DOSI - Systèmes d'exploitation – TP 3
programmation csh

12 septembre 2010

Objet du TP : csh

1 Petits algorithmes

Q1 : Contruire un prgramme en C-shell permettant l'addition de deux nombres entiers. On proposera une version dans laquelle les nombres sont saisis au clavier par l'utilisateur et une version dans laquelle les nombres sont des paramtres du programme.

Q2 : Construire un programme C-Shell calculant le minimun de trois nombres donnés en paramètres. On vérifiera le nombre de paramètres avant de procéder à la recherche du plus petit.

Q3 : Concevoir un programme qui permet de calculer la somme des tailles des fichiers du répertoire courant. On ne fera pas ici de recherche arborescente.
Compléter le programme pour connaitre le fichier le plus grand et le fichier le plus petit.

Q4 : Soit le programme suivant `prog.csh` :

```
#!/bin/csh -f

if ( $#argv == 1 ) then
        set val = $1

        if ( $val == 1 ) then
                echo "1"
        else
                @ nouv_val = $1 - 1
                set resul = `./prog.csh $nouv_val`
                @ resul = $resul * $val
                echo $resul
        endif
else
        echo " usage : prog.csh <argument>"

endif
```

Que fait ce programme ? Vérifiez votre analyse avec les valeurs 3 (resultat théorique : 6) , 5 (resultat théorique : 120) et 14 (resultat théorique : 87 178 291 200).

Q5 : Construire un programme C-shell permettant d'acquérir dix nombres tapés au clavier, de caluler leur somme, leur produit ainsi que la valeur minmum et maximum.

Q6 : Concevoir un programme qui permet de calculer la somme des tailles des fichiers d'une arborescence. Utilisez pour cela une exploration recursive.

Q7 : On souhaite trier un tableau de 10 entiers positifs à l'aide d'un programme écrit en C-Shell. On utilisera l'algorithme suivant :

```
pour i depuis 1 jqa 10
     trouver la plus petite valeur dans la tableau A
     ranger cette valeur dans la case i du tableau B
     remplacer dans le tableau A, la valeur par -1
fpour
afficher le tableau B
```

Construire un programme C-shell permettant d'implémenter complètement cet algorithme.

2 Compression

La commande `compress` permet la compression de fichiers avec un taux d'environ 50%. La commande `gzip` réalise la même opération, mais avec un taux de compression proche de 70%. Afin d'économiser de la place dans une arborescence UNIX, on se propose de construire un programme C-shell permettant automatiquement de décompresser (commande `uncompress`) les fichiers obtenus par `compress` (fichiers suffixés `.Z`) et de les recompresser en utilisant `gzip`.

Q8 : Ecrire en C-shell un programme qui réalise une telle opération dans un répertoire donné en paramètre sans exploration des sous répertoires.

Q9 : Ecrire en C-shell un programme qui réalise une telle opération dans un répertoire donné en paramètre avec exploration des sous répertoires.

Q10 : Modifiez le programme précédant pour déterminer le gain de place réalisé.

Remarque 1 : utilisation des *modifiers* `:r` et `:e` du C-shell :

le programme suivant :

```
#!/bin/csh

set file = toto.csh
echo $file:r
echo $file:e
```

produit le résultat :

```
toto
csh
```

3 Date

Q11 : La commande date écrit la date en anglais. Construisez un programme qui permet de l'écrire en français.

Exemple : `Fri Apr 7 15 :59 :10 MET DST 1995`
devient
`Vendredi 7 Avril 1995, il est 15 :59 :10`