TP n°2 *Utiliser des objets : la classe* `str`

Dans ce TP, nous allons utiliser une **classe** disponible nativement dans le langage Python : la classe **`str`** (les chaînes de caractères, ou *string* en anglais).

Une classe correspond à *un type de données évolué* (on parle aussi de type complexe) par opposition aux types *primitifs* tels que les entiers, réels, booléens, etc., dont les valeurs sont généralement atomiques.

Pour utiliser une classe (c'est-à-dire un type de donnée complexe), il faut *instancier* (on dit aussi *construire*, ce qui revient à déclarer et affecter) un élément du type en question. On peut par exemple construire une chaîne de caractères en utilisant les guillemets simples :

```
a = 'toto'
```

L'instance de la classe (ici dénotée par la variable `a`) est appelée **objet** (d'où le nom du paradigme de programmation : *Programmation Orientée Objets*).

Les classes définissent non seulement des types de données spécifiques, mais aussi des **opérations** autorisées sur ces types. Par exemple, il est possible de passer le premier caractère d'une chaîne en majuscule au moyen de la fonction (on parle aussi de **méthode**) `capitalize`, qui s'utilise comme suit :

```
print('toto'.capitalize())
```

Une description (on parle aussi d'**API** ou interface de programmation d'application, *Application Programming Interface* en anglais) des méthodes disponibles pour (et dans !) la classe **`str`** vous est fournie en annexe. À partir de celle-ci, vous devez répondre aux questions ci-dessous.

Exercice 1. Traitements sur un fichier CSV

Sachant qu'il est possible de charger le contenu d'un fichier CSV au moyen du code suivant :

```
import csv
with open(chemin_fichier, 'r') as f:
    contenu = csv.reader(f, delimiter=';')
    for tableau_ligne in contenu :
        pass # exemple d'instruction qui ne fait rien
```

1. Écrivez une fonction `nbchar` prenant en entrée un chemin vers un fichier CSV et retournant un entier représentant le nombre de caractères du fichier CSV.
   Vous testerez votre fonction sur les fichiers CSV fournis au TP1 (TP bibliothèque).
2. Écrivez une fonction `suffixe` qui, à partir d'un chemin vers un fichier CSV, et une chaîne de caractère, retourne la liste des champs (cellules) du fichiers CSV contenant un mot avec le suffixe passé en paramètre.
   Vous testerez votre fonction sur les fichiers CSV fournis au TP1, avec le suffixe `'ion'`.

Exercice 2. Traitements sur du texte

Dans cet exercice, nous allons travailler avec l'oeuvre "Les Trois Mousquetaires" d'Alexandre Dumas, dont le contenu (en UTF8) est disponible librement à l'adresse suivante :

`http://www.gutenberg.org/cache/epub/13951/pg13951.txt`

Il est possible de charger l'intégralité de l'oeuvre en mémoire au moyen des instructions python suivantes : [1]

```
with open('pg13951.txt', 'r') as f:
    contenu = f.read()
```

1. Écrire une fonction `nbmot` qui prend en entrée un chemin vers un fichier et un nombre de mot, et retourne le nombre d'occurence de ce mot dans le fichier.

   Testez votre fonction avec le mot `'Artagnan'` et le livre "Les Trois Mousquetaires".
2. Écrivez une fonction `present` qui prend en entrée un chemin vers un fichier, un mot, une position de début incluse et une position de fin exclue (en termes de caractères) et retourne un booléen indiquant si le mot est présent dans la portion de texte concernée.

   Testez votre fonction avec le mot `'Artagnan'` et les caractères 1000 à 2000 du livre "Les Trois Mousquetaires".
3. Écrivez une fonction `position` qui prend en entrée un chemin vers un fichier, un mot, une position de début incluse et une position de fin exclue (en termes de caractères) et retourne un booléen indiquant la position de la première apparition du mot dans la portion de texte concernée (et -1 si le mot n'y apparaît pas).

   Testez votre fonction avec le mot `'Artagnan'` et les caractères 5000 à 8000 du livre "Les Trois Mousquetaires".
4. Écrivez une fonction `lignes` qui prend en entrée un chemin vers un fichier et un mot clé, et retourne un tableau contenant toutes les lignes commençant par le mot clé.

   Testez votre fonction avec le mot `'CHAPITRE'` et le livre "Les Trois Mousquetaires".

Exercice 3. Traitements sur un code python

1. Écrivez une fonction `detabifier` qui prend en entrée un chemin vers un fichier python et retourne une chaîne contenant le contenu du fichier python mais où toutes les tabulations auront été remplacées par 4 espaces.

   Vous testerez votre fonction sur votre programme python lui-même.
2. Écrivez une fonction `tabifier` qui prend en entrée un chemin vers un fichier python et retourne une chaîne contenant le contenu du fichier python mais où toutes les occurrences de 4 espaces auront été remplacées par une tabulation.

   Vous testerez votre fonction sur le résultat de la question précédente.

1. Dans cet exercice, nous ferons ainsi, mais il est conseillé de toujours se demander quel est le coût en espace d'un programme et de le comparer à la mémoire disponible.