Pratique des machines
TP8 : Expressions régulières (suite)

am@up8.edu

Novembre 2022

Dans ce TP :
— Plus d'exercices sur les expressions régulières
— Construction d'un jeu de piste : construisez vous-même les épreuves

(rappel) syntaxe des expressions régulières

Une expression régulière est une séquence de caractère qui décrit un ensemble de séquences de caractères.
La plus simple exemple est la séquence "abc" qui correspond à la séquence "abc".
Les expressions régulières utilisent certains caractères pour désigner des ensembles de caractères :

— **.** (le point), n'importe quel caractère : **.bc** désigne abc, bbc, cbc, etc.
— [ ] (les crochets) désignent un ensemble (ou un intervalle) de caractères possibles (*ou impossibles* avec un chapeau en début) :
  — [**ab**]**bc** désigne abc et bbc ;
  — [**^ab**]**bc** désigne cbc, dbc, ebc, etc. mais pas abc et bbc
  — [**a-c**]**bc** désigne abc, bbc, cbc
  ;
— | (le pipe), une alternative : a|b désigne a ou b
— **?** signifie : 0 ou une fois : abc ? désigne abc et ab
— ***** signifie : 0, une ou plusieurs fois : **a*bc** désigne bc, abc, aabc, aaabc,aaabc, etc.
— **+** signifie : une ou plusieurs fois : **a+bc** désigne abc, aabc, aaabc, etc.
— **()** (les parenthèses) groupe une séquence, cela s'applique en combinaison avec les opérateurs précédents : **(ab)+c** désigne abc, ababc, abababc etc.
— ^ (le chapeau), désigne le début d'une ligne
— **$** (le dollar), désigne la fin d'une ligne

1 Exercice : Ecriture d'expressions régulières et test avec la commande grep

Dans cet exercice, je vous donne une suite de **motifs**. Pour chaque motif, vous devez :
— écrire l'expression régulière correspondante ;

— la tester sur une chaîne de caractère.

Par exemple, si on souhaite reconnaître le motif "ri" suivi d'un caractère, je peux écrire l'expression régulière : "**ri.**". Pour tester mon expression, je peux par exemple exécuter la commande suivante :

```
echo "ri, tri, rit, ribambelle, Paris, il a ri" | grep -E --color "ri."
```

Les occurrences du motif apparaissent en couleur, que remarque-t-on au passage ?

Pour faire un test sur plusieurs lignes, on peut utiliser "\n" dans une chaîne de caractères, par exemple la commande suivante permet de reconnaître la lettre « u » en début de ligne seulement :

```
echo "un\ndeux\ntrois" | grep -E --color "^u"
```

Vous pouvez également créer un fichier texte chaines_de_test.txt dans laquelle vous écrivez les chaînes à tester, et appliquer votre commande **grep** :

```
cat chaines_de_test.txt | grep -E --color "^u"
```

Faites de même en notant dans un fichier `tests_regexp.sh` vos commandes de test les unes à la suite des autres pour les motifs suivants.

Attention c'est bien TOUT le motif qui doit apparaître en couleur.

La commande `egrep` est un raccourci pour `grep -E`. Voici quelques options utiles :

— `-n` Affiche la ligne où l'occurrence a été trouvée
— `-c` Compte le nombre d'occurrences trouvées
— `-r` Recherche dans tous les fichier d'un dossier
— `-B n` Affiche n lignes avant l'occurrence trouvée
— `-A n` Affiche n lignes après l'occurrence trouvée
— `-C n` Affiche n lignes avant et après : le contexte
— `-v` : grep inversé, affiche les lignes qui ne contiennent *pas* le motif

1.1 Plus d'expressions régulières

1. Les noms propres sans espaces (pensez aux caractères pouvant apparaître dans un nom propre)
   `[A-Z][^ ]+`
2. les balises html (, , , etc.)
   `<[^<]+>`
3. les acronymes, formés de lettres en majuscule suivies d'un point (S.N.C.F.)
   `([A-Z].)+`

1.2 Options de grep

Que font les commandes suivantes (recopiez-les, ne faites **pas** de copier-coller) :

1. `ps aux | egrep "^$ LOGNAME" -n`
2. `grep "e" -r . -c`

3. `man bash | grep -A 2 "select"`
4. `ls -ap   | grep -v "/"`

1.3 Tricher aux mots croisés

Pour pouvoir tricher aux mots-croisés, téléchargez le fichier liste_français.txt et trouvez :

Rappel : vous pouvez tester vos expressions régulières sur `https://regex101.com/`

1. la liste des mots de trois lettres qui finissent par Z
   ^..z$
2. la liste des mots de six lettres de la forme _ _ D _ _ E
   ^..d..e$
3. la liste des mots de quatre lettres de la forme A _ D _ ou E _ D _
   ^(a|e).d.$
4. la liste des mots qui termine par OU ou EAU
   ^.*(o|ea)u$
5. la liste des mots qui contiennent un Q suivi d'une consonne
   ^.*q[zrtpqsdfghjklmwxcvbn].*$

1.4 La rétro-référence

On peut vouloir trouver des suites de motifs qui se répètent. Testez le motif suivant sur la liste des mots et déduisez-en le sens de \1 :

1. ^([a-z]+)\1$

Téléchargez le code source de la page `https://www.w3.org/Style/Examples/011/firstcss.fr.html` (en utilisant `wget` par exemple et utilisez `grep` pour récupérer les lignes contenant une balise ouvrante et fermante, par exemple : `<title> le titre </title>` ou `<cite> citation </cite>`. Attention la balise fermante doit être la même que la balise ouvrante.

<([^<]+)>.*</\1>

2 (à rendre pour la semaine prochaine) des énigmes à résoudre en Bash

Définissez au moins 3 énigmes en bash en mobilisant différentes commandes vues ce semestre. L'idée est de pouvoir construire un jeu de piste ou chaque résolution d'épreuve amène à une nouvelle énigme.

Les énigmes peuvent être de la forme :

— « L'énoncé de l'étape suivante se trouve dans le plus gros fichier de ce répertoire »

— « Le nom du fichier contenant la prochaine épreuve se trouve à la fin de la seule ligne du fichier x.txt qui contient deux fois le mot "chat". »

Pour chaque énigme, cous devez fournir en plus de son énoncé le ou les fichiers permettant de la résoudre.

Soyez imaginatif(ve)s !