# Cours et TD n° 1



Durée: 2h

## Au programme

- Création d'un fichier csv à partir d'une liste de fichiers.
- Automatisation des tâches liées à la manipulation de fichiers.

## Avant propos

**Définition 1 (Chemin absolu et chemin relatif)** *Un* ***chemin absolu*** *commence par la racine du système de fichiers, c'est-à-dire* `/` *dans les systèmes de fichiers UNIX ou par* `C:` *dans les systèmes Windows. Par exemple, C : \Users\Karim\Documents\Script_python\data est le chemin absolu du dossier* `data`*. Un* ***chemin relatif*** *commence à partir du dossier courant. Par exemple, si nous nous trouvons dans le dossier* `Documents` *et nous souhaitons accéder au fichier* `toto.txt` *qui se trouve dans le sous-dossier* `tata`*, alors le chemin relatif est le suivant : Documents\tata\toto.txt*

**Définition 2 (Expressions régulières)** *Les expressions régulières sont une séquence de caractères conçue dans le but de retrouver des patrons dans un texte ASCII, c'est à dire saisi dans une chaîne de caractères avec un ordinateur. Les expressions régulières sont définies dans le package python* `re`*.*

```
import re
```

Par exemple, afin de trouver une séquence de textes on utilise la syntaxe suivante

```
re.match(patron, chaine de caracteres, flags=0)
```

**Définition 3 (Les opérations de lecture et écriture)** *Les opérations de lecture/écriture servent à interagir avec le système de fichiers de votre ordinateur. Un système de fichier est l'environnement qui permet de gérer, stocker, et mettre à disposition ce qui se trouve sur votre disque dur. La sytaxe python pour lire et écrire dans un fichier est présenté dans l'exemple suivant*

```
# -*- coding: utf-8 -*-
fichier_d_entree = open('data/contenu.txt')
outfile = open("output.txt", mode="w")

## variable de type fichier en lecture
print (fichier_d_entree)

## methode pour acceder au contenu
print (outfile)
outfile.write("Ce qu'on ecrit dedans.")


## stockage dans une variable d'un contenu
texte = fichier_d_entree.read()
print(texte)

## fermeture de fichier
```

Durée: 2h

```
fichier_d_entree.close()
outfile.close()
```

### Manipulation des fichiers et chemin vers les fichiers

Nous utilisons le module `os` (*Operating System*, Système d'exploitation) et les différentes fonctions qui y sont définies pour manipuler les dossiers, les fichiers, leurs chemins relatifs et absolus. Par exemple, pour afficher le dossier courant à partir duquel nous avons lancé l'invite de commande python, nous importons d'abord le module `os` et utilisons ensuite la fonction `getcwd()` pour *get current working directory* en anglais ou retourner le dossier de travail courant en français.

```
import os
dossier_courant = os.getcwd()
print (dossier_courant)
```

Un deuxième exemple est celui de la fonction `abspath()` pour *absolute path* en anglais. Cette fonction retourne le chemin absolu d'un fichier et elle se trouve dans le sous-module `path`.

```
import os
os.path.abspath('toto.txt')
```

Veuillez noter que dans l'exemple ci-dessus, il faut remplacer le nom de fichier `toto.txt` par un fichier qui se trouve sur le disque dur de votre ordinateur.
Un dernier exemple est celui de la fonction `isdir()` pour *is directory ?* en anglais. Cette fonction permet de vérifier si le chemin, rentré en argument à la fonction, est celui d'un dossier ou d'un fichier. Si le chemin correspond à celui d'un dossier, alors la fonction retournera `TRUE`, sinon elle retournera `FALSE`.

```
import os
if os.path.abspath('toto.txt'):
    print ("c'est bien un dossier")
else:
    print ("non ce n'est pas un dossier")
```

Veuillez noter que dans l'exemple ci-dessus, il faut remplacer le nom de fichier `toto.txt` par un fichier qui se trouve sur le disque dur de votre ordinateur.

## Exercice 1

Dans cet exercice, nous allons créer un fichier `CSV` qui contiendra deux colonnes. La première est relative au nom du fichier et la deuxième à son identifiant. Nous allons dans une première étape parcourir l'ensemble des fichiers dans un dossier et dans une seconde étape récupérer l'identifiant à partir du nom des fichiers parcouru.

**1)** Ecrire le pseudo-code qui correspond à la description au-dessus. Votre pseudo-code doit contenir une seule fonction qui prend en entrée le chemin du dossier contenant le fichier et donne en sortie un fichier csv avec deux colonnes. Notez que si votre programme ce trouve dans le même endroit que le dossier alors vous pourrez fournir le chemin relatif, sinon vous êtes invité à fournir le chemin absolu.

**2)** Écrire la fonction en python en utilisant les modules `os`, `csv` et les fonctions `open()` et `write()`.

Durée: $2^{h}$

## Exercice 2

Dans cet exercice, nous allons construire un corpus ou une collection de documents à partir d'un fichier texte. Ce fichier contient plusieurs lignes qui correspondent à des tweets. D'abord, et après avoir ouvert le fichier, pour chaque ligne dans ce dernier nous allons créer un nouveau fichier. Cette étape nous donnera un dossier contenant un nombre de fichiers égal au nombre de ligne dans le fichier d'origine. Ensuite, et suivant une certaine proportion que nous allons fournir comme paramètre d'entrée nous allons diviser l'ensemble de fichiers en trois dossiers.

**1)** Écrire le pseudo-code qui correspond à la description au-dessus. Votre pseudo-code doit contenir deux fonctions : la première prendra en entrée le fichier d'origine et donnera en sortie un dossier avec un nombre de fichiers égales au nombre de lignes dans le fichier d'origine. La deuxième fonction prendra comme entrée le chemin relatif ou absolu du dossier fraîchement créé ainsi que trois proportions. C'est-à-dire que la deuxième fonction donnera en sortie trois dossiers avec par exemple 20% des fichiers seront copié dans le premier dossier, 30% des fichiers seront copié dans le deuxième dossier et 50% des fichiers seront copiés dans le troisième dossier.

**2)** Écrire la première fonction en python en utilisant le module `os` et les fonctions `open()` et `write()`.

**3)** Écrire la deuxième fonction en python en utilisant le module `os` et la fonction `copy2()` du module `shutil`. La fonction `copy2()` permet de copier des éléments du système de fichiers.