Titouan Brierre, TD3, TP5, S2.02

Compte rendu étape 1

CODE :

```
import json
from math import sin, cos, acos, pi
import numpy as np

with open('donneesbus.json') as donneesbus:
    databus = json.load(donneesbus)

noms_arrets = list(databus.keys()) #liste les arrets du dictionnaire

print(noms_arrets) #affiche les arrets

n=len(noms_arrets) #taille de la liste

#liste les voisins d'un arret donné
def voisin(nom_som):
    return databus[nom_som][2]

#reourne l'indice à partir du nom de l'arret
def indice_som(nom_som):
    ind=noms_arrets.index(nom_som)
    return ind

#crée le dictionnaire des arrets et de leurs voisins
dic_bus={}
for nom_arret in databus.keys():
    dic_bus[nom_arret]=voisin(nom_arret)

#crée la matrice d'adjacence des arrets
mat_bus=[[0 for loop in range(n)] for loop in range(n)]
for nom_arret in noms_arrets:
```

```
    for a in voisin(nom_arret):
        mat_bus[indice_som(a)][indice_som(nom_arret)]=1

#retourne le nom d'un arret à partir de son indice
def nom(ind):
    nom_som=noms_arrets[ind]
    return nom_som

#retourne la latitude d'un arret donné
def latitude(nom_som):
    latitude=databus[nom_som][0]
    return latitude

#retourne la longitude d'un arret donné
def longitude(nom_som):
    longitude=databus[nom_som][1]
    return longitude

############################################################
# calcul de la distance entre deux points A et B dont #
# on connait la lattitude et la longitude #
############################################################
def distanceGPS(latA,latB,longA,longB):
 # Conversions des latitudes en radians
 ltA=latA/180*pi
 ltB=latB/180*pi
 loA=longA/180*pi
 loB=longB/180*pi
 # Rayon de la terre en mètres (sphère IAG-GRS80)
 RT = 6378137
 # angle en radians entre les 2 points
 S = acos(round(sin(ltA)*sin(ltB) + cos(ltA)*cos(ltB)*cos(abs(loB-loA)),14))
 # distance entre les 2 points, comptée sur un arc de grand cercle
 return S*RT

#retourne la distance à vol d'oiseau entre deux arrets
def distarrets(arret1,arret2):
```

```
    return distanceGPS(latitude(arret1), latitude(arret2), longitude(arret1),
longitude(arret2))


#retournela distance entre deux arrets voisins donnés
def distarc(arret1,arret2):
    if arret2 in voisin(arret1):
        return distarrets(arret1,arret2)
    else:
        return np.inf


#crée la matrice des poids entre les arrets
poids_bus=[[np.inf for loop in range(n)] for loop in range(n)]
for src in range(n):
    for dest in range(n):
        if mat_bus[src][dest]==1:
            poids_bus[src][dest]=distarc(nom(src),nom(dest))
```

#Aucune difficulté rencontrée