```python
#! /usr/bin/env python
# -*- coding: utf-8 -*-
from math import sin,sqrt,cos,asin,pi
fichier = "OuCest.kml"


# Question 3
def is_char_in_text(ligne,lettre):
    for i in range(len(ligne)):
        if ligne[i]==lettre:
            return True
    return False

# Question 4
def is_word_in_text(ligne,mot):
    for i in range(0,len(ligne)-len(mot)+1):
        if ligne[i:i+len(mot)]==mot:
            return True
    return False

# Question 5
def Affiche_GPS_n(fichier,n):
    # Ouverture du fichier
    fid=open(fichier,'r')
    i=0
    for ligne in fid.readlines() :
        if "when" in ligne and i<n:
            print(ligne)
        if "gx:coord" in ligne and i<n:
            print(ligne)
            i=i+1
        elif i>=n:
            break
    fid.close()

def coord(ligne):
    if "gx:coord" in ