TP n°4 – filtres, redirections et tubes

Ce TP propose de découvrir une application des redirections et des tubes au traitement d'images, mais en commençant d'abord par les filtres simples sur des textes. Vous pourrez approfondir les traitements d'image en dehors des TP. Attention, le but du TP n'est pas d'apprendre à taper le plus vite possible, mais de comprendre les mécanismes. Vous pourrez rédiger les réponses importantes sur votre wiki.

1) Étude des filtres usuels

Taper le texte suivant dans un fichier appelé `fruits` (attention à ne pas rajouter d'espaces ni de ligne vide). Pour aller plus vite, vous pouvez le copier-coller de ce pdf. Les colonnes sont le nom du fruit, la quantité, la couleur, le prix et la provenance.

```
pomme:2:rouge:1,35:nzelande
cerise:6:rouge:3,25:france
pomme:3:jaune:2,25:bretagne
peche:4:jaune:1,70:espagne
peche:5:blanche:1,60:france
banane:9:jaune:1,15:guadeloupe
pomme:4:jaune:1,10:france
```

Voici un autre fichier à saisir, `langages`, alors attention entre les colonnes, il y a un certain nombre d'espaces (pas des tabulations) de manière à bien aligner les colonnes, de plus certaines colonnes contiennent plusieurs mots, vous essaierez de reproduire ce format ou alors copiez-collez si c'est possible. Les colonnes sont le nom, la date de naissance du langage, le type et la machine qui permet d'exécuter les programmes.

```
C++        1983  compilé    code natif
Forth      1978  compilé    machine virtuelle
Java       1995  compilé    machine virtuelle
PHP        1994  interprété machine virtuelle
Javascript 1995  interprété machine virtuelle
Pascal     1970  compilé    code natif
Ada        1983  compilé    code natif
Python     1990  interprété machine virtuelle
```

Dans tout ce qui suit, on demande d'écrire les réponses aux questions dans le wiki.

a) Sélection de colonnes : commandes colrm et cut

Utiliser la commande `cut` pour n'afficher que les noms des fruits : faites `cut -d: -f1 < fruits`. La commande pour n'afficher que le nom et la couleur est : `cut -d: -f1,3 < fruits`. Quelle est la commande pour afficher le nom et la provenance uniquement ? Quelle est la commande pour n'afficher que le nom, la quantité et la couleur ? Au lieu de mettre une liste comme `2,3,4` , on peut mettre un intervalle : `2-4`.

Utiliser la commande `colrm` pour n'afficher que les noms des langages de programmation présents dans le fichier langages : cela revient à supprimer tout ce qui est au delà d'une certaine position sur chaque ligne. Quand on compte les caractères sur la ligne, on voit que les noms des langages tiennent sur 10 caractères. Il suffit de faire ceci si vos colonnes sont correctement alignées : `colrm 12 90 < langages`. On peut enlever le 2e nombre, et dans ce cas, cela signifie de supprimer jusqu'au bout de la ligne. Inversement, on peut éliminer seulement le nom : `colrm 1 12 < langages`. Faire de même pour ne montrer que le nom du langage et sa date de naissance. Comment faire pour afficher uniquement le nom du langage et son type (compilé) ?

Peut-on utiliser la commande `colrm` sur le fichier fruits ? Pourquoi ? Essayez de le faire pour voir.

Peut-on utiliser la commande `cut` sur le fichier langages ? Pourquoi ? Essayez de le faire pour voir.

b) Sélection de lignes : commande head, tail et grep

Utiliser la commande `head` pour n'afficher que les 3 premières lignes du fichier des fruits : faites `head -n 3 < fruits`. Utiliser la commande `tail` pour n'afficher que les 2 dernières lignes des langages : faites `tail -n 2 < langages`. Sauriez-vous utiliser ces deux commandes pour n'afficher que les lignes 3 et 4 du fichier des fruits ? Essayez de modifier cette commande, de déplacer la redirection pour voir... Sauriez-vous afficher les lignes 3 à 5 du fichier langages ?

