

UNIX

TP n°4 (Corrections)

Exercice 1 Scripts C-shell

1. *Écrire un script C-Shell qui affiche le message "Bonjour" sur le terminal.*

```
#!/bin/csh

echo "Bonjour"

et droits d'accès modifiés avec chmod u+x Bonjour
```

2. *Écrire un script C-Shell contenant deux variables A et B de valeurs respectives 2 et 5 et qui affiche la somme de ces deux variables.*

```
#!/bin/csh

@ A=2
@ B=5
@ somme = $A + $B
echo La somme est $somme
```

3. *Modifier le script précédent en demandant les valeurs de A et B à l'utilisateur.*

```
#!/bin/csh

echo "Entrer A"
@ A=$<
echo "Entrer B"
@ B=$<
@ somme = $A + $B
echo La somme est $somme
```

4. *Écrire un script C-Shell affichant si l'utilisateur a ou non passé des paramètres à la ligne de commande.*

```
#!/bin/csh

if ($# == 0) then
echo "Aucun paramètre"
else
echo "$# paramètres"
endif
```

5. *Modifier le script précédent pour afficher les paramètres lorsqu'il y en a.*

```
#!/bin/csh

if ($# == 0) then
echo "Aucun paramètre"
else
echo "$# paramètres que voici"
foreach i ($*)
echo $i
end
endif

ou

#!/bin/csh

if ($# == 0) then
echo "Aucun paramètre"
else
echo "$# paramètres que voici $*"
endif
```

6. *Écrire un script C-Shell qui affiche "Bonjour", "Bonsoir", "Salut" ou "Hi" selon que le paramètre passé à la ligne de commande est respectivement 1, 2, 3 ou un autre chiffre.*

```
#!/bin/csh

if ($# == 0) then
echo "Donner un paramètre"
else
switch ($1)
case 1 :
echo "Bonjour"
breaksw
case 2 :
echo "Bonsoir"
breaksw
case 3 :
echo "Salut"
breaksw
default :
echo "Hi"
endsw
endif
```

Exercice 2 Question

1. A quoi sert la séquence C-shell suivante ?

```
#!/bin/csh
set question='reponse (oui/non) ->'
set reponse=non
while ($reponse != oui)
    echo -n $question " "
    set reponse=$<
end
echo "on sort par $reponse"
```

Proposer une exécution possible prenant en compte les différents cas de figure.

Réponse : Permet d'itérer une lecture au clavier jusqu'à l'obtention de la chaîne oui.
-n permet de ne pas passer à la ligne

```
> ./question
reponse (oui/non) -> aa
reponse (oui/non) -> non
reponse (oui/non) -> oui
on sort par oui
```

2. On désire à présent offrir à l'utilisateur de quitter une boucle en tapant non seulement oui mais également tout mot commençant par o. Utiliser un `switch` pour préciser à l'utilisateur si l'on a supposé qu'il voulait écrire oui (mot commençant par o) ou qu'il voulait dire non (mot commençant par n) ou que le programme ne comprend pas.

```
#!/bin/csh
set reponse="non"
while ($reponse != oui)
    echo -n "Voulez-vous quitter? (oui/non)"
    set reponse = $<
    switch ( $reponse )
        case oui:
            echo "Vous avez répondu oui"
            breaksw
        case non:
            echo "Vous avez répondu non"
            breaksw
        case o*:
            echo "Je suppose que vous voulez répondre oui"
            set reponse = "oui"
            breaksw
        case n*:
            echo "Je suppose que vous voulez répondre non"
            breaksw
        default:
            echo "Je ne comprends pas votre réponse"
            breaksw
    endsw
end
```

Nouveautés : o* dans le switch

3. Créer à présent des alias permettant de préciser si l'on sort de la boucle ou non et les utiliser dans le script précédent.

```
#!/bin/csh

# Déclaration d'alias
alias pas_sortir 'echo "On ne sort pas" '
alias sortir 'echo "On sort" '

set reponse="non"
while ($reponse != oui)
  echo -n "Voulez-vous quitter? (oui/non)"
  set reponse = $<
  switch ( $reponse )
    case oui:
      echo "Vous avez répondu oui"
      sortir
      breaksw
    case non:
      echo "Vous avez répondu non"
      pas_sortir
      breaksw
    case o*:
      echo "Je suppose que vous voulez répondre oui"
      set reponse = "oui"
      sortir
      breaksw
    case n*:
      echo "Je suppose que vous voulez répondre non"
      pas_sortir
      breaksw
  default:
    echo "Je ne comprends pas votre réponse"
    pas_sortir
    breaksw
  endsw
end
```

Exercice 3 Calculatrice

Écrire un script C-shell appelé *Calculatrice* prenant en argument un nombre, demandant une deuxième valeur, puis une opération à effectuer parmi +, - et / (pour simplifier, on ne considèrera pas la multiplication) et qui affiche le résultat. Une exécution possible sera

```
./Calculatrice 15
Quelle valeur ?
2
Quelle operation voulez-vous faire ?
+
Le resultat est 17
```

