Langages de script
TP n° 6 : Expressions régulières (2)

HTML

Nous traitons ici des fichiers `html` en utilisant des expressions régulières (attention : dans certains cas compliqués il vaut mieux utiliser des librairies dédiées au `html` ou `xml`). Utilisez les fichiers `accueil.html` (sans commentaires `html`) et `accueilfull.html` (avec) sur Moodle pour tester.

Exercice 1 :

1. Écrire une fonction qui prend comme entrée un nom de fichier `html` sans commentaires et qui imprime sur écran tous les contenus des balises `html` (sans `<`, `>` et `/`). On suppose que les balises ne contiennent pas de retour à la ligne et qu'il peut y avoir plusieurs balises sur une ligne.
2. Écrire une fonction qui prend comme entrée un nom de fichier `html` sans commentaires et qui imprime toutes les adresses des pages Web (les chaînes après `href="http://`) qui apparaissent comme liens dans le fichier et leur nombre. Pour le fichier `accueil.html` cela devrait donner 85 adresses.
3. Modifier la fonction précédente de sorte qu'elle affiche uniquement le nom de domaine de premier niveau (fr, com, etc.)
4. Écrire une fonction qui prend comme entrée un nom de fichier `html` et qui imprime le contenu de fichier sans les commentaires `html`. Un commentaire peut être sur plusieurs lignes. Le résultat du fichier `accueilfull.html` devrait être le fichier `accueil.html`.

Flaubert

Nous analysons et traitons ici le texte de Flaubert `educ.txt` (sur Moodle).

Exercice 2 :

1. Générer les listes suivantes : toutes les occurrences des mots qui contiennent la lettre `k` ou `w` (minuscules, il y en a 65) ; toutes les années du 19-ième siècle (écrites en chiffres, il y en a 21) ; toutes les formes du verbe `aimer` (il y en a 109). Modifier les expressions pour ne chercher que les mots en fin de ligne (il y en a 3,1 et 15, il peut y avoir des caractères spéciaux (ponctuations, etc.) entre les mots et la fin de la ligne).
2. Afficher les numéros de lignes qui contiennent les années de 19-ième siècle (il y en a 21).
3. Écrire une fonction qui trouve tous les mots qui contiennent 2 fois les mêmes trois lettres consécutives (comme `barbarie` ou `entrent`, il y en a 543).
4. Écrire une fonction qui trouve les parties (entre deux ponctuations) de phrases qui contiennent 2 fois le même mot de taille au moins 5 (il y en au moins 59, attention aux apostrophes).

5. Afficher les numéros de lignes et les positions dans ces lignes de toutes les formes du verbe `aimer` (il y en a 109).
6. Le texte est formé de chapitres annoncés par une ligne contenant `Chapitre N` où `N` est le numéro du chapitre. Écrire un script qui charge le fichier en mémoire et crée un fichier pour chaque chapitre, intitulé `chapitre-i.txt` (où `i` est le numéro du chapitre).
7. Modifier le script pour qu'il remplace partout le prénom "Frédéric" par "Gustave".
8. Modifier le script pour qu'il remplace tous les mots commençant par "aim" suivi d'un suffixe par le mot "détest" suivi du même suffixe.
9. Modifier le script pour qu'il extraie les dialogues de chaque chapitre dans un fichier nommé `chapitre-dialogues-i.txt`.