Cours 9
Scripts

I. Principe

On peut mémoriser une série de commandes (un script) en les notant dans un fichier (fichier de texte). Il suffit alors de rendre le fichier exécutable (commande `chmod`) pour que le script s'exécute quand on appelle le nom du fichier comme une commande.

Quand on écrit des sripts, on peut utiliser les arguments de la ligne de commande comme en C. Le nom du script est $0, les mots suivants sur la ligne de commande sont $1, $2, etc. Le nombre de mots est $#.

On peut créer des paramètres par `set` (ex.: `set source=/home/users/TPINFO`). On utilise les paramètres en ajoutant un `$` devant leur nom (ex. `cp $source ~`, ou `cp ${source}/* ~`). $< lit une entrée clavier.

On dispose aussi de structures de contrôle : branchements (`if`, …), boucles (`while`, …). Attention, les structures de contrôle dépendent de l'interpréteur de commandes utilisé.

II. Structures de contrôle du Cshell (csh)

```
if (expression) commande
```

```
if (expression) then
commande
…
[else if (expression) then
commande
…]
[else
commande
```

```
switch (chaîne)
case label :
        commande
        [breaksw]
[case label :
        commande
        [breaksw]]
…
[default :
        commande
```

```
foreach nom (liste)
commande
…
end
```

```
while (expression)
commande
…
end
```

```
repeat nbre commande
```

Dans le `switch`, les labels de cas admettent les caractères de remplacement du shell (`?` pour n'importe quel caractère, `*` pour n'importe quelle chaîne, `[]`pour énumérer des chaînes).

Pour les tests, les **expressions** sont groupées entre parenthèses. On dispose des opérateurs de calcul du langage C : opérations arithmétiques (avec les mêmes priorités qu'en C : en cas de doute, il vaut mieux mettre des parenthèses), opérations bit à bit, opérateurs logiques, opérateurs de comparaison, plus les opérateurs `=~` et `!~` (filtre, ne filtre pas), qui admettent à droite les caractères de remplacement du shell. On peut aussi utiliser le résultat d'une fonction, mais le vrai est 1 et le faux 0, alors que la convention Unix est qu'un test réussi renvoie 0 (par exemple `grep` renvoie 0 s'il trouve au moins une ligne, 1 sinon). La mise entre accolades `{…}` inverse ces valeurs, de façon à ce que les tests fonctionnent comme attendu. On dispose aussi de tests sur les fichiers -d, -e, -f, etc.

III. Exemples de scripts

Le # commence une remarque qui va jusqu'à la fin de la ligne. le ; sépare deux commandes et équivaut à un changement de ligne.

1) Lister les noms de login et noms réels des utilisateurs

```
#!      /bin/csh
```

```
#       appeler listlog nom_du_groupe
set group=$1  # ou $argv[1]
grep $group /etc/passwd | cut -d: -f1,5
```

2) Faire un script **`ou_est`** qui cherche le fichier dont le nom est passé en premier argument dans les catalogues dont les noms sont passés ensuite (en nombre indéterminé). Par ex :

```
> ou_est exam algo/1A algo/2A sys99/1A sys00/1A
algo/2A/exam
sys99/1A/exam
```

```
#!      /bin/csh
#  Appeler ou_est nom_de_fichier catalogue1 [catalogue2 […]]
foreach i ($2*) # $2* : tous les arguments à partir de $2
if (-e ${i}/$1 ) then
echo ${i}/$1
end
```

3) Script **print** : choisir la méthode d'impression en fonction du suffixe

```
#!      /bin/csh
#       appeler print nomfic1.suf [nomfic2.suf […]]
foreach i ($*) # $* = $1*
switch ($i)
case *.ps: lp $i ; breaksw
case *.dvi: dvips $i ; breaksw
case *.txt: pr $i | lp ; breaksw
default: echo –n Je ne connais pas la methode
         echo d'impression pour $i
         breaksw
endsw  # fin du switch
end     # fin du foreach
```

4) **chsuf** : renommer les fichiers de suffixe s1 en suffixe s2

```
#!      /bin/csh
# appeler chsuf suffixe1 suffixe2
foreach fich (*.$1)
set name=`basename $fich .${1}` # entre backquote
mv $fich ${name}.$2
end
```

5) **diffuse** : un script permettant de diffuser un fichier (texte) à tous les utilisateurs (dans leur *home directory*) en provenance d'un répertoire fixé ; si le fichier existe déjà, l'envoie en mail.

```
#!            /bin/csh
# diffuse : appeler cat /etc/passwd | diffuse <nom_de _fichier>

set sourcedir=$HOME/exemple
set fich=$1
while (1)
   set lect="$<"  # lit stdin, donc une ligne de /etc/passwd
   if  ("$lect" == "" ) then       # saute les lignes vides
      break
   endif
   set ligne=(`echo -n $lect | tr ": " " :" ` )
   if ( -f ${ligne[6]}/$fich ) then
      (echo nouvelle version de $fich ; cat ${sourcedir}/$fich)  | mail $ligne[1]
   else cp ${sourcedir}/$fich ${ligne[6]}
   endif
end
```