Documents autorisés
Calculatrice interdite

Outils système – L2 info.

Contrôle terminal (A. Scheuer)

Jeudi 8 avril, 14 h 30 – 16 h 30

Exercice 1 — Expressions régulières (7 points)

Donnez le sens des expressions régulières suivantes :

a) `^([^[:space:][:cntrl:]]+[[:space:]])+[[:alnum:]]*` ; *(1,5 point)*
(vous aurez noté qu'un caractère alphanumérique est représenté par `[:alnum:]`, et non `[:alphnum:]` comme écrit dans le cours ;)

b) `([bcdfghjklmnpqrstvwxz][aeiouy]{1,2}){2,3}[-_.]?[0-9]+$`. *(2 points)*

Trouvez une expression régulière pour reconnaître :

c) une immatriculation récente valide, c'est-à-dire formée uniquement de deux lettres majuscules suivies par un tiret, trois chiffres, un tiret et deux lettres majuscules ; *(1 point)*

d) l'adresse d'une page `html`, sécurisée ou non, autrement dit constitué uniquement d'un préfixe `http` avec un `s` optionel, suivi d'un `://`, d'un ensemble d'au moins un caractère contenant des caractères alphanumériques, des points ou des tirets, suivi d'un point et d'au moins un caractère alphabétique, puis d'une suite d'au moins un ensemble d'au moins un caractère contenant des caractères alphanumériques, des points, des soulignés (`_`) ou des tirets, ces ensembles étant suivis d'un `/`, sauf le dernier suivi d'un point, le tout terminé par `html`, le `l` final étant optionel ; *(2,5 points)*
on rappelle qu'une partie de cette expression régulière a été vue en travaux dirigés.

Exercice 2 — Liste des utilisateurs réels (7 points)

On souhaite obtenir la liste des utilisateurs reconnus par un ordinateur sous Linux.

La liste des identifiants reconnus est disponibles dans le fichier `/etc/passwd`. Cependant, bon nombre de ces identifiants ne correspondent pas à des utilisateurs réels. Ces derniers se distinguent par le fait qu'il leur est associé (en fin de ligne) un vrai interpéteur de commande (se terminant par `sh`, et non une commande comme `/usr/sbin/nologin`, `/bin/false` ou `/bin/sync`).

a) En utilisant `grep`, comment peut-on obtenir la liste des utilisateurs réels (c'est-à-dire les lignes de `/etc/passwd` se terminant par `sh`) ? *(1 point)*

b) Afin de rendre le résultat plus lisible, on souhaite n'afficher que l'identifiant de ces utilisateurs, et non toutes les données contenues dans `/etc/passwd`.
Sachant que ces données sont séparées par des caractères `:`, et que l'identifiant est le premier champ, comment peut-on utiliser `awk` pour restreindre ainsi l'affichage ? *(2 points)*

c) Enfin, pour un résultat optimal, on souhaite que ces identifiants soient sur la même ligne, séparés par des virgules.

.../...

Pour cela, il faut (toujours avec `awk`) remplacer le séparateur de sortie des enregistrements par `""`, le remettre à `"\n"` à la fin, afficher `", "` avant le premier champ si l'identifiant n'est pas le premier (il faut donc une variable qui compte les identifiants affichés), et terminer l'affichage par un passage à la ligne.

Traduisez cet algorithme en langage `awk`. *(4 points)*

Exercice 3 — Makefile (6 pts)

On considère un système constitué :

— de 9 formats de fichiers, dont les suffixes sont `a`, `b`, `c`, `d`, `e`, `f`, `g`, `h` et `i`,

— et de 6 commandes de conversion `a2b`, `c2d`, `e2f`, `bd2g`, `df2h` et `gh2i`.

Les trois premières commandes prennent deux fichiers comme arguments, et transforment le premier fichier en celui donné en second : `a2b f.a f.b` transforme le fichier `f.a` au format `a` en un fichier `f.b` au format `b`, les commandes `c2d` et `e2f` faisant de même repectivement pour les formats `c` et `d`, et `e` et `f`.

Les trois commandes suivantes prennent trois fichiers comme arguments, et génèrent le dernier à partir des deux premiers : `bd2g f.b f.d f.g` génére le fichier `f.g` à partir des fichiers `f.b` et `f.d`, et ainsi de suite...

a) À partir de trois fichiers `essai.a`, `essai.c` et `essai.e`, dessiner le graphe de dépendences permettant d'obtenir la cible `essai.i`. *(1 point)*

En déduire la suite de commandes permettant d'obtenir cette cible. *(1 point)*

b) Proposer les règles génériques pour un `Makefile` afin d'automatiser la génération de n'importe quelle cible au format `i` à partir des fichiers associés aux formats `a`, `c` et `e`. *(2,5 points)*

c) Si on a exécuté `make essai.i` une première fois, et que l'on modifie `essai.h`, que se passe-t-il si on relance `make essai.i` une nouvelle fois ? Justifiez votre réponse. *(1,5 points)*