Initiation à l'automatisation des tâches (2I012)
TME Solo – Durée: 1 heure

Thibaut Verron

26 février 2015

Documents autorisés

Tous les documents sont autorisés.

Fichiers à récupérer

Vous trouverez des fichiers d'exemple dans le répertoire `/Infos/lmd/2014/licence/ue/2i012-2015fev/tmesolo/Groupe3` Aucun des scripts demandés ne vous demandera de modifier ces fichiers, vous pouvez donc simplement travailler en déclarant le chemin avec `export DONNEES="/Infos/lmd/2014/licence/ue/2i012-2015fev/tmesolo/Groupe3"` et en utilisant ensuite `$DONNEES/fichier` pour pointer vers les fichiers.

Écriture de vos scripts

Vous pouvez composer vos scripts où vous voulez. Nous vous conseillons cependant de les placer dans un même dossier, par exemple `~/TME_solo`, puis de placer ce dossier dans le path avec `PATH=~/TME_solo:$PATH`.

Les scripts que vous aurez à écrire ne doivent pas dépendre du répertoire où ils sont placés ; ils ne doivent pas contenir de référence à des chemins absolus et doivent pouvoir être exécutés sans problème depuis tout répertoire.

N'oubliez pas de rendre vos scripts exécutables par `chmod +x` *`script`* et en mettant le shebang fonctionnel `#!/bin/bash` en début de fichier. N'oubliez pas que sous `emacs`, en mode `shell-script` (activé avec `M-x sh-mode`), vous pouvez réaliser ces deux opérations simplement avec la commande `C-c :`

Aucun de vos scripts ne doit laisser après son exécution de fichier non prévu.

Conseils généraux

Les questions sont indépendantes, n'hésitez pas à commencer par ce qui vous paraîtra le plus facile.

N'oubliez pas de tester vos scripts !

Vous avez des fichiers d'exemple à votre disposition, vous avez des exemples textuels dans le sujet que vous pouvez copier/coller (le sujet est disponible au format `.pdf` dans le dossier des exemples), et vous pouvez si ça ne suffit pas créer vos propres exemples de toutes pièces.

Si votre script ne fonctionne pas, pensez à l'option `-x` de `bash` qui permet de tracer l'exécution du script (préfixez votre script par `bash -x`).

De même si vous voulez savoir si un script est syntaxiquement correct avant d'essayer de l'exécuter, pensez à l'option `-n` de `bash` : `bash -n` *`script`* vous affiche les erreurs de syntaxe de *`script`* sans l'exécuter et le code de retour de cette instruction est nul si et seulement si *`script`* est syntaxiquement correct.

Vous trouverez dans le fichier d'exemple un script `test.sh` qui applique les exemples de l'énoncé à vos scripts : assurez-vous que le répertoire de vos scripts est dans le `PATH` et exécutez `$DONNEES/test.sh`. Ceci est un test minimal que doivent passer vos scripts, cela ne vous dispense en aucun cas de réaliser vos propres tests.

Les questions sont indépendantes. Si vous pensez pouvoir répondre à une question à l'aide d'un script d'une question précédente que vous n'êtes pas parvenu à écrire, n'hésitez pas à écrire la solution "comme si" vous aviez écrit le script.

Conventions typographiques et résultats contractuels

Nous utilisons dans ce sujet la convention typographique utilisée dans le cours pour les exemples : les lignes en gras commençant par `%` correspondent à des commandes, le `%` ne faisant pas partie de la commande (c'est le prompt du shell dans notre convention). Les lignes qui suivent (jusqu'à une nouvelle ligne en gras ou la fin du paragraphe) sont le résultat de l'exécution de la commande.

Rendu du TME

Placez vos scripts (et uniquement vos scripts, pas les fichiers d'exemples !) dans un seul répertoire, et lancez le script `rendu.sh` que vous trouverez dans le fichier d'exemples, avec comme argument le chemin de ce répertoire. Si vous le souhaitez, vous pouvez également inclure un fichier texte appelé `README`, où vous pourrez mettre tout commentaire que vous souhaiteriez que je lise (difficultés rencontrées, etc.). N'oubliez pas également que vous pouvez placer des commentaires dans vos scripts.

