Université de Lille — FACULTÉ DES SCIENCES ET TECHNOLOGIES — Département Informatique

TDM 2 - Automates et Langages Réguliers

septembre 2021

Manipulation d'expressions régulières (2ème partie : avec **egrep**)

1 Introduction

egrep est un outil UNIX permettant une recherche de **motif** (expression régulière) dans un ou plusieurs fichiers. Dans son fonctionnement « normal » :

```
egrep motif fichiers
```

En pratique, on protège le motif par des quotes afin d'empêcher son interprétation par le shell Unix. Ces quotes ne font bien évidemment pas partie du motif

```
egrep 'motif' fichiers
```

1.1 Recherche de sous-chaînes, ligne par ligne

egrep recherche, **séparément dans chaque ligne du fichier**, un texte correspondant au motif. Il n'est pas nécessaire que la ligne entière corresponde au motif : egrep cherche une (ou plusieurs) sous-chaîne(s) qui conviennent. Voici quelques exemples :

motif	ligne (occ. trouvées en rouge)	nb
`à\s[a-z]+`	`Que j'aime à faire apprendre ce nombre utile aux sages`	1
`re`	`Que j'aime à faire apprendre ce nombre utile aux sages`	4
`.+`	`Que j'aime à faire apprendre ce nombre utile aux sages`	1
`[0-9]+`	`Que j'aime à faire apprendre ce nombre utile aux sages`	0
`\s.+`	`Que j'aime à faire apprendre ce nombre utile aux sages`	1

Par défaut, egrep recherche toujours la plus longue chaîne possible correspondant au motif (pour le premier exemple, la recherche ne s'arrête pas à « `à f` », qui convient pourtant au motif, mais se poursuit pour trouver « `à faire` »)

1.2 Fonctionnement et options

egrep affiche chaque ligne du ou des fichiers dans laquelle le motif a pu être trouvé.
Par exemple : `egrep [A-Z][0-9] *.txt` affichera toutes les lignes des fichiers `.txt` contenant une majuscule suivie d'un chiffre décimal (quoi qu'il se trouve avant ou après). Voici quelques options qui modifient le fonctionnement de egrep :

Modifient la forme du résultat	
–color	affiche en couleur les sous-chaînes trouvées
-n	préfixe chaque ligne par son numéro
-c	affiche uniquement le nombre de lignes trouvées, pas les lignes elles-même
-h	n'affiche pas le nom du fichier en début de ligne
-o	n'affiche que la partie de la ligne qui correspond au motif
Modifient les critères de recherche	
-i	ignore la casse majuscules/minuscules
-x	ne cherche que les lignes qui correspondent en totalité (et pas les facteurs propres)
Modifient la syntaxe ou la source des motifs	
-P	utilise la syntaxe Perl pour les motifs
-f nom de fichier	lit le motif dans le fichier et non sur la ligne de commande

2 Syntaxe des expressions

En plus de ce que nous avons vu lors de la séance précédente, voici quelques fonctionnalités disponibles dans les expressions (avec egrep).

2.1 Classes de caractères prédéfinies

Nom symbolique	Classe correspondante
`[:alnum:]`	les caractères alpha-numériques
`[:alpha:]`	les caractères alphabétiques
`[:cntrl:]`	les caractères de contrôle
`[:digit:]`	les caractères chiffres
`[:lower:]`	les lettres minuscules
`[:punct:]`	les caractères de ponctuation
`[:space:]`	les caractères espace (équivaut à `\s`
`[:upper:]`	les lettres majuscules

Par exemple, pour désigner une lettre on pourra écrire `[[:alpha:]]`
Pour désigner une lettre ou un `@`, ou pourra écrire `[[:alpha:]@]`
Attention : quand le motif est entré sur la ligne de commande (donc dans la plupart des cas) les caractères spéciaux Unix doivent être échappés. Il est conseillé de mettre l'expression régulière entre deux signes '

2.2 Assertions

Une assertion est un élément de la syntaxe des expressions régulière qui fixe une condition portant sur le contexte dans lequel le motif est recherché. Les deux assertions les plus connues sont ^ et $

^	début de ligne
$	fin de ligne

Exemples

motif	ligne (occ. trouvées en rouge)	nb
`[0-9]+`	`ab cd 452`	1
`^[0-9]+`	`ab cd 452`	0
`[0-9]+$`	`ab cd 452`	1
`[0-9]+$`	`ab cd 452x`	0
`^[0-9]+$`	`ab cd 452`	0
`^[0-9]+$`	`452`	1

Un mot « usuel » est une suite de caractères qui peuvent être une lettre de l'alphabet usuel, un chiffre ou l'underscore. Des assertions permettent de vérifier si le motif apparaît au début ou en fin d'un mot usuel.

`\<`	début d'un mot usuel
`\>`	fin d'un mot usuel
`\b`	début ou fin d'un mot usuel

Exemples

motif	ligne (occ. trouvées en rouge)	nb
`\<[abc]+`	`axaa xbb ccxcy`	2
`\b[abc]+`	`axaa xbb ccxcy`	2
`[abc]+\>`	`axaa xbb ccxcy`	2
`[abc]+\b`	`axaa xbb ccxcy`	2
`\b[0-9]+\b`	`a99b 1234 x56 a78`	1
`\<[0-9]+\>`	`a99b 1234 x56 a78`	1

3 Exercices

Exercice 1 : *Cyrano*
Dans cet exercice vous rechercherez des textes dans le fichier Cyrano.txt.
Pour chacune des questions ci-dessous, vous afficherez les lignes correspondant aux caractéristiques demandées, en utilisant l'option `--color` pour visualiser le motif trouvé.
NB : vous utiliserez, en le complétant, le script shell fourni.

1. contenant la sous-chaîne « nez »
2. contenant 3 points successifs
3. contenant 3 voyelles non accentuées successives
4. commençant par un A
5. se terminant par un ?
6. contenant un l non suivi d'un e

7. contenant exactement 2 occurrences de "
8. contenant le mot usuel « nez »
9. contenant un mot usuel se terminant par 3 voyelles non accentuées successives
10. contenant un mot usuel composé de 7 lettres alphabétiques exactement
11. contenant un mot usuel composé de 9 à 12 lettres alphabétiques
12. contenant un mot usuel commençant par l ou L et se terminant par e

Exercice 2 : *XML/HTML*

Q 1 . En reprenant et adaptant ce que vous avez fait pour le premier TP, définissez dans le script shell une variable `nomXML` qui contient l'expression régulière correspondant à un nom XML. Puis définissez une variable `refEntite` qui contient l'expression régulière correspondant à « référence entité » (en utilisant la variable précédente)
Affichez les lignes du fichier `l2.html` qui contiennent une référence entité (en colorisant les références entités trouvées)

Q 2 . ajoutez au script la définition d'une variable `valeurAttribut` qui contient l'expression régulière correspondant une « valeur d'attribut XML ». (vous utiliserez la variable `refEntite` précédemment définie).
Affichez les lignes du fichier `l2.html` qui contiennent une une chaîne respectant cette syntaxe (en colorisant les chaînes trouvées)

Q 3 . Définissez également une variable `baliseOuvrante` en utilisant les variables précédentes.
Affichez les lignes du fichier `l2.html` qui contiennent une balise ouvrante (en colorisant les les balises trouvées)