

Contrôle court de système d'exploitation S2, Semestre initial

Département d'informatique IUT Villetaneuse

Mercredi 24 mars 2010, 75 minutes

Remarques : tous les documents de cours, td/tp sont autorisés. Le barème est indicatif.

Nom :

Prénom :

Groupe :

1- Génération d'un fichier **HTML**

On souhaite améliorer un script bash nommé `RenduTpHtml.sh`. On commence par chercher à comprendre ce que fait le script. Pour rappel, un fichier HTML est un fichier texte avec une extension `.html`. Il est constitué de 'balises' : `< début balise> contenu </ fin balise >`, qui peuvent s'imbriquer. Par exemple, un fichier HTML commence par une balise `<html>` et se termine par la balise `</html>`.

a) Comment s'utilise le script suivant appelé `RenduTpHtml.sh` ? (1pt)

```
1- #!/bin/bash
2- if [ $# -ne 2 ]
3- then
4-     echo "usage: $0 PRENOM NOM"
5-     exit
6- else
7-     # $1=PRENOM $2=NOM
8-     echo "<html>"
9-     echo "<head>"
10-    echo "<title>"
11-    echo "Generation HTML"
12-    echo "</title>"
13-    echo "</head>"
14-    echo "<body>"
15-    echo "<h1>"
16-    echo "Solutions des TPs de $1 $2"
17-    echo "</h1>"
18-
19-    j=1
20-    for myfichier in `ls`
21-    do
22-        echo "<h2>${j}) $myfichier</h2>"
23-        j=$((j+1))
24-        echo "<pre>"
25-        cat "$myfichier"
26-        echo "</pre>"
27-    done
28-    echo "</body>"
29-    echo "</html>"
30- fi
Répondre ici :
```

`./RenduTpHtml.sh` Christophe Cérin

b) Que fait ce script ? (1pt)

Le script construit un fichier Html qui contient les sources/contenus de tous les fichiers présents dans le répertoire courant. Le nom de chaque fichier précède son contenu. Au début de la page Html apparaît également les noms et prénoms passés en paramètre au script.

c) Modifiez le script pour que le script ne fasse pas partie du résultat (indiquez uniquement les lignes que vous ajoutez / supprimez) : (3pt)

Il convient de modifier la boucle do... done (à partir de la ligne 22) pour conditionner les lignes 22-26 par un test sur le nom du fichier :

```
if [ "$myfichier" != "$0" ]; then
    # lignes 22 à 26
fi
```

d) On suppose que tous les fichiers sont de la forme « qXXX.sh » avec X étant un digit. Modifiez le script pour que le script présente le résultat trié c.à.d avec le fichier portant le nom avec le plus petit entier en premier et le fichier portant le nom avec le plus grand entier en dernier (indiquez uniquement les lignes que vous ajoutez / supprimez) : (7 pt)

```
# Début de vos explications
```

Ici il suffit de modifier la génération de la liste des fichiers pour la boucle aux lignes 22 à 26 pour que les fichiers apparaissent en ordre croissant. Ceci se réalise par un 'sort' :

```
for myfichier in `ls | sort`
```

Début du code proposé

2- Commandes simples - 2 pts

a) On lance la commande `ls` et on veut récupérer la liste des fichiers qui comportent au moins un caractère `digit` en première position du nom. Donnez une solution avec la commande `grep -E (egrep)`.

b) On lance la commande `ls` et on veut récupérer la liste des fichiers qui comportent au plus un caractère `digit` en dernière position du nom. Donnez une solution avec la commande `grep -E (egrep)`.

3- Mélange d'entités lexicales - 6 pts

Ecrire un script qui prend en paramètre un fichier texte et qui va générer sur la sortie standard le contenu du fichier passé en paramètre mais avec les mots dans un autre ordre qu'au départ. On pourra utiliser un tableau pour ranger les mots ainsi que le bout de texte suivant (exemple) pour générer 10 entiers aléatoires (qui représenteront des positions dans un tableau dans votre cas) :

```
$ c=`python -c "import random;print random.sample(xrange(10),10);"`  
$ echo $c  
[4, 9, 1, 5, 0, 7, 6, 8, 2, 3]
```

Exemple d'utilisation du script à écrire si le fichier en entrée vaut :

```
/bin 120  
/etc 60  
/var 30
```

```
$ /usr/local/bin/bash OrdreAleatoire.sh num.txt  
30  
/etc  
/var  
60  
120  
/bin
```

```
# Début de vos explications ici. Veuillez discerner les problèmes purement  
# techniques et les problèmes d'organisation algorithmique. Vous devez faire  
# des choix : quel algorithme suivez-vous ? Quelles structures de données  
# utilisez-vous et pour faire quoi ?
```

Pour la correction, nous vous proposons deux codes. Nous vous demandons de revenir à l'algorithme / choix algorithmiques qui ont guidés à ces solutions.

```
#!/usr/local/bin/bash  
#  
# Ce script lit un fichier passe en parametre et melange les mots du texte  
# dans un ordre aleatoire.  
#  
if [ $# -ne 1 -a -e $1 ]  
then  
    echo "Il faut un argument qui doit etre un fichier existant"  
    exit  
else  
  
    # on compte le nombre de mots du fichier texte  
    nb=`cat $1 |wc -w`  
  
    # on aloue deux tableaux de nb elements  
    declare -a MonTableau  
    declare -a MonTableauBis  
  
    # on genere une sequence aleatoire MonTableau de nb entiers aleatoires  
    c=`python -c "import random;print random.sample(xrange($nb),$nb);"`  
    c=${c:1:${#c}-2}  
    c=`echo "$c" |sed 's/,/ /g'`  
    #echo $c  
    j=0;  
    for i in `echo $c`  
    do
```

```

        MonTableau[$j]=$i
        j=$((j+1))
done

#j=0
#for i in `echo $c`
#do
#    echo "${MonTableau[$j]}"
#    j=$((j+1))
#done

#exit

# on range le ieme mots de $1 a la position MonTableau(i) dans le tableau
MonTableauBis
j=0
for mot in `cat $1`
do
    MonTableauBis[${MonTableau[$j]}]=${mot}
    #echo "j=$j ** ${MonTableauBis[${MonTableau[$j]}]} **"
    j=$((j+1))
done

# on affiche le contenu du tableau
for (( i=0;i<${nb};i=$((i+1)) ))
do
    echo "${MonTableauBis[$i]}"
done
echo

```

```

#!/usr/local/bin/bash
#
# Ce script lit un fichier passe en parametre et melange les mots du texte
# dans un ordre aleatoire.
#

if [ $# -ne 1 -a -e $1 ]
then
    echo "Il faut un argument qui doit etre un fichier existant"
    exit
else
    #1] on copie les mots du fichier dans un tableau, au passage, on les compte...
    nb=0
    for mot in `cat $1`
    do
        MonTableau[$nb]=${mot}
        nb=$((nb+1))
    done

    #2] on genere une sequence aleatoire MonTableau de nb entiers aleatoires
    c=`python -c "import random;print random.sample(xrange($nb), $nb);"`
    c=${c:1:${#c}-2} # on enleve les []
    c=`echo "$c" |sed 's/,/ /g'` # on remplace les ',' par des ' '

    #3] on affiche le contenu du tableau dans un ordre aleatoire
    for i in `echo $c`
    do
        echo "${MonTableau[$i]}"
    done
fi

```