

Ce projet se déroulera sur les deux prochaines séances. Il demande un travail personnel pendant le TP et en dehors du TP. A une date fixée par votre encadrant de TP, un compte rendu vous sera demandé par binôme. Ce dernier devra faire apparaître votre démarche pour résoudre le problème (pour permettre au correcteur de comprendre votre approche et de faciliter sa correction) ainsi que les résultats attendus.

## **PARTIE 1 : PREPARATION DU PROJET**

### **Exercice 1 : Scripts Shells (échauffement)**

**Q1.** Écrire un script « test » qui vérifie qu'il y a entre 3 et 10 arguments passés en paramètres et qui les affiche tous dans l'ordre et numérotés.

#### **Voici le code :**

```
#!/bin/sh
if [ $# -gt 3 ]      # si le nombre de paramètres est supérieur a 3
then
    if [ $# -le 10 ] # si le nombre de paramètres est inférieur a 10
    then
        j=1
        for i in @$@ # i prend la valeur des paramètres
        do
            echo $j $i # écrit la valeur de j et la valeur du paramètre
            let "j+=1" # j=j+1
        done
    else echo trop de paramètres
    fi
else echo pas assez de paramètre
fi
```

**Q2.** Écrire un script qui renvoie la valeur de la variable d'environnement MONREP sauf si celle-ci est vide auquel cas il renvoie le répertoire courant.

**Voici le code :**

```
#!/bin/sh
if [ -z $MONREP ] #si MONREP est vide
then
  pwd #affiche le rep courant
else
  echo $MONREP #affiche la valeur de MONREP
fi
```

**Q3.** Modifier “à la main” la valeur de cette variable et tester le script précédent. Utiliser le mécanisme (la commande) de export pour comprendre son fonctionnement.

**Exercice 2 :** Commandes avancées pour le projet

Cet exercice à pour but de vous familiariser avec les commandes suivantes :

- grep,
- cut,
- sort,
- find,
- file,
- sleep,
- mp3info.

N'hésitez pas à consulter la documentation (man) !

**Q1.** Trouver et télécharger sur Internet (Google) le paquetage mp3info.rpm (plate-forme i586). L'installer.

Il n'y a pas de code a vous donner pour cette exercice.

**Q2.** Écrire un script qui classe tous les fichiers du répertoire courant selon leur type : un sous répertoire est créé pour chaque type rencontré et les fichiers sont déplacés dans le répertoire correspondant à leur type. Par convention, on considère que le type d'un fichier est le premier mot renvoyé par la commande file -b exécutée sur ce fichier.

```
#!/bin/bash
REP="$PWD"
for fichier in `ls`
do
  #si fichier n'est pas un dossier
  if [ ! -d "$fichier" ]
  then
```

```

echo $fichier
type=`file -b $fichier | cut -d" " -f1`
echo $type
#si le dossier n'existe pas
if [ ! -e $type ]
then
    mkdir $type
fi
cp "$fichier" "$type"
fi
done

```

Notre script consulte pour tous les fichiers dans le répertoire courant, son type et crée un dossier du nom du type si celui-ci n'existe pas. Ensuite il copie le fichier dans ce répertoire.

**Q3.** Écrire un script qui liste tous les fichiers du répertoire courant en les triant selon leur type.

## **PARTIE 2 : GESTION D'UNE BIBLIOTHEQUE DE MP3 (EN SHELL !)**

Le but de ce projet est d'écrire un ensemble de scripts qui permettent de gérer une bibliothèque de morceaux mp3. Pour cela, une bibliothèque vous est donnée, elle est disponible dans le groupe « Système d'Exploitation » de votre cartable. Cette dernière est dans une archive tgz. Il vous incombe donc de la décompresser dans un répertoire approprié.

**ATTENTION : pour tester cette bibliothèque, aucun fichier mp3 protégé par copyright ne doit être utilisé.**

Le projet comprend plusieurs objectifs détaillés ci-après :

**Objectif 1 :** Le lecteur

**Q1.** Créer un script permettant de lire un mp3, c'est-à-dire affichage des métas informations tel que le genre, nom artiste, etc., puis de simuler une lecture du fichier (attente de 10% de la durée en secondes du morceau).

```

#!/bin/bash
for i in $@
do
    type=`file -b $i | cut -d"," -f1`
    ext=${i##*.}
    if [ "$ext" = "mp3" ]
    then
        artiste=`mp3info -p %a $i`
        genre=`mp3info -p %g $i`
    fi
done

