Université de Nice-Sophia Antipolis — Vendredi 7 Octobre 2005
Master 2 TIM — Durée: 4h00

MN 2.1 Initiation Unix – TP3 : Shell avancé et Regexp

Olivier Dalle (`Olivier.Dalle@unice.fr`)

---

1 Echauffement

Terminez la feuille d'exercice de la première séance si ce n'est pas déjà fait ...

- Combien d'espace disque occupent les fichiers de votre compte ?
- Combien d'espace occupent les fichiers qui se trouvent dans les sous-répertoires de votre *Home Directory* dont le nom commence par le caractère '.' ("point").
- Combien d'espace disque reste-t-il sur les partitions *montées* par votre machine ?

2 Fichiers de Commandes simples

Première expérience : Créez un fichier de nom `infos_zsh` contenant le script de la dernière page du chapitre 4 du cours et testez-en le résultat dans les différents cas d'exécution proposés.

2.1 Lecture - Test - Boucle

Créer un FC (de nom *fc1*) :

1. Qui affiche textuellement le message "Repondez par OUI ou NON :", lit la réponse de l'utilisateur au clavier, puis affiche "C'est OUI" lorsque l'utilisateur a tapé OUI, "C'est NON" s'il a tapé NON, "C'est ni OUI, ni NON" dans les autres cas. Respecter scrupuleusement les messages à afficher.
   Tester le FC.
2. Modifier le FC de telle sorte que l'on puisse reconnaître minuscules et majuscules.
   Tester.
3. Modifier le FC de sorte que :
   il affiche le message "'Répondez par OUI ou NON :", lit la réponse de l'utilisateur au clavier, puis boucle jusqu'à ce que l'utilisateur tape effectivement *OUI* ou *NON* (en majuscules), puis termine en affichant "C'est OUI" lorsque l'utilisateur a tapé OUI, "C'est NON" s'il a tapé NON.
   Pensez aux formules de Morgan !

   Tester.

2.2 Paramètres - Variables - Décalage

Ecrire un FC (de nom *liste*) qui liste les fichiers : (1) de suffixe donné en premier paramètre et, (2) qui se trouvent dans une liste de répertoires donnés dans les paramètres suivants.

Syntaxe d'Appel (SA) : liste <suffixe> [<rep1> <rep2> ...]

Notes :
- Contrôler que le nombre de paramètres est au moins égal à 1.
- En l'absence de répertoire, travailler sur le répertoire en cours.
- Chercher les options de la commande de test avec l'aide en ligne : *man zshmisc puis CONDITIONAL EXPRESSIONS*.
- Utiliser impérativement : $# -lt -eq

2.3 Question subsidiaire

Ecrire un fichier de commandes Unix (*zsh*) ayant la syntaxe d'appel

traiter MOT F1 F2 F3 F4 ...

et dont le rôle est de mettre dans un fichier de nom *result* toutes les lignes des fichiers F1 F2 F3 ... qui contiennent le mot MOT.
Dans le cas où il n'y a aucun paramètre, on affiche le message "Paramètres manquant", s'il n'y en a qu'un seul, on affiche le message "Au moins un fichier doit être spécifié". Après le traitement de chaque fichier, on affiche le message "Fichier F*i* traité".
Si l'on exécute plusieurs fois de suite le fichier de commande *traiter*, le fichier *result* doit contenir uniquement les résultats de la dernière exécution.

3 Fichiers de Commandes avancés

3.1 Boucle FOR

Ecrire un FC (de nom *listechaine*) qui affiche le contenu de tous les fichiers qui :
- contiennent dans leur nom une chaîne de caractère spécifiée en premier paramètre et,
- se trouvent dans les répertoires donnés dans les paramètres suivants.

Syntaxe d'Appel : `listechaine <chaine> <rep1> [<rep2> ... ]`

Avant chaque répertoire, afficher son nom et faire une pause :

```
Répertoire suivant: non_repertoire
Tapez return pour continuer ...
```

Avant chaque fichier, afficher le message :

```
Fichier suivant: non_fichier
Tapez return pour continuer ...
```

Utiliser une boucle *FOR*.

3.2 Fichier de Commande de type Menu

1. Ecrire un FC (sans paramètre) qui permet d'obtenir les fonctionnalités suivantes :

   0) Sortir du programme
   1) Nom de login de l'utilisateur en cours
   2) Numero du processus courant
   3) Numero du processus père
   4) Afficher tous les processus d'un utilisateur (nom à saisir)
   5) Informations sur un utilisateur avec la commande *finger* (nom à saisir)
   6) Nombre d'utilisateurs connectés
   7) Lancer une nouvelle fenêtre (dans le cas d'un terminal graphique)
   8) Rechercher de fichiers (lisibles) *gif* ou *jpg* depuis un répertoire, affichage *xv*

   Réaliser un menu correspondant à ces fonctionnalités ; le FC ne se termine que lorsque l'utilisateur choisit "0". Utiliser la structure de contrôle CASE.
2. Question subsidiaire (2) : Ajouter une commande au menu (8) qui tue le dernier xterm lancé (7).

4 Manipulation sur le contenu des fichiers, en utilisant des expressions régulières

1. Après avoir tapé la commande ci-après sous Solaris, `ls -lL /bin > Liste`, quel est l'effet des commandes suivantes dont le but est de faire un filtre selon certaines caractéristiques du nom de fichier :
   – `awk '$9 ~ /[0-9]/' Liste`

– `grep '[0-9][^ ]*$' Liste` ; pourquoi n'est-il pas suffisant de taper `grep '[0-9]' Liste` ?

2. Indiquez ce que font les commandes suivantes ; servez-vous en pour donner une commande répondant aux spécifications données
   – `ls -A /usr/bin | grep '^[^0-9]*$'`
   – afficher les noms de fichiers dont le nom ne contient qu'un seul chiffre
   – `ls -A /usr/bin | grep '[0-9].*[0-9]'`
   – afficher les noms de fichiers dont le nom ne contient pas que des chiffres
   – `ls -A /usr/bin | grep '^[^0-9]*.[^0-9]*.[^0-9]*$'`
3. Affichez les noms des fichiers de `/usr/bin` dont le nom commence et finit par la même lettre (toutes les 2 soit en minuscule, soit en majuscule).
4. Recherchez les noms de tous les fichiers de votre arborescence dont le nom commence par la lettre "c", en utilisant find. Sachez que find accepte derrière l'option `name` un modèle fonctionnant comme ceux acceptés par le shell.
5. Parmi **tous** les processus qui s'exécutent sur votre machine, utilisez la combinaison de commandes `ps X | grep Y | awk Z`, après avoir remplacé les morceaux *X,Y,Z* de sorte à : n'afficher que les colonnes indiquant le PID, le PPID, et la COMMAND et ce uniquement pour les processus correspondant à des shells zsh.