L3 Informatique
Université de Bourgogne

U.E. Systèmes & Réseaux I

Exercices shell - awk - C

1 Shell

Exercice 1- *Quotation*

Expliquer le comportement de chacune des lignes de commandes suivantes:

```
A='echo *'
echo '$A'
echo "$A"
echo $A
$A
```

Exercice 2- *Bilan sur un fichier*

Écrire un programme shell qui lit un nom de fichier et affiche les renseignements le concernant sous la forme :

```
nom du fichier : blabla
numéro de i-noeud : blabla
nom du propriétaire : blabla
nom du groupe : blabla
taille : blabla
nombre de liens : blabla
```

Exercice 3- *Répertoire courant*

Écrire, sans utiliser de structure de contrôle, les commandes suivantes:

1. `nf` : affichage du nombre de fichiers du répertoire courant,
2. `ra` : affiche oui si le répertoire courant est le répertoire d'accueil, non sinon;
3. `prc` : affiche la profondeur du répertoire courant;
4. `ouest` : `ouest <nom>` affiche `<nom> est dans <rep>` où `<rep>` est le répertoire d'accueil de `<nom>`, si nom est un utilisateur défini et `<nom>` n'existe pas sinon.

Exercice 4- *En-tête programmes C*

Ecrire le script ifdef tel que l'exécution de `ifdef <nom>.h` place en tête de `<nom>.h` les lignes

```
#ifndef <NOM>_H
#define <NOM>_H
```

et la ligne

```
#endif /* <NOM>_H */
```

à la fin.

Modifier ensuite le script ifdef de façon à ce qu'il

- ne modifie pas le fichier s'il commence déjà par une ligne `#ifndef <NOM>_H`;
- accepte l'option -C: ajout du commentaire `/* Fichier <nom>.h */`;
- rajoute automatiquement le suffixe .h s'il n'est pas présent dans `<nom>`.

Exercice 5- *Arborescence* Dans tout cet exercice, on suppose que les lettres sont non accentuées.

1. Écrire un programme shell qui reçoit en paramètre deux noms de fichiers. Le premier fichier (texte) contient, ligne par ligne, des noms et prénoms d'étudiants. Le second fichier est un document pdf. Le script doit créer un répertoire par étudiant (dont le nom est constitué de ses noms et prénoms, en majuscules, sans espace ni tiret) et placer dans ce répertoire une copie du document passé en second paramètre. Tous les répertoires étudiants seront ensuite regroupés dans une archive tar.

2. Soit l'arborescence constituée d'un répertoire principal Examen contenant contenant 7 sous-répertoires GroupeA,... , GroupeG. Chaque répertoire Groupe[A-G] contient deux répertoires Sujet1 et Sujet2 et chaque répertoire Sujet[1-2] contient plusieurs répertoires contenant chacun le fichier Sujet[1-2].odt.

   Sur cette arborescence, écrire un programme shell qui vérifie que chaque répertoire Sujet1 ne contient que des documents Libre Office Sujet1.odt et que chaque répertoire Sujet2 ne contient que des documents Libre Office Sujet2.odt. Votre programme shell reçoit en paramètre le chemin d'accès au répertoire Examen.

3. Toujours sur cette arborescence, donner la suite de commandes pour renommer chaque fichier Libre Office sous la forme groupeX_nom_Sujet1.odt ou groupeX_nom_Sujet2.odt.

4. Créer ensuite deux répertoires SUJET1 et SUJET2 dans le répertoire Examen et y copier tous les fichiers correspondants.

5. Les documents odt sont en fait des archives .zip. Les décompacter et vérifier dans le répertoire obtenu par décompression (Sujet1 ou Sujet2) que le fichier meta.xml contient bien une balise `meta:image-count="1"` et que le répertoire `Pictures` contient bien une et une seule image dont vous afficherez le nom.

2 AWK

Exercice 6- Que produisent les scripts awk suivants ?

1. `{print $2 , $1}`
2. `$1 != p {print ; p = $1 }`

Exercice 7- Écrire les scripts awk permettant d'afficher

1. les deux premiers champs en ordre inversé avec des champs séparés par des virgules et/ou des espaces et tabulations;
2. la somme et la moyenne du premier champ;
3. toutes les lignes entre "start" et "stop";
4. toutes les lignes plus longues que 72 caractères;
5. tous les champs de chaque ligne en ordre inverse.

Exercice 8- Écrire les scripts awk permettant

1. de tester si tous les enregistrements d'un fichier ont le même nombre d'éléments;
2. d'inserer des numéros de page dans un fichier, sachant que le fichier d'entrée contient déjà une ligne commençant par "Page" à chaque endroit ou le numéro de page doit être inséré.

Exercice 9- Je garde la trace de mes dépenses et de mes rentrées d'argent dans le fichier texte `comptes.txt`, chaque ligne ayant la forme suivante : `date dépense rentrée motif` (l'un des deux champs dépense ou rentrée vaut toujours 0). Par exemple,

```
$ cat compte.txt
21/10/16 53 0 chaussures
05/12/16 0 35000 salaire
10/12/16 75 0 courses
```

Ecrire les scripts awk suivants :

1. vérification que l'un des deux champs dépense ou rentrée vaut 0 pour chaque ligne;
2. affichage du total des dépenses, des rentrées et des périodes durant lesquelles le compte à été à découvert;
3. affichage du nombre de jours min, max et moyen entre deux dépenses.

3 C et makefile

Exercice 10- Soit un programme en C, `main.c`, utilisant les bibliothèques utilisateurs `gui.h` et `mesmath.h`.

1. La bibliothèque `mesmath.h` contient la fonction `int monexp(int a,int b)`. Donner le code de `mesmath.h`
2. La bibliothèque `gui.h` utilise également `mesmath.h`. Dessiner le graphe de dépendance associé à ces programmes.
3. Donner le fichier makefile associé à leur compilation.
4. En supposant qu'aucun fichier .o n'existe, donner les commandes déclenchées par `make main`.
5. Existe-t-il des cibles parallélisables ?
6. Donner les commandes déclenchées par `make main` une nouvelle fois.
7. En supposant que tous les fichiers soient à jour, mais que `mesmath.c` soit modifié, donner les commandes déclenchées par `make main`.

Exercice 11- Soit un système utilisant 9 formats de fichiers, suffixés par `a`, `b`, `c`, `d`, `e`, `f`, `g`, `h`, `i` et incluant les commandes de conversion de format suivantes : `a2b, c2d, e2f, bd2g, df2h, gh2i`. On suppose que ces commandes s'utilisent ainsi : pour les trois premières, `x2y fic.x fic.y` va transformer le fichier `fic.x` au format `x` en un fichier `fic.y` au format `y`; pour les trois dernières, `xy2z fic.x fic.y fic.z` va produire le fichier `fic.z` à partir des fichiers au format `fic.x` et `fic.y`.

1. Pour les fichiers `essai.a, essai.c, essai.e`,
   (a) Dessiner le graphe de dépendences de la cible `essai.i`.
   (b) Donner le suite de commandes pour obtenir la cible `essai.i`.
2. Proposer un fichier Makefile pour automatiser l'obtention de n'importe quelle cible au format .i à partir des fichiers au format .a .c .e.
3. Si on a exécuté `make essai.i` et que l'on modifie `essai.h`, que se passe-t-il si on relance `make essai.i` une nouvelle fois ?