```
import json as js
import pandas as pd
from math import sin, cos, acos, pi
import numpy as np
donneesbus=pd.read_csv('donneesbus.csv',sep=';')

arrets={}
for c in range (len( donneesbus)):
    arrets[donneesbus['arret'][c]]=[float(donneesbus['lattitude']
[c].replace(",",".")),float(donneesbus['longitude']
[c].replace(",",".")),donneesbus['listesucc'][c].strip('],[').replace("'",
'').replace(' ', '').split(',')]

nom=list(arrets.keys())









nom_arrets=[]
for i in arrets:
    nom_arrets.append(i)

def nom(ind):
    """Cette fonction retourne le nom d'un arrêt en fonction par son indice dans la
liste de nom"""
    return nom_arrets[ind]

def indice_som(nom_som):
    """Cette fonction retourne l'indice d'un arrêt en fonction par son nom dans la
liste de nom"""
    return nom_arrets.index(nom_som)

def latitude(nom_som):
    return arrets[nom_som][0]

def longitude(nom_som):
    return arrets[nom_som][1]

def coordonnes(nom_som):
    return latitude(nom_som),longitude(nom_som)

def voisin(nom_som):
    return arrets[nom_som][2]


#Question D

def dic_adjacence(donnees):
    dic={}
#Création d'un dictionnaire vide
    for i in donnees:
#Récupération des clés du dictionnaires "donnees"
        dic[i]=donnees[i][2]
```

```
#Récupération des arrêts succedant de l'arrêts i
    return dic
#Retour du dictionnaire d'adjacence crée


dic_bus=dic_adjacence(arrets)







def lst_adjacence(donnees):
    """Cette fonction renvoie une matrice d'ajacence à partir d'un dictionnaire
d'adjacence
    """
    lst = [[0]*len(donnees) for _ in range(len(donnees))]                #Création
d'un tableau à double entrées initialisé à 0 pour tous les arrêts

    for c in arrets:
#Pour tous les arrêts dans le dictionnaires des arrêts
        succ=voisin(c)
#On enregistre la liste des arrêts succedant de l'arrêt c
        for i in succ:
#Pour tous ses successeurs
            lst[indice_som(c)][indice_som(i)]=1
#On ajoute l'adjacence entre les deux arrêts

    return lst
#Retour de la matrice d'adjacence

mat_bus=lst_adjacence(arrets)




#Question E

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



def distarrets(arret1,arret2):
    """Cette fonction retourne la distance en mètres entre deux arrets grâce a
leurs coordonnées GPS"""
    lat1=latitude(arret1)       #Recuperation de la latitude de l'arret1
```

```
    lat2=latitude(arret2)       #Recuperation de la latitude de l'arret2

    long1=longitude(arret1)     #Recuperation de la longitude de l'arret1
    long2=longitude(arret2)     #Recuperation de la longitude de l'arret2

    return distanceGPS(lat1,lat2,long1,long2) #Appel et retour du résultat de la
fonction de calcule à vol d'oiseau de deux points GPS


def distarc(arret1,arret2):
    """Cette fonction renvoie la distance en mètres entre deux arrêts donnés en
paramètres:
                - Si l'arret2 est un successeur de l'arret1 le retour sera la
distance a vol d'oiseau entre ces deux arrêts
                - Si l'arret2 n'est pas un successeur de l'arret1 le retour sera
une distance infinie
    """
    if arret2 in voisin(arret1) :               #Si l'arret2 est un successeur de
l'arret1
        res=distarrets(arret1,arret2)           #Appel de la fonction calculant la
distance des deux arrêts
    else:                                       #Sinon
        res=np.Inf                              #La distance est dite Infinie
    return res

print(distarrets('NOVE','FINE'))



# Question F
poids_bus=[x[:] for x in mat_bus]                   #Copie profonde de la matrice
d'adjacence qui possède le même ordre


for i in range (len(poids_bus)):                    #On parcours la matrice
poids_bus
    for y in range(len(poids_bus[i])):
        poids_bus[i][y]=distarc(nom(i),nom(y))      #On y inscrit la distance
entre les deux arrêts selectionnés [floatant ou Infini]



for i in range (len(poids_bus)):                    #On parcours la matrice
poids_bus
    for y in range(len(poids_bus[i])):
        print(poids_bus[i][y],mat_bus[i][y],'\n')
```