```

*#pour la question2 on prend tous les paramètres*

*#recupere l'extension du fichier*

*#on récupère le nom de l'artiste*

*#on récupère le genre*

```

        echo "titre " `mp3info -p %f $i`      # on affiche le titre
        echo "artiste :" $artiste           #on affiche l'artiste
        echo "genre :" $genre              #on affiche le genre
        echo "duree :" `mp3info -p %s $i`  #on affiche la durée
        duree=`mp3info -p %s $i`
        dureetemp=$duree*0.10
        #sleep $dureetemp                  #fait dormir le tps de la chanson
    else
        echo "pas un fichier mp3"
    fi
done

```

**Q2.** Modifier le script pour qu'il puisse prendre en paramètres un ou plusieurs fichiers mp3.

voir question 1

**Q3.** Modifier le script pour qu'il puisse prendre en compte un ou plusieurs fichiers m3u.

```

#!/bin/bash
for j in $(cat $1) #parcours toutes les lignes du fichier passé en paramètre
do
    ./Partie2question1buber $j #exécute le programme avec le fichier $j
done

```

**Rappel :** Le format Playlist m3u contient une ligne par fichier à lire, la ligne consiste en le chemin absolu du fichier Modifier le script afin qu'il puisse marquer dans un fichier de log les musiques jouées.

**Objectif 2 :** La médiathèque

Définissez le dossier à prendre en compte comme dossier racine de la médiathèque au moyen d'une variable. A partir de ce dossier, la médiathèque prendra en compte, et ce, de manière récursive (dossiers et sous-dossiers etc.) tous les fichiers portant le format mp3. Par ailleurs, la médiathèque permettra d'afficher de manière numérotée tous les mp3 présent dans les dossiers contenus dans le dossier racine.

**Q1.** Ecrivez ces fonctionnalités.

```

#!/bin/bash
IFS=$'\n'          #remplace les espaces
i=0
echo "entrez le repertoire racine :"
read racine
chose=`find $racine -name "*.mp3"` #chose est égal à tous les noms de fichiers finissant par
.mp3
for fichier in $chose

```

```
do
  echo $i $fichier      #On affiche le numéro du fichier mp3 ainsi que son nom
let i=$i+1
done
```

**Q2.** Modifiez le script de la médiathèque afin de prendre en compte une commande de tri, c'est-à-dire trier par titre, par album, par artiste, par année et par genre.

```
#!/bin/bash
echo "" >info_mp3.txt      #initialise info_mp3.txt
IFS=$'\n'
echo "entre le mode de tri"
read choix
echo "entrez le repertoire racine :"
read racine
CPTE=0
VAL=""
chose=`find $racine -name "*.mp3"` #chose est égal a tous les noms de fichiers finissant par .mp3

if [ "$choix" = "artiste" ]
then
    #on parcourt tous les fichiers de racine
    for fichier in $chose #fichier prend toutes les valeurs de $chose
    do
        #on récupère l'artiste du fichier
        artiste=`mp3info -p %a $fichier`
        echo "$artiste;$fichier" >> info_mp3.txt #met le nom de $fichier et $fichier
    dans le fichier info_mp3.txt
    done
else if [ "$choix" = "genre" ]
then
    #on parcourt tous les fichiers de racine
    for fichier in $chose #fichier prend toutes les valeurs de $chose
    do
        #on récupère le genre du fichier
        genre=`mp3info -p %g $fichier`
        echo "$genre;$fichier" >> info_mp3.txt #met le nom de $fichier et $fichier
    dans le fichier info_mp3.txt
    done
else if [ "$choix" = "album" ]
then
    #on parcourt tous les fichiers de racine
    for fichier in $chose #fichier prend toutes les valeur de $chose
    do
        #on récupère le nom de l'album du fichier
        album=`mp3info -p %1 $fichier`
        echo "$album;$fichier" >> info_mp3.txt #met le nom de $fichier et $fichier
    dans le fichier info_mp3.txt
    done
else if [ "$choix" = "titre" ]
then
```

```

#on parcourt tous les fichiers de racine
for fichier in $chose #le fichier prend toutes les valeurs de $chose
do
    #on récupère l'année du fichier
    titre=`mp3info -p %t $fichier`
    echo "$titre;$fichier" >> info_mp3.txt #met le nom de $fichier et $titre
dans le fichier info_mp3.txt
done
else
#on parcourt tous les fichiers de racine
for fichier in $chose #le fichier prend toutes les valeurs de $chose
do
    #on récupère l'année du fichier
    annee=`mp3info -p %y $fichier`
    echo "$annee;$fichier" >> info_mp3.txt #met le nom de $fichier et $annee
dans le fichier info_mp3.txt
done
fi
fi
fi
fi
#on parcourt tous les fichiers info_mp3.txt ligne par ligne
for i in $(cat ./info_mp3.txt)
do
    LIGNE=$i

#Nous faisons l'affichage
if [ ! -z $LIGNE ]
then
    echo $LIGNE | cut -d';' -f2
fi

fich=`echo $i | cut -f2 -d";"`
echo "$fich" >> fichier_trier.txt
done

