TD nº 5 - Création d'un album d'images en *shell*

Le but de ce TD est d'écrire un script prenant en argument un mot-clé qui va aller chercher des images correspondant au mot-clé sur un site web et les placer dans un album au format pdf.

1 Avant toute chose, créez un nouveau répertoire et placez-vous dans ce dernier pour éviter d'encombrer votre répertoire principal avec tous les fichiers que nous allons créer.

La première étape consiste à télécharger des images. On décide ici d'aller les chercher sur le site `www.freeimages.com`.

2 Dans un navigateur web, allez sur le site `www.freeimages.com` et effectuez la recherche d'un mot-clé (par exemple : bears). Observez l'*URL* qui s'affiche dans la barre d'adresse. Déduisez-en le format de la page qu'il faudra demander pour obtenir des images correspondant à un mot-clé quelconque.

La commande `wget` permet de télécharger un fichier en donnant l'addresse où le trouver. Par exemple la commande `wget www.unsite.com/index.html` téléchargera le fichier `index.html`. Il est possible de donner un nom spécifique au fichier ainsi créé avec l'option `-O nom_fichier`.

3 Lancez la commande `wget` permettant de récupérer dans un fichier nommé `index.html` la page de résultats du site `freeimages.com` pour la recherche correspondant au mot-clé "alligator".

4 Ouvrez le fichier téléchargé dans un éditeur de texte et observez son contenu. Faites une recherche du texte " `.jpg` " et observez la syntaxe qui permet d'insérer une image dans du HTML.
Indication : il faut chercher les balises spécifiques et la spécification du fichier source.

On va vouloir extraire de ce fichier les adresses de toutes les images affichées, pour pouvoir les télécharger toutes avec `wget` par la suite.

5 En utilisant la commande `grep` vue précédemment, affichez dans un terminal toutes les lignes du fichier `index.html` qui contiennent l'adresse d'une image.
Indication : Il faut rechercher un motif commençant par le motif " `src="` ", suivi d'un certain nombre de caractères qui ne sont pas des guillemets , suivi du motif " `jpg"` ".

Il va maintenant falloir extraire les adresses des images contenues dans ces lignes. Il est possible d'y arriver avec la commande `grep` en utilisant l'option `-o` qui n'affiche que le motif trouvé, mais on peut aussi utiliser la commande `sed` que nous avons déja étudié.

Rappel : la commande `sed` permet de manipuler un texte en entrée selon une série de règles à appliquer, ligne par ligne. Le programme renvoie alors le texte modifié selon les règles fournies. La syntaxe de base consiste à appeler `sed` en lui donnant comme argument une seule règle. Par exemple `sed s/toto/titi/` transforme la première occurrence de `toto` en `titi` dans chque ligne.

6 Puisque `sed` travaille sur l'entrée standard et la sortie standard, comment peut-on appeler le programme pour qu'il prenne en entrée un fichier `debut.txt` et enregistre le résultat dans un fichier `fin.txt` ?

La règle la plus utile de `sed` est la substitution : `s/motif_initial/motif_final/`. Les motifs indiqués sont des expressions régulières et peuvent donc être assez complexes.

7 Créez un fichier contenant des phrases dans lesquelles le mot "chien" apparaît plusieurs fois. Utilisez la commande `sed` pour changer les occurrences de "chien" en "chat". Que se passe-t-il lorsque le mot "chien" apparaît plusieurs fois dans une même ligne ?
Essayez de relancer la commande `sed` en ajoutant un `g` à la fin de la règle (après le dernier `/`).

Parfois, on veut réutiliser une partie du motif initial dans le motif final, sans savoir exactement la forme du motif initial trouvé avec des expressions régulières. On peut alors marquer des blocs dans le motif initial avec `\(` et `\)`. Les résultats correspondant à ces blocs peuvent ensuite être utilisés dans le motif final à l'aide des mots-clés `\1`, `\2`, etc.
Par exemple, la règle `s/ab\(.*\)cd/cd\1ab/` échange le premier `ab` d'une ligne avec le dernier `cd` sur cette même ligne (les expressions telles que " `.*` " remplacent toujours la plus longue chaîne possible).

8 Donnez une règle de `sed` permettant d'extraire d'une ligne du fichier `index.html` précédemment téléchargé l'adresse d'une image s'il y en a une (la substitution ne fait rien si le motif initial n'est pas trouvé dans la ligne). **Indication :** Il faut remplacer une ligne contenant `src="` , suivi d'un certain nombre de caractères et suivi du mot `.jpg` par ce qui se trouve entre les guillemets.

Une fois que l'on sait extraire les adresses des lignes les contenant, il y a deux façons (au moins) de faire pour ne garder que les adresses dans le fichier initial. La première consiste à utiliser `grep` pour ne garder que les lignes contenant une adresse et à lancer `sed` sur la sortie de `grep` pour ne garder que les adresses (on utilise une redirection de la sortie de `grep` vers l'entrée de `sed`).

La seconde méthode consiste à n'utiliser que `sed`, en ajoutant l'option `-n` qui indique au programme de ne pas afficher les lignes traîtées par défaut, et ajouter la lettre `p` à la fin de la règle de substitution pour demander au programme d'afficher le résultat des substitutions effectuées.

9 Donnez les deux lignes de commande permettant de ne garder que les adresses des images dans le fichier `recherche.html`, correspondant aux deux méthodes décrites précédemment.