Le script devra renvoyer un message d'erreur lorsque le nombre d'arguments est incorrect.

```
#!/bin/csh
if ( $#argv != 1 ) then
    echo "Passer une premiere valeur en argument"
    exit 0
else
    echo "Quelle valeur ?"; set n=$<
    echo "Quelle operation voulez-vous faire ?"; set op=$<
    switch ($op)
    case + :
        @ res=$1 + $n ; breaksw
    case - :
        @ res=$1 - $n ; breaksw
    case [*] :
        @ res=$1 * $n ; breaksw
    case / :
        @ res=$1 / $n ; breaksw
    default :
        echo "on ne sait pas faire cette opération..."
    endsw
endif
if ($?res) then
echo "Le resultat est " $res
endif
```

Exercice 4 Horloge

Écrire un script C-shell qui, en fonction de l'heure courante (retournée par commande `date`) affiche "Bonjour" entre 0h et 12h, "Bon après-midi" de 12h à 17h, et "Bonne soirée" de 17h à 24h. On supposera que la réponse à la commande `date` est de la forme `mardi 6 novembre 2007, 15:50:07`. Indication : la commande `awk -F:` s'utilise de la même façon que `awk` mais considère le séparateur : au lieu de l'espace.

```
#!/bin/csh
@ heure = `date | awk '{print $5}' | awk -F: '{print $1}`
echo $heure
if ( $heure < 12 ) then
    echo "Bonjour !"
else if ( $heure < 17 ) then
    echo "Bon après-midi !"
else
    echo "Bonsoir !"
endif
```

Exercice 5 Lister le contenu d'un répertoire

Écrire un script C-shell appelé `CompterFich` qui permet de lister les fichiers du répertoire passé en paramètre en spécifiant pour chacun d'eux le type de fichier (fichier ordinaire, répertoire ou autre) et d'afficher à la fin le nombre de fichiers ordinaires et le nombre de répertoires trouvés.

```

#!/bin/csh
@ d = 0
@ f = 0
if ($#argv == 1) then
    set Rep="$1/*"
else set Rep="*"
endif
foreach nomfic ('ls $Rep')
    if ( -f $nomfic ) then
        echo $nomfic " : fichier ordinaire"
        @ f = $f + 1
    else if ( -d $nomfic ) then
        echo $nomfic " : repertoire"
        @ d = $d + 1
    else echo $nomfic " : autre type de fichier"
    endif
end
echo "Fichiers : $f      Repertoires : $d"

```

Exercice 6 Renomme (4 points)

Exercice pris dans le Rifflet page 42.

Écrire un script C-shell permettant de renommer une liste de fichiers du répertoire de travail. On supposera que les références de fichiers à changer sont données en paramètres et plus précisément sont tous les paramètres à partir du second. Les nouvelles références seront obtenues à partir du préfixe commun donné en premier argument, le suffixe étant constitué par un entier compris entre 1 et n si n est le nombre de fichiers à renommer. Par exemple ce script appliqué aux arguments `f g h i j`, renommara respectivement `g`, `h`, `i` et `j` en `f1`, `f2`, `f3` et `f4`. On veillera à ce que le script ne renomme que les fichiers ordinaires. On décidera par ailleurs que si la commande n'a qu'un seul paramètre, tout le répertoire de travail est traité. Enfin, on supposera pour simplifier le travail qu'aucun nouveau nom ne correspond à un lien existant dans le répertoire.

```

#!/bin/csh
switch ($#)
case 0 :
    #pas de parametre
    echo "nombre de parametres incorrect"
    exit 1
    breaksw
case 1 :
    # un seul parametre -> on traite tous les fichiers
    # du repertoire de travail
    @ a = 1
    foreach i (*)
        if (-f $i) then
            echo "mv $i $1$a"
            @ a = $a + 1
        endif
    end
end

```

```

        breaksw
default :
    # plus d'un parametre -> on traite la liste des parametres
    @ a = 1
    set b=$1
    shift
    foreach i ($*)
        if (-f $i) then
            echo "mv $i $b$a"
            @ a = $a + 1
        endif
    end
    breaksw
endsw

```

Remarque : Dans la correction, j'ai mis des " au lieu des ' pour ne pas executer la commande mv à chaque fois.

2eme version numérotant à l'envers :

```

#!/bin/csh
switch ($#)
case 0 :
    #pas de parametre
    echo "nombre de parametres incorrect"
    exit 1
    breaksw
case 1 :
    # un seul parametre -> on traite tous les fichiers
    # du repertoire de travail
    @ a = 1
    foreach i (*)
        if (-f $i) then
            echo "mv $i $1$a"
            @ a = $a + 1
        endif
    end
    breaksw
default :
    # plus d'un parametre -> on traite la liste des parametres
    set b=$1
    while ($# != 0)
        if (-f $1) echo "mv $1 $b$#"
        shift
    end
    breaksw
endsw

```

Nouveauté : utilisation du test d'existence d'un fichier et du shift