TP n°6 : manipulations de fichiers

PTSI Lycée Eiffel

12 décembre 2013

1 Lecture et écriture de fichiers texte avec Python

Ceux qui ont déjà eu droit au cours sur le sujet peuvent sauter les explications, mais sont tout de même priés de faire les petits exercices. Pour pouvoir faire tourner sous Python des programmes permettant d'analyser de gros fichiers de données, il est indispensable de pouvoir interagir directement avec un fichier (autre qu'un fichier .py de programme Python, on ne pourra pas se permettre de recopier dans Python toutes les données s'il s'agit par exemple d'un texte de plusieurs dizaines de pages). On se contentera dans ce TP de gérer des fichiers textes (extension .txt ou pas d'extension du tout ; sous Windows, cela correspond en gros aux fichiers directement lisibles à l'aide du Bloc-Notes). Commençons pour cela par donner les quelques commandes nécessaires à la manipulation de ces fichiers.

La première chose à faire est de pouvoir ouvrir un fichier avec Python, ce qui se fait à l'aide de la méthode open. En fait, techniquement, on créée un objet Python (dont le nom est complètement indépendant de celui du fichier texte) qui est un pointeur vers le fichier texte (un peu comme les listes sont constituées de pointeurs vers les cases mémoires contenant les variables de la liste), et sur lequel on peut appliquer différente méthodes, avec la syntaxe que vous devez commencer à maîtriser pour les méthodes en Python. La méthode pour ouvrir un fichier s'appelle simplement **open**, mais elle est assortie d'options qui spécifient ce qu'on veut faire avec le fichier une fois ouvert. On peut ainsi ouvrir un fichier en lecture (option 'r' sous Python) si on souhaite simplement lire le fichier, c'est-à-dire accéder à son contenu sans le modifier (puisqu'on travaille avec des fichiers texte, le contenu sera une chaîne de caractères) ; en écriture (option 'w') si on souhaite écrire des choses dans le fichier ; et en ajout (option 'a') si on veut ajouter du contenu à un fichier en contenant déjà. Attention, l'option 'w' créée automatiquement un nouveau fichier texte dans lequel écrire, mais surtout va écraser un fichier texte déjà existant portant le même nom (si on veut modifier un fichier existant, il faut absolument utiliser l'option 'a').

Pour résumer, et donner les autres méthodes utiles :

- **objet=open('fichiertexte','r')** permet d'ouvrir le fichier texte fichiertexte en mode lecture, sous le nom objet (bien évidemment, en pratique, fichiertexte et objet auront des noms moins génériques).
- **objet.read()** permet ensuite de lire le fichier texte, et d'en transférer tout le contenu dans une variable de type chaîne de caractères (par exemple via la commande **c=objet.read()**). Si on souhaite ne pas lire l'intégralité du fichier, on peut préciser un argument entier dans le *read*, et on ne lira qu'un nombre de caractères correspondant à l'argument. Si on fait un nouveau *read* ensuite, la lecture reprendra à l'endroit où on s'est arrêté au *read* précédent.
- **objet.readline()** permet de lire une ligne du fichier. La méthode **readlines()** (avec un s !) créée une liste contenant les différentes lignes du fichier texte.
- **objet=open('fichiertexte','a')** permet d'ouvrir le fichier fichiertexte en mode ajout.
- **objet.write('blablabla')** permet ensuite d'écrire le texte blablabla dans notre fichier. Si on effectue plusieurs *write* de suite, les différents textes seront écrits les uns à la suite des autres (à la fin du fichier). Le premier argument de cette méthode doit toujours être une chaîne de caractères.

- **objet=open('fichiertexte','w')** permet de créer un fichier nommé fichiertexte accessible à l'écriture.
- La méthode **write** est alors accessible de façon identique à ce qui a été décrit pour les fichiers en mode ajout.
- **objet.close()** permet de fermer le fichier, quel que soit le mode dans lequel on l'a ouvert. Prenez l'habitude de toujours fermer vos fichiers après les avoir manipulés, même si vous ne remarquerez sûrement aucune différence si vous oubliez de le faire, ça évite de surcharger la mémoire de Python avec des pointeurs inutiles.

Exercices

Une connaissance correcte du cours sur les chaînes de caractères peut être utile à la réalisation des programmes demandés.

1. Créer un fichier texte (via *open* en mode écriture) et y écrire un texte sur plusieurs lignes (rappelons que le retour à la ligne est codé par \n en Python). Bien. Maintenant, trouvez où Python a été fourrer votre fichier, soit à l'aide de la recherche Windows, soit à l'aide de la commande **getcwd** du module **os** de Python. Si vous ne voulez pas que vos fichiers atterrissent ici, utilisez la commande **chdir** de ce même module **os**.
2. Lire le fichier que vous venez de créer avec la méthode **readlines** et afficher le résultat.
3. Écrire un petit programme comptant le nombre de caractères de votre fichier texte. Comptez ces caractères à la main. Qu'en pensez-vous ?
4. Écrire un programme comptant le nombre de 'e' dans votre texte. Si vous êtes plus courageux, écrire un programme créant une liste contenat le nombre d'occurences de chaque lettre de l'alphabet.
5. Écrire un programme créant une copie de votre fichier texte où tous les caractères ont été mis en majuscules.
6. Écrire un programme créant un nouveau fichier texte ne contenant que les lignes du fichier précédent qui commencent par un 'e'. Même chose avec les lignes commençant par une majuscule (si votre fichier initial rend le résultat inintéressant, changez-le).
7. Écrire un programme affichant la phrase la plus longue de votre fichier.
8. Écrire un programme recopiant le texte dans un nouveau fichier, mais formaté de façon à contenir exactement 50 caractères par ligne.

Ceux qui préfèrent travailler avec un « vrai » fichier texte plus gros prendront cet extrait de *L'Assomoir* de Zola :

http ://www.normalesup.org/∼glafon/eiffel13/zola.txt

2 Traitement de données à l'aide de Python

Pour tous les exercices de cette partie, utilisez le fichier suivant, qui contient la liste des candidats à un concours récent, avec les quatre champs suivants : numéro de candidat, Nom (en majuscules) et prénom (en minuscules), résultat à l'écrit du concours, puis série d'oral pour les admissibles. Ces quatre champs sont séparés par des tabulations, le nom et le prénom par un simple espace. Attention quand même, le codage des caractères accentués est buggué !

http ://www.normalesup.org/∼glafon/eiffel13/admissibles.txt

Exercice

1. Compter le nombre total de candidats (avec Python, bien sûr), puis le nombre de candidats admissibles.

2. Créer quatre nouveaux fichiers texte contenant la liste des candidats de chaque série d'oral (uniquement nom et prénom).
3. Compter le nombre de candidats dont le prénom commence par chacune des lettres de l'alphabet.
4. Créer un copie du fichier de candidats où ces derniers sont classés par ordre de numéro et non par ordre alphabétique (ou par ordre alphabétique des prénoms si vous préférez).