ANNEXE 1
Rappels sur Python

Rappels sur les listes

```
#On suppose que maListe est une liste
len(maListe) # donne la longueur de maListe
maListe.append(objet) # ajoute l'objet à maListe
#il sera le dernier élément une fois ajouté
#soit autreListe une liste
maListe.append(autreListe)#ajoute tous les éléments de autreListe à maliste
#autreListe est non modifiée
```

Rappels sur les tuples

Un tuple est la donnée d'un n-uplet non mutable (qui ne peut pas être modifié).

```
monTuple=(donnee1,donnee2)
monTuple[0]# permet d'accéder à donnee1
monTuple[1]# permet d'accéder à donnee2
#on peut aussi accéder aux données en faisant
val1,val2=monTuple
print(val1)
>>>donnee1
print(val2)
>>>donnee2
#Pour parcourir un tuple :
for x in monTuple:
    print(x)
>>>donnee1
>>>donnee2
```

Une fonction peut retourner un tuple dont on peut en extraire le contenu directement.
Exemple :

```
def retourneTuple(x,y):
    return (x+y,x-y)
som,diff=retourneTuple(10,3)
print(som)
>>>13
print(diff)
>>>7
```

Rappels sur les chaînes de caractères

Parcours d'une chaîne :

```
mot="test"
for lettre in mot:
    print(lettre)
#Résultat console :
T
e
s
t
```

Rappels sur le Slicing :

```
print(mot[:3])
>>>Bon
print(mot[3:])
>>>jour
```

Rappel sur la fonction len :

```
print(len(mot))
>>>7
```

ANNEXE 2
Présentation des données `exif`

Module exif

Chaque fichier de photo numérique contient des informations appelées `données exif` (`exif` signifie « EXchangeable Image File » ou fichier d'échange de données). Ces informations, créées par l'appareil photo lors de la prise de vue, sont stockées dans le fichier généré par l'appareil.
Voici deux exemples d'informations accessibles :

. Date de la prise de vue.
. Coordonnées GPS du lieu de la prise de vue.

Les `données exif` sont accessibles dans un script Python en important le module `exif`, puis en accédant à `Image` dans ce module. À l'issue du script suivant, la variable `my_image` permet d'accéder aux `données exif` :

```
from exif import Image
with open("photo.jpg", 'rb') as imageFile:
    my_image= Image(imageFile)
```

Afin de faciliter le codage, on donne ci-dessous la fonction `openImage` qui sera utilisée lors de ce sujet :

```
def openImage(nom): #nom est une chaîne de caractères qui représente ici une photo
    with open(nom, 'rb') as imageFile:
        return Image(imageFile)
    imageFile.close()
#Exemple d'appel de cette fonction :
my_image=openImage("photo.jpg")
```

Suite à l'appel de la fonction, la variable `my_image` permet d'accéder aux `données exif`.

La liste des `données exif` qui nous intéressent est :

```
my_image.datetime # une date au format chaîne de caractères
my_image.gps_longitude # un tuple de trois nombres
my_image.gps_longitude_ref # une chaîne de caractères 'E' ou 'W'
my_image.gps_latitude # un tuple de trois nombres
my_image.gps_latitude_ref # une chaîne de caractères 'N' ou 'S'
```

Coordonnées GPS - Règles de conversion

Le module folium, dont on a besoin ici, utilise des coordonnées `gps` au format tuple `(latitude, longitude)` où les deux valeurs sont en décimales signées suivant l'orientation Nord-Sud ou Est-Ouest. Les données `gps` issues des données `exif` d'une image sont au format tuple `(degré, minutes, secondes)`. Il faut donc procéder à une conversion.
On donne la règle de conversion suivante :
1 degré = 1
1 minute = 1/60
1 seconde = 1/3600
L'orientation pour la latitude est `'N'` ou `'S'` et pour la longitude `'E'` ou `'W'`.
Les valeurs décimales sont signées : négative si on est `'W'` ou `'S'` et positive sinon. L'accès à cette orientation est donnée par le code Python ci-dessous :

```
my_image.gps_latitude_ref # donne 'N' ou 'S'
my_image.gps_longitude_ref # donne 'E' ou 'W'
```

Un exemple :

```
my_image.gps_latitude # donne par exemple (36,10,11.78)
my_image.gps_longitude # donne par exemple (115,8,23.38)
my_image.gps_latitude_ref # donne par exemple 'N'
my_image.gps_longitude_ref # donne par exemple 'W'
```

Ce qui donne au format décimal avec l'orientation Nord et Ouest : `36.169939,-115.13983`.

Date de prise de vue
La date de prise de vue est une chaîne de caractères de longueur 19, le format est :

```
#année:mois:jour heure:minute:seconde avec un seul espace entre jour et heure
'1967:06:17 11:15:00'
'2020:12:24 23:59:59'
```

Pour accéder à l'information de date, il suffit d'écrire le script :

```
var_date=my_image.datetime
#var_date sera une chaîne de caractères qui contient les informations de date.
```