T.P.2 : Projet CPI 2015

P. Pittoli

1 Recherche de similarités

Faire un script qui prend 2 fichiers en paramètre, on doit trouver toutes les lignes en commun, avec 2 algorithmes :

— version naïve qui prend le premier fichier ligne par ligne et recherche la ligne dans le second
— version optimisée (indice : ne chercher qu'une fois !)
— optionnel : version très optimisée (je vous laisse chercher ;))

Le script doit accepter un paramètre (-v). Si ce paramètre est passé, alors on doit afficher les lignes en commun lorsqu'elles sont trouvées. Si ce paramètre n'est pas passé à l'application, on n'affiche rien, on se contente d'écrire dans un fichier le temps d'exécution.

Les temps d'exécution seront enregistrés dans un fichier, puis on en fera des graphes avec gnuplot (script fourni). Pour que gnuplot sache faire le graphe il faut lui donner un fichier avec les données brutes. Ce fichier devra ressembler à ceci :

```
taille1 temps1 temps2
taille2 temps1 temps2
taille3 temps1 temps2
```

Exemple :

```
1000000000 10 1
10000000000 1000 10
100000000000 100000 20
```

Avec :

— taille = la taille du second fichier, dans lequel on fait notre recherche (ce fichier changera)
— temps 1 = le temps pris par la méthode naïve
— temps 2 = le temps pris par la méthode optimisée

Les fichiers passés en paramètres seront de taille conséquente. À peu près 1 Mio pour le fichier de référence, et de de 1 Mio à 100Mio pour les autres. Au final, voici une exécution de programme possible :

```
./similarities smallfile bigfile
1000000000 10 2
```

Si gnuplot n'est pas installé sur vos ordinateurs personnels, il suffit de l'installer avec (sur Ubuntu et Debian) :

```
apt update && apt install gnuplot
```

2 La musique c'est bien !

Quand elle est partagée c'est mieux.

Jamendo.com, très beau site, de la musique gratuite et souvent libre. On peut y écouter de la musique sans limite, y découvrir des groupes sympa, et surtout on peut télécharger des albums pour les écouter à la maison. Problème, ces albums arrivent dans des fichiers .zip, avec un nom pas très propre.

Votre mission (que vous acceptez) sera de faire un script qui prend un répertoire en paramètre (là où vous faites vos téléchargements), et qui pour tous les albums que vous avez téléchargé vous crée un répertoire avec un nom correct et y dépose vos musiques.

Exemple, vous recevez le fichier « For The Broken - From Sinners To Sinners - 139563 — Jamendo - MP3.zip », vous allez créer le répertoire « For The Broken - From Sinners To Sinners ». Pour être plus explicite, le nom original contient « - 139563 — Jamendo - MP3.zip » en trop.

Un second problème est que les fichiers reçus ont parfois le droit d'exécution, ce qui n'est pas normal pour un fichier de musique. De plus, vous souhaitez que seuls vous et votre groupe puissiez lire votre musique, les autres n'ont pas le droit ne serait-ce que de voir les musiques que vous avez téléchargé (vos goûts musicaux ne regardent que vous !). Si vous voyez un autre changement de droit pertinent à faire, dites-le et faites-le.

N'hésitez pas à aller rechercher des exemples sur le site, comme LukHash, Aygan, Epic Soul Factory... autant d'exemples qu'il vous faudra pour « tester votre script » !

Si vous souhaitez aller plus loin que ce qui est demandé, n'hésitez pas ! Des points bonus à la clé ! ;)

3 Consignes supplémentaires

3.1 La lisibilité

Le code doit être lisible et commenté. Un code non indenté vaut 0. Un code non commenté vaut 0. Les commentaires doivent être pertinents, et montrer que vous comprenez ce que vous faites. Un saut de ligne de temps et en temps ne peut pas faire de mal.

Vous êtes libres d'organiser votre code comme bon vous semble, s'il vous paraît judicieux de séparer le code en plusieurs fichiers pour améliorer la lisibilité ou pour rendre votre code plus modulaire, n'hésitez pas !

Vous avez explicitement le droit de rendre vos sorties d'exécution plus jolies si vous le souhaitez, tant que cela ne nuit pas à la lisibilité du code ; par exemple en écrivant le nom du répertoire que vous allez créer dans l'exercice 2.

3.2 Un tout petit projet INDIVIDUEL

Un code qui ressemble d'un peu trop près à un autre vaut 0. Le projet est à faire individuellement. Je n'hésiterai pas à utiliser le code du premier exercice pour vérifier les copies. ;)

3.3 Trouver de l'aide

Les pages de manuel peuvent vous aider, de même que de nombreux sites. N'utilisez un moteur de recherche que si vous ne trouvez pas dans le manuel, vous en apprendrez plus ! ;)

Si vous avez des questions, n'hésitez pas à envoyer un mail !

3.4 Le rendu

Le rendu doit se faire dans une archive : `nom-prenom_projet-cpi.tar.gz`. Le barème sera décidé au doigt mouillé, inutile de me spam pour ça. Bon courage.