Utiliser la commande `egrep` pour n'afficher que les bananes : faites `egrep 'banane' < fruits`. Sauriez-vous afficher uniquement les langages compilés ? Sauriez-vous afficher les fruits dont la quantité est exactement 4 ? Et ceux dont la quantité est exactement 5 ? L'astuce consiste à inclure les séparateurs à droite et à gauche dans la recherche. Sauriez-vous afficher les langages compilés qui tournent sur une machine virtuelle ?

Alors maintenant, si on veut afficher les langages nés dans les années 1970 (il y en a 2), il faut utiliser un joker : `egrep '197[0-9]' < langages`. Sauriez-vous afficher les fruits dont le prix est compris entre 1 et 1,5 ? Et ceux dont le prix est supérieur ou égal à 2 ? Même question avec ceux présents en 3 ou 5 exemplaires.

Utiliser la commande `egrep` pour afficher les fruits qui ne sont pas rouges : faites `egrep -v 'rouge' < fruits`. Sauriez-vous afficher les fruits qui ne viennent pas de France et qui ne sont pas jaunes ? Et les fruits dont le prix est inférieur à 2 et la quantité est au moins égale à 4 ?

c) Classements

Utiliser la commande `sort` pour classer les fruits par prix croissant fruits : faites `sort -t: -k4 -n < fruits`. Relire le cours pour savoir ce que signifient ces options. Sauriez-vous classer les langages par ordre d'année de naissance [dans ce cas ne mettez pas l'option -t qui perturbe sort, car le séparateur par défaut est déjà une suite d'espaces] ? Classez les langages par ordre alphabétique des noms. Classez les fruits par provenance (ne pas mettre l'option -n puisque ce ne sont alors pas des nombres).

Sauriez-vous combiner trois commandes précédentes pour afficher le nom du plus ancien langage présent dans ce fichier ? Sauriez-vous afficher le nom du fruit le plus cher ?

d) Comptage

Utiliser la commande `wc` pour compter le nombre de fruits : faites `wc -l < fruits`. Sauriez-vous compter les lignes concernant les pêches ? Sauriez-vous compter les langages de type compilés ?

Utiliser la commande `uniq` pour compter le nombre de variétés de fruits différentes : utilisez d'abord `cut` pour ne garder que les variétés (le nom) des fruits. Puis rajoutez une commande qui va classer ces noms. Puis rajoutez uniq pour n'avoir plus aucun doublon. Enfin rajoutez la commande de comptage précédente. Ouf, il y a bien 4 variétés différentes ? À chaque fois, il faut vérifier que les commandes fournissent le bon résultat en regardant dans le fichier (tant qu'il n'est pas trop gros).

e) Divers

Utiliser la commande `tr` pour remplacer les `:` par des espaces dans le fichier fruits : faites `tr ':' ' '< fruits`.Utiliser la commande `tr` pour compacter les espaces dans le fichier langages : faites `tr -s ' '< langages`. Remarquez que du coup, on peut utiliser la commande cut pour faire certaines coupures dans ce fichier. Enfin, supprimer tous les chiffres du fichier fruits en faisant : `tr -d '0123456789' < fruits`.

2) Applications plus ou moins utiles

Construire les tubes suivants en utilisant les indications fournies et en vérifiant pas à pas.

a) Inverser un fichier

On veut inverser de haut en bas un fichier (par exemple `fruits`). L'idée est de numéroter les lignes puis de les trier dans l'ordre décroissant puis d'enlever les numéros. Voici les commandes :
• la commande `cat -n fruits` affiche les lignes de `fruits` avec des numéros.
• `sort -nr` trie les lignes dans l'ordre inverse des nombres,
• `colrm 1 8` élimine les 8 premiers caractères des lignes.

b) Trouver le nombre de processus d'une personne

On veut connaître le nombre de processus de chaque utilisateur. Utiliser :
• `ps -edf` pour obtenir la liste,
• `colrm 9` pour ne garder que les noms,
• `sort` pour la trier,
• `uniq -c` pour compter les occurrences.

c) Trouver les personnes connectées plusieurs fois

On veut connaître le nombre de connexions de chaque utilisateur. Utiliser :
• `who` pour obtenir la liste,
• `colrm 9` pour ne garder que les noms,
• `sort` pour la trier,
• `uniq -c` pour compter les occurrences identiques successives,
• `egrep -v 1` pour garder les lignes qui ne contiennent pas de 1 (en espérant que...)