10 Par la méthode de votre choix, créez un fichier `liste.txt` contenant la liste des adresses des images présentes sur la page `index.html` (une adresse par ligne).

11 Créez un répertoire nommé selon le mot clé de recherche (par exemple `alligator` ou `cat` ...le mot-clé que l'on avait cherché initialement sur le site) et placez-vous dans ce répertoire. Il est possible de dire à `wget` de récupérer les fichiers correspondant à une liste d'adresses enregistrées dans un fichier à l'aide de l'option `-i`. Après vous être placé dans le répertoire nouvellement créé, lancez la commande `wget` en utilisant l'option `-i` suivie du nom du fichier contenant les adresses à télécharger.
Remarque : Dans notre cas, il faut faire attention à bien indiquer où se trouve le fichier `liste.txt` puisqu'il n'est pas dans le répertoire dans lequel on se trouve pour lancer la commande `wget`.

Normalement, si tout se passe bien, vous devriez maintenant avoir un répertoire contenant toutes les images qui se trouvaient sur la page de résultats du site `www.freeimages.com`.

Nous sommes presque au bout de notre périple. Il reste maintenant à mettre les images dans un fichier *pdf*. Pour cela, on va utiliser la commande `convert` qui permet d'effectuer de nombreuses manipulations sur les images.

12 Allez voir la page `http://www.imagemagick.org/script/convert.php` et soyez ébahis par le nombre d'opérations que l'on peut exécuter en ligne de commande avec `convert`

Dans sa forme la plus simple, `convert` permet de convertir des images d'un format dans un autre. Ainsi, la commande `convert image.jpg image.png` convertira automatiquement l'image initiale `image.jpg` au format PNG (du coup il faut faire attention parce que les extensions des fichiers en sortie sont importantes pour `convert`).

C'est cette commande que l'on va utiliser pour créer un album PDF, en donnant tout simplement une liste de fichiers en entrée et un nom de fichier PDF en sortie.

13 Si vous êtes toujours dans le dossier contenant toutes les images récupérées précédemment, comment peut-on désigner pour le shell l'ensemble des fichiers dont le nom termine par `.jpg` ?

14 Essayez d'utiliser la commande `convert` pour transformer tous les fichiers dont le nom termine par `.jpg` en un fichier PDF. Ouvrez le résultat et vérifiez que la manipulation a fonctionné.

Si vous avez bien fait la question précédente, vous devriez maintenant avoir un fichier PDF contenant les images qui ont été prises sur le site de départ. Cependant ces images ont des tailles différentes, ce qui est moche dans un PDF parce que les pages ont toutes des formats différents...

Nous allons donc utiliser `convert` pour mettre toutes les images au même format. Une rapide observation des dimensions des images récupérées permet de voir que la largeur maximale d'une image est de 300 pixels et la hauteur maximale de 200 pixels. Nous allons donc agrandir toutes les images pour qu'elles fassent exactement 300×200 pixels.

La commande `convert` permet cette opération. En fait il y a même de nombreuses façons d'agrandir une image, selon ce que l'on veut obtenir comme résultat. Dans notre cas, on ne veut pas déformer les images (on veut conserver les proportions sur le dessin), mais simplement rajouter des pixels noirs pour que les dimensions de l'image augmentent jusqu'à la taille voulue.

La commande que l'on veut est

```
> convert image.jpg -background black -gravity center -extent 300x200
image-bis.jpg
```

Dans cette commande, `gravity` permet de décrire où l'on veut ajouter les nouveaux pixels (ici, tout autour), `background` définit la couleur des pixels à ajouter et `extent` indique que l'on veut augmenter les dimensions de l'image sans modifier les pixels existants.

On veut maintenant appliquer cette commande à tous les fichiers JPG se trouvant dans le répertoire. Pour celà on utilise une syntaxe de *bash*[1] qui permet d'exécuter une commande de manière répétée sur plusieurs noms de fichiers. C'est la boucle `for; do; done` dont voici un exemple :

```
for i in toto*; do mv $i ${i/toto/titi}; done
```

Cette ligne indique que l'on veut exécuter la commande `mv` pour chaque nom de fichier de la forme `toto*`. Le premier argument de la commande est le nom du fichier que l'on est en train de traiter (`$i` désigne la valeur de la variable `i`), tandis que le second argument est le nom du fichier traîté dans lequel `toto` a été remplacé par `titi` (cette syntaxe est inspirée de la règle de substitution de `sed` mais en beaucoup plus simple puisqu'on ne peut pas utiliser d'expressions rationnelles ou d'options).

15 En vous inspirant de l'exemple de boucle `for` donné, écrivez une ligne en *bash* permettant de transformer toutes les images du répertoire en une image de 300×200 pixels dont le nom est le nom de l'image initiale auquel on a ajouté `-bis` avant l'extension.

16 Créez un fichier PDF contenant toutes les images redimensionnées.

17 Enfin, écrivez un script shell qui prend en argument un mot-clé et fabrique un fichier PDF dont le nom est le mot-clé indiqué suivi de l'extension `.pdf` contenant les images obtenues en recherchant le mot-clé sur le site `www.freeimages.com` redimensionnées pour avoir toutes la même taille.

1. Les boucles `for` et la syntaxe de substitution ne sont pas communes à tous les shell. Les exemples présentés ici sont spécifiques à *bash*