Par exemple, si vous placez vos scripts dans le dossier `~/TME_solo`, et si vous avez suivi les instructions préliminaires, il suffira de lancer la commande

```
$DONNEES/rendu.sh ~/TME_solo
```

Pensez à bien vérifier que tous vos scripts sont bien présents dans le répertoire avant de lancer la commande. En cas d'erreur, n'hésitez pas à la relancer, seule la dernière soumission sera corrigée.

En cas de problème avec le script `rendu.sh`, mettre les scripts dans une archive `<numéro-étudiant>.tar` et envoyer l'archive par mail à l'adresse `thibaut.verron@lip6.fr`.

Carnet d'adresses

On travaille avec un carnet d'adresses, c'est-à-dire une liste de lignes sous la forme suivante

```
Nom/Prénom/Mail/Telephone/Ville
```

Voici à titre d'exemple les premières lignes de ce fichier :

```
% head -n5 $DONNEES/carnet.txt
Gerard/Bruno/Bruno.Gerard@yahoo.fr/0978210017/Strasbourg/
Guerin/Audrey/Audrey.Guerin@laposte.net/0862230380/Rennes/
Lacroix/Delphine/Delphine.Lacroix@outlook.fr/0334186214/Grenoble/
David/Sandrine/Sandrine.David@laposte.net/0626231157/Nantes/
Lopez/Thomas/Thomas.Lopez@gmail.com/0702728556/Rouen/
```

Tous les scripts peuvent être écrits sans expressions régulières.

1 Recherche de données 1

Écrivez un script `chercher_mail` qui lit sur son entrée standard un carnet d'adresses et prend en argument un nom de fournisseur d'adresses mail f. Il doit afficher sur sa sortie standard les noms et numéros de téléphone des personnes ayant une adresse mail chez le fournisseur f. Ces informations seront séparées par des espaces.

```
% chercher_mail "gmail.com" < $DONNEES/carnet.txt | head -n3
Lopez 0702728556
Roy 0357225134
Francois 0103721167
```

2 Recherche de données 2

Écrivez un script `compter_ville` qui lit sur son entrée standard un carnet d'adresses et prend en argument un nom de ville. Il doit afficher sur sa sortie standard le nombre de personnes habitant la ville donnée.

Attention, certaines personnes peuvent porter le nom ou le prénom d'une ville.

```
% compter_ville Caen < $DONNEES/carnet.txt
1
```

3 Extraction d'une ligne

Écrivez un script `ligne` qui lit sur son entrée standard un carnet d'adresses, et prend en argument un nombre n. Il doit afficher sur sa sortie standard la n-ème ligne du carnet d'adresses.

Si n est vide, nul, négatif ou supérieur au nombre de lignes du fichier, afficher une ligne vide sur la sortie standard.

```
% ligne 5 < $DONNEES/carnet.txt
Lopez/Thomas/Thomas.Lopez@gmail.com/0702728556/Rouen/

% ligne < $DONNEES/carnet.txt

% ligne 60 < $DONNEES/carnet.txt
```

4 Recherche de données 3

Écrivez un script `chercher_ville` qui lit sur son entrée standard un carnet d'adresses et prend en argument un nom de ville. Il doit afficher sur sa sortie standard le nom et le numéro de téléphone (séparés par des espaces) de la première personne du carnet d'adresses qui habite dans cette ville.

```
% chercher_ville Caen < $DONNEES/carnet.txt
Aubert 0958524014
```

5 Recherche de données 4

Écrivez un script `verifier_mail` qui lit sur son entrée standard un carnet d'adresses, et affiche sur sa sortie standard le nom, le prénom et l'adresse mail des personnes dont l'adresse mail n'a pas la forme "Prénom.Nom@fournisseur". Ces données seront séparées par des espaces.

Pour simplifier le problème, si *Paul Dupont* a pour adresse mail *paul.dupont@mail.com* (sans majuscules), il devra apparaître dans les résultats du script.

```
% verifier_mail < $DONNEES/carnet.txt | wc -l
4

% verifier_mail < $DONNEES/carnet.txt | head -n 3
Rousseau Jeremy Rousseau.de.Amiens@laposte.net
Picard Guy Picard.de.Nice@wanadoo.fr
Hilton Paris americanactress@mail.com
```