```

**Q3.** Ajoutez une fonctionnalité au tri permettant d'indiquer au moyen d'un paramètre -r (passé au script) que le tri s'effectue de manière décroissante au lieu d'être effectué de manière croissante.

Pour pouvoir choisir dans quel sens on tri le fichier, nous avons rajouté :

```

echo "voulez vous trier en mode inverser ? 0/1 " #On déclare les variables
read INVERSE

```

```

.....
.....
.....
if [ $INVERSE -eq 1 ]
then
    sort -o info_mp3.txt -r info_mp3.txt #tri dans le sens inverse
else
    sort -o info_mp3.txt info_mp3.txt #tri du fichier
fi

```

**Q4.** Modifiez le script afin de pouvoir filtrer les fichiers musicaux en fonction des méta-informations qu'ils contiennent. Par exemple, pour obtenir toutes les chansons de rock ou hard rock etc., il faudra utiliser un paramètre -f avec la commande suivante genre= « rock » | « hard-rock »

pour cette question on demande a l'utilisateur de rentré son filtre suivant le tri qu'il veut c'est a dire s'il veut trier ses fichiers par genre son filtre devra etre un genre.

```

echo "entrez le mode de tri"
read choix
echo "entrez votre type "
    read filtre
...
... (idem question precedentes)
...
#on parcourt tous le fichier info_mp3.txt ligne par ligne
for i in $(cat ./info_mp3.txt)
do
LIGNE=$i # Nous selectinons que ce qui est necessaire
DEBUT=`echo "$LIGNE" | cut -f1 -d";"`
    if [ "$DEBUT" = "$FILTRE" ]
        # si notre debut est different du filtre
then
fich=`echo $LIGNE | cut -f2 -d";"`
echo "$fich" >> fichier_trier.txt
fi
    #Nous faisons l'affichage des lignes que l'on veut
aff=echo `"$LIGNE" | cut -d';' -f2`

```

```
echo $aff done
```

**Q6.** Faites en sorte de pouvoir utiliser l'étoile (\*) dans les filtres.

Nous n'avons pas réussi à réaliser ce script.

**Q7.** Ajoutez un paramètre a la médiathèque permettant de jouer dans le lecteur en fonction des données fournies le résultat du tri par la médiathèque. Par exemple pour jouer la chanson classée 1, puis la 3 puis celles de 6 à 12, on devra taper le paramètre 1 ; 3 ; 6-12

Pour cette question nous n'avons pas réussi à faire jouer les chansons que l'on veut. Nous sommes arrivé a faire jouer une suite de morceaux (6-12) mais pas 1 puis la 3 etc...

```
#!/bin/bash
echo "" > lecture.m3u
#on selectionne les fichiers
./tri_bis

echo "entrez les morceaux que vous voulez ecouter"
read morceaux
debut=`echo $morceaux | cut -f1 -d"-"` #on recupere le permeier morceau que
l'on veut ecouter
fin=`echo $morceaux | cut -f2 -d"-"` # on recupere le dernier

for j in $(cat ./fichier_trier.txt) #on parcours tout le fichier
do
    num=`echo $j | cut -f1 -d";"`
    if [ "$debut" != "$((fin+1))" ] #si on n'a pas lu tous les morceaux
    then
        if [ "$num" = "$debut" ] # si c'est le morceau que l'on veut
ecouter
        then
            fich=`echo $j | cut -f2 -d";"`
            echo $fich >> lecture.m3u
            let debut=$debut+1
        fi
    fi
done
./donne_m3u ./lecture.m3u #lit tous les mp3 du fichier
```

**Q8.** Remplacer la variable permettant de configurer le dossier racine par une portion de script permettant de lire dans un fichier de configuration quels sont les dossiers à prendre en compte par la médiathèque.