3) Ensemble PPM

Le système Unix offre en standard un ensemble d'outils pour manipuler des images. Ces outils font appel à un format de données commun : le format PNM (portable any map) qui se décline en plusieurs variantes : le format PPM (pixmap) représente des images en couleur, le format PGM (graymap) représente des images en niveaux de gris, le PBM (bitmap) représente des images en noir ou blanc.

Tous ces formats de fichiers sont représentés assez simplement. Le but du TP n'est pas d'étudier ce format ; pour les détails, se reporter à man ppm, man pgm, man pbm, man pnm. Il faut seulement savoir que ces fichiers commencent par les caractères 'P1', 'P2' ou 'P6' suivis des dimensions de l'image et ensuite les couleurs des pixels. Cette simplicité a pour conséquence une taille énorme pour ces fichiers car les images ne sont pas compressées. Il n'est donc pas recommandé de stocker ces fichiers. On s'en sert uniquement pour faire des traitements.

A chaque format d'image correspond un jeu de commandes. Le préfixe des commandes indique ce qu'elles acceptent. Par exemple, pour les ppm, on trouve ppmrelief qui ne peut prendre que des images ppm ; pour les pgm, on trouve pgmnorm qui n'accepte que des images pgm… Les commandes pnm* acceptent tous les formats d'image PPM, PGM, PBM.

Toutes les commandes P*M sont des filtres : sur l'entrée standard on fournit une image, sur la sortie standard on retrouve cette image transformée. L'idée est de rediriger les entrées et les sorties vers des fichiers ou d'autres filtres ppm, p.ex. : `pnmscale -reduce 5 < macareux1.ppm > petitmacareux.ppm`

> NB : en absence de redirection de sortie, les données binaires de l'image sont envoyées à l'écran et le font planter (bip bip + caractères illisibles). Taper `reset` en aveugle pour réinitialiser l'écran.

Le répertoire `/usr/local/ano*/SYS/1A/tp4` contient trois images jpg à recopier chez vous pour faire les manipulations de ce TP. Essayer de les afficher par ces trois variantes de commandes.

a) Mode d'emploi général

✍ *commande options < image_d'entrée > image_de_sortie*

Les options sont spécifiques à chaque commande, il faut consulter leur documentation, p.ex : man pnmscale.

Exemple :

✍ `ppmrelief < macareux.ppm > macareux_relief.ppm`

Par défaut, si on ne spécifie pas de fichier de sortie, c'est à l'écran que sortent les données de l'image (nombreuses et pas lisibles). La commande display permet d'afficher un fichier image à l'écran, quelque soit son format. On a le choix, on peut passer le fichier par redirection, par un tube ou par paramètre :

- ✍ `display macareux.ppm`
- ✍ `display < macareux.ppm`
- ✍ `more macareux.ppm | display`

b) Décodage/codage d'une image

- ✍ `anytopnm fichier > ...`
- ✍ `anytopnm < fichier > ...`
- ✍ `...| anytopnm | ...`

La commande anytopnm transforme une image quelconque vers le format ppm ; on peut fournir l'image soit par redirection d'entrée soit par paramètre. Inversement, on peut mettre une image ppm aux formats png, gif, jpeg par pnmtopng, ppmtogif, ppmtojpeg…

Dans tous les cas, l'image résultante sort sur la sortie standard. Il faut donc la rediriger vers un fichier :

- ✍ `anytopnm < macareux.jpg > macareux.ppm`

- Transformez au format PPM les images JPG fournies en vous servant de la commande anytopnm redirigée correctement en entrée et en sortie. Afficher simultanément les paires d'images (macareux1.jpg et macareux1.ppm) pour vérifier leur similarité (utiliser le & pour mettre en arrière-plan). Comparer les tailles des fichiers jpg et ppm : regardez la 5e colonne de `ls -l macareux1.*`
- Transformez macareux1.ppm en macareux1.png par la commande pnmtopng (attention à ne pas faire de fatue de frappe) redirigée comme il faut. Afficher l'image png pour vérifier. Quelle taille fait-elle ? Pour finir, effacer tous les fichiers ppm et png.
- Avec un tube, sans aucun fichier intermédiaire, faites-en sorte de transcoder macareux.jpg directement en macareux.png. Vérifier à nouveau la transformation.

