

TP n° 3 - Boucles, tests et arithmétique

Pour cette séance, se placer dans le répertoire `data`.

Exercice 1. Écrire un script shell qui tire au hasard un mot dans le dictionnaire du système `/usr/share/dict/words`, à l'aide de `wc` et de la variable `$RANDOM` qui contient à chaque utilisation un entier aléatoire. Le mot est ensuite affiché.

```
# exécuter ce script dans le répertoire data

# on récupère le nombre de lignes du fichier
# /usr/share/dict/words dans nbLignes
nbLignes=$( wc -l < /usr/share/dict/words)

# on génère un entier aléatoire l
# le fichier étant gros, il faut obtenir un reste qui balaie
# l'intervalle [1..nbLignes[ et donc diviser par un entier
# grand
l=$((($RANDOM * 32768 + $RANDOM) % $nbLignes + 1))

# affiche le contenu de la ligne n° l
cat /usr/share/dict/words| head -n $l | tail -n 1
```

Exercice 2. Écrire un script shell qui forme un juron du capitaine Haddock en tirant au hasard un adjectif et un qualificatif dans les fichiers `haddock/noms` et `haddock/qualificatifs`.

```
#exécuter ce script dans le répertoire data
# les variables fnoms et fqual contiennent les noms
# des fichiers noms et qualificatifs du répertoire haddock
fnoms="haddock/noms"
fqual="haddock/qualificatifs"

# les variables tnoms et tqual contiennent les tailles respectives
# des fichiers
tnoms=$( wc -l < "$fnoms")
tqual=$( wc -l < "$fqual")

# nom et qual contiennent le contenu d'une ligne aléatoire
nom=$( cat "$fnoms" | head -n $((($RANDOM*$tnoms + 1)) | tail -n 1)
qual=$( cat "$fqual" | head -n $((($RANDOM*$tqual + 1)) | tail -n 1)

# affichage du juron
echo $nom $qual
```

Exercice 3. Écrire un script qui crée dans le même répertoire un fichier `cyrano_num` obtenu à partir du fichier `txt/cyrano.txt`, en numérotant ses lignes à l'aide de `seq` et `paste`. On notera au passage que `cat -n` permet d'en faire autant plus simplement.

```
# exécuter ce script dans le répertoire data
f="txt/cyrano.txt"
taille=$( wc -l < "$f")
# construit le fichier numeros dans le répertoire /tmp
# où chaque ligne contient un entier de 1 à $taille
seq 1 $taille > /tmp/numeros

# construit le fichier cyrano_num
```

```

paste /tmp/numeros "$f" > cyrano_num
# supprime le fichier temporaire numeros
rm /tmp/numeros

# Solution 2, qui ne crée pas le fichier temporaire numeros :
# noter l'utilisation de - qui fait référence au résultat de
# la commande précédente
seq 1 $( wc -l < "$f" ) | paste - "$f" > cyrano_num2

```

Exercice 4. Écrire un script qui affiche les termes de la suite de Syracuse, définie par :

$$u(n+1) = u(n) / 2 \text{ si } u(n) \text{ est pair et } 3 * u(n) + 1 \text{ sinon.}$$

partant de $u(0) = 42$. On s'arrête lorsque la valeur 1 est atteinte.

```

# la variable u contient le terme courant de la suite
u=42
echo $u
while [ $u -ne 1 ]; do
  if [ $((u%2)) -eq 0 ]; then
    u=$((u/2))
  else
    u=$((3*u + 1))
  fi
  echo $u
done

```

Exercice 5. Écrire un script qui convertit en décimal un nombre binaire contenu dans la variable \$binaire (initialiser par exemple binaire à 110110).

```

#Solution 1: on indique la base 2 dans laquelle est écrit le nombre
# contenu dans la variable binaire, on évalue en décimal avec la commande bc
binaire=110110
echo "ibase=2; $binaire" | bc

```

```

#Solution 2: calcul de la valeur décimale avec le schéma de Horner
#res = (((((2*0+1)*2 +1)*2+0)*2+1)*2+1)*2+0
binaire=110110
res=0
# calcule dans nbbits la longueur (nb de caractères) de $binaire
nbbits=$(( ( echo $binaire | wc -c ) -1 )) # ou nbbits=${#binaire}
for i in $( seq 1 $nbbits ); do
  bit=$( echo $binaire | cut -c $i )
  res=$(( 2*$res + $bit ))
done
echo $res

```

Exercice 6. Écrire un script qui met les 3 premières lignes des fichiers texte du répertoire haddock dans un fichier resume.txt. Le fichier resume.txt contiendra pour chaque fichier texte, son nom et ses 3 premières lignes.

```

#Solution 1:
# récupérer dans /tmp/temp les noms de fichiers ordinaires
find haddock -type f -print > /tmp/temp
# taille est le nombre de ces fichiers
taille=$( wc -l < /tmp/temp )

for i in $(seq 1 $taille); do
  # on met dans ligne le ième nom de fichier
  cat -n /tmp/temp | head -n $i | tail -n 1 | cut -f2 > ligne

```

```

# fnom contient le nom du fichier sans le préfixe "haddock/"
cat ligne | cut -d '/' -f 2 > fnom
# on ajoute ce nom au fichier resume.txt
cat fnom >> resume.txt
# de nouveau, le nom du fichier est complet (avec le préfixe haddock/)
fnom=$( cat ligne)
# # on ajoute les 3 premières lignes de ce fichier
cat -n "$fnom" | head -n 3 >> resume.txt
done

```

```

#Solution 2:
rep="haddock"
for nom in $(ls $rep); do
# le 1er champ donne les droits d'accès
droits=$( ls -al "$rep/$nom" | tr -s ' ' | cut -d ' ' -f1)
# si le 1er caractère de ce champ est '-', il s'agit d'un fichier ordinaire
if [ $(echo $droits | cut -c 1) = "-" ]; then
# écrit le nom du fichier
echo "$nom" >> resume.txt
# écrit ses 3 lères lignes
cat -n "$rep/$nom" | head -n 3 >> resume.txt
fi
done

```

Exercice 7. Écrire un script qui affiche à l'écran toutes les lignes paires du fichier `/etc/passwd`.

```

# taille est la taille du fichier /etc/passwd
taille=$( wc -l < /etc/passwd)

for i in $(seq 1 $taille); do
nb=$(cat -n /etc/passwd | head -n $i | tail -n 1 | cut -f 1)
# teste si la ligne est paire
if [ $($nb % 2) -eq 0 ]; then
cat -n /etc/passwd | head -n $i | tail -n 1 | cut -f 2
fi
done

```

Exercice 8. Écrire un script qui affiche pour chacun des fichiers texte du répertoire `data/haddock` les informations : nom, nombre de lignes, nombre de caractères, le nom du propriétaire.

```

Solution 1:
# si on est placé dans le répertoire data,
# sinon, si vous êtes placés dans votre home directory,
# indiquer le chemin dataTP3/haddock
rep="haddock"
for nom in $(ls $rep); do
droits=$( ls -l "$rep/$nom" | tr -s ' ' | cut -d ' ' -f1)
if [ $(echo $droits | cut -c 1) = '-' ]; then
echo "fichier : $nom"
echo "nb de lignes : $( cat "$rep/$nom" | wc -l)"
echo "nb de caracteres : $( cat "$rep/$nom" | wc -c)"
prop=$( ls -al "$rep/$nom" | tr -s ' ' | cut -d ' ' -f3)
echo "proprietaire : $prop"
fi
done

```

Exercice 9. Écrire un script qui affiche des informations de chaque fichier du répertoire `data/haddock` dans un langage plus convivial, par exemple "Le propriétaire a les droits : lire, écrire". Indication : utiliser `ls -l`, `cut -f` et `cut -c`.

```
rep="haddock"
for nom in $( ls $rep); do
  echo "fichier: $nom"
  chaine="Le proprietaire a les droits : "
  droits=$( ls -l "$rep/$nom" | tr -s ' ' | cut -d ' ' -f1)
  if [ $( echo $droits | cut -c2) != '-' ]; then
    chaine=$chaine"lire"
  fi
  if [ $( echo $droits | cut -c3) != '-' ]; then
    chaine=$chaine", ecrire"
  fi
  if [ $( echo $droits | cut -c4) != '-' ]; then
    chaine=$chaine", executer"
  fi
  echo $chaine
done
```