????

**Objectif 3 :** Le menu

On souhaite réaliser un menu pour faciliter l'accès aux fonctionnalités de notre médiathèque. Dorénavant, chaque fonctionnalité (actuelle, une variante ou future) devra apparaître dans ce menu.

**Q1.** Créez un menu interactif permettant de contrôler les fonctions du lecteur et de la médiathèque.

```
#!/bin/bash

echo "Bienvenue sur Buber mp3!!"
echo "1: liste des mp3"
echo "2: tri les fichier par..."
echo "3: affiche les info d'un fichier m3u"
echo "4: statistiques"
echo "0: Sortir"

read choix
while [ $choix -ne 0 ]
do
    case $choix in
        1) ./affiche_liste;;
        2) ./tri;;
        3) ./donne_m3u;;
        4) ./stat
        *) echo "Ce choix n'existe pas" ;;
    esac
done

echo "1: liste des mp3"
echo "2: tri les fichiers par..."
echo "3: affiche les info d'un fichier m3u"
echo "4: statistiques"
echo "0: Sortir"

read choix
done
```

Ce script permet de lancer les autres scripts par l'intermédiaire d'un menu.

#### **Objectif 4 :** L'analyseur de statistiques

L'analyseur manipule un fichier log (avec un nom arbitraire qu'il est possible de gérer à partir du fichier de configuration). A chaque simulation (lecture d'un morceau musical), une entrée dans le fichier log est rajoutée indiquant le morceau qui a été joué ainsi que toutes ces métas informations. Il est à noter que ces informations seront organisées à l'aide de délimiteurs (ex : espace, deux points, etc.).

**NB :** Numéroté les entrées sera peut être utile pour les prochaines questions.

Le but est ici de réaliser des statistiques en présupposant que plus un fichier est lu (plus il a d'entrées), plus il est intéressant.

**Q1.** Modifiez votre script qui simule la lecture d'un fichier MP3 pour qu'il rajoute une entrée dans le fichier de log à chaque fois qu'il accède à un fichier.

Dans le fichier `affiche_donne_mp3` on a rajouté :

```
...
...
if [ -e ./lecture.txt ] # si le fichier existe
then
    for k in $(cat ./lecture.txt) #on parcourt tout le fichier
    do
        nb=`echo $k | cut -f1 -d";"` #nb sera egal au dernier numero du
fichier
        done
    else
echo "" > lecture.txt # on initialise le fichier
    fi
if [ "$nb" = "" ] # si le fichier est vide, on met nb=1 pour le premier
morceau
then
    let nb=1
fi
let nb=$nb+1
echo "$nb;$artiste;$titre;$genre">> lecture.txt #on met dans le fichier les
info que l'on veut
```

Dans les prochaines questions, on supposera que l'analyseur est relativement indépendant du lecteur. De ce fait, vous pourrez vous même éditer votre fichier de log (avec pourquoi pas des morceaux qui n'existent pas dans la réalité) pour construire une base de test un peu plus étoffée.

**Q2.** Créez la partie analyse qui permettra d'afficher le pourcentage de lecture en fonction du genre de musique. (ex : 5% rock, 10% hard-rock, ).

Pour cette question l'utilisateur doit entrer le genre dont il veut les stats

```
#!/bin/bash
echo "entrez le genre dont vous voulez les stat"
```

```
read mon_genre
nb_genre=0
for i in $(cat ./lecture.txt)
do
total=`echo $i | cut -f1 -d";"` #total contient les nb total de morceaux qui
ont ete joué
genre=`echo $i | cut -f4 -d";"`
if [ "$genre" = "$mon_genre" ] #si c'est le genre que l'on veut
then
let nb_genre=$nb_genre+1
fi
done
let pourcent=($nb_genre/$total)*100 #on calcule
echo "il y a $pourcent % de de $mon_genre dans vos morceaux les plus joue"
```

**Objectif 6** : Tri qualitatif

On souhaite sélectionner le fichier le plus joué de chaque genre et le dupliquer dans un dossier ayant le même nom que le genre.

**Q1.** Créez cette fonctionnalité.

**NB** : Si le fichier n'existe pas (car vous avez fait de fausses entrées dans votre fichier de log) alors créez un fichier texte portant le même nom que ce dernier.