Constatez qu'on a le choix, utiliser des fichiers temporaires ou bien une seule commande avec des tubes :

version avec des fichiers intermédiaires	version tube sans aucun fichier intermédiaire

`cmd1 < image > f1`
`cmd2 < f1 > f2`
`cmd3 < f2 > f3`
`cmd4 < f3 > res`
`rm f1 f2 f3`
	`cmd1 < image | cmd2 | cmd3 | cmd4 > res`

Que préférez-vous : une seule commande ou plein de commandes avec des fichiers intermédiaires ?

Par contre, qu'est-ce qui semble le plus facile à mettre au point quand on n'est pas sûr du résultat ?

c) Altération d'une image

Il existe de très nombreuses commandes de modification d'image : `ls /usr/bin/*p[pngb]m*` (remarquer les jokers). Dans ce TP, on se contentera des commandes suivantes avec ces options :

`pnmscale` *`facteur`* permet d'agrandir ou rétrécir une image selon le facteur (nbre réel)

`pnmscale -w` *`largeur`* `-h` *`hauteur`* permet d'amener une image à la taille indiquée

`ppmlabel -text` *`'texte'`* ajoute le texte en surimpression.

`pnmgamma` *`coefficient`* modifie le contraste de l'image selon le coefficient (nbre réel).

`pnmrotate` *`angle`* fait une rotation de l'image selon l'angle indiqué

`pnmsmooth -s` *`coef coef`* permet de lisser une image pour enlever les escaliers, on l'utilise en général pour améliorer le résultat d'un agrandissement. Les coef indiquent l'importance du lissage, il faut que ce soient deux nombres entiers impairs (3 5 7…).

`ppmrelief` met en évidence les contours de l'image

Quand une commande ne reçoit pas un fichier correct, elle affiche une erreur : bad magic number. Si le résultat est incorrect, display affiche un magicien. Ça se produit quand une précédente commande a échoué ou qu'on fournit un mauvais format de fichier en entrée.

- Commencer par décompresser macareux1.jpg pour obtenir m1.ppm. Ensuite, réduire la taille de m1.ppm à 128x96 en produisant m2.ppm. Ensuite, ajouter le nom de l’image à m2.ppm pour obtenir m3.ppm. Enfin compresser m3.ppm en png pour obtenir macareux1.png. Cette dernière est l’imagette du fichier initial. Effacer les fichiers intermédiaires.
- Effectuer toutes ces opérations en une seule commande grâce à un tube. Écrivez-le dans le wiki.
- Par un tube, modifier le contraste (coefficient 2) de macareux1.jpg et le ré-enregistrer dans macareux1.png. Faire de même avec macareux2. Éventuellement, refaites les imagettes avec ce nouveau contraste.
- Par un tube, reprendre macareux1.png (l’imagette), l’agrandir 4 fois, la lisser avec un coef de 5, l’afficher.

d) Vous avez fini ?

Chercher dans la documentation comment faire d’autres modifications sur l’image, changer les couleurs, faire une page d’index… Prenez note ce que vous avez découvert d’intéressant.

Serait-il difficile conceptuellement de créer un ensemble de commandes à mettre en tubes pour traiter les fichiers audio : convertir des wma en mp3, augmenter le volume, jouer un son sur le haut-parleur, concaténer deux sons, capturer un son venant du micro… Comment ces commandes pourraient-elles se présenter ? Quels sont les traitements qui seraient impossible à faire à l’aide de tubes et de redirections ? Ces traitements sont alors qualifiés de « non linéaires », pourquoi ?

Et la vidéo ? Est-ce que ça se prête à de tels traitements ? N’oubliez pas qu’il y a un flux audio et un flux d’images animées. En général, les traitements sont différents sur ces deux flux. On trouve des éditeurs de vidéo non linéaires sur internet, regardez par exemple gstreamer et avisynth (il n'est plus maintenu mais ses concepts sont très intéressants).