **&&** : signifie "et"
- || : signifie "ou"

Il faut encadrer chaque condition par des crochets. Prenons un exemple :

Code : Bash

```
#!/bin/bash
if [ $# -ge 1 ] && [ $1 = 'koala' ]
then
echo "Bravo !"
echo "Vous connaissez le mot de passe"
else
echo "Vous n'avez pas le bon mot de passe"
fi
```

Le test vérifie 2 choses :

Qu'il y a au moins un paramètre ("si $# est supérieur ou égal à 1") **et**

Que le premier paramètre est bien koala ("si $1 est égal à koala")

Si ces 2 conditions sont remplies, alors le message indiquant qu'on a trouvé le bon mot de passe s'affichera.

Code : Console

```
$ ./conditions.sh koala
Bravo !
Vous connaissez le mot de passe
```

Remarque :

Notez que les tests sont effectués l'un après l'autre, et seulement s'ils sont nécessaires. Bash vérifie d'abord s'il y a au moins un paramètre. Si ce n'est pas le cas, il ne fera pas le second test puisque la condition sera de toute façon fausse.

2. Inverser un test

Le point d'exclamation "!" exprime la négation d'un test.

Code : Bash

```
if [ ! -e fichier ]
then
echo "Le fichier n'existe pas"
fi
```

3. case : tester plusieurs conditions à la fois

Reprenons le dernier script de la fiche de TP n° : 04 (1/3) :

Code : Bash

```
#!/bin/bash
if [ $1 = "Karim" ]
then
echo "Salut Karim !"
elif [ $1 = "Zakaria" ]
then
echo "Bien le bonjour Zakaria"
elif [ $1 = "Ali" ]
```

```
then
echo "Hé Ali, ça va ?"
else
echo "Je ne te connais pas, ouste !"
fi
```

Le rôle de case est de tester la valeur d'une même variable, mais de manière plus concise et lisible.
Voyons comment on écrirait la condition précédente avec un case :

Code : Bash

```
#!/bin/bash
case $1 in
"Karim")
echo "Salut Karim !"
;;
"Zakaria")
echo "Bien le bonjour Zakaria"
;;
"Ali")
echo "Hé Ali, ça va ?"
;;
*)
echo "Je ne te connais pas, ouste !"
;;
esac
```

Analysons la structure du case !

Code : Bash

```
case $1 in
```

Tout d'abord, on indique que l'on veut tester la valeur de la variable $1.

Code : Bash

```
"Karim")
```

Là, on teste une valeur. Cela signifie "Si $1 est égal à Karim". Notez qu'on peut aussi utiliser une étoile comme joker : "K*" acceptera tous les mots qui commencent par un K majuscule.
Si la condition est vérifiée, tout ce qui suit est exécuté jusqu'au prochain double point-virgule :

Code : Bash

```
;;
```

Important, il ne faut pas l'oublier : le double point-virgule dit à bash d'arrêter la lecture du case là. Il saute donc à la ligne qui suit le "esac" qui signale la fin du case.

Code : Bash

```
*)
```

C'est en fait le "else" du case. Si aucun des tests précédents n'a été vérifié, alors c'est cette section qui sera lue.

Code : Bash

```
esac
```

Marque la fin du case (esac, c'est "case" à l'envers !).

Exercice 1 :

Créer un script dans lequel deux nombres opérandes et un signe opérateur (+-*/) devront être donnés en paramètres, ou saisis. Le script doit réaliser l'opération souhaitée en utilisant la structure de contrôle case.

Exemple :

```
[~] ./calculette.sh 7 + 4
Le résultat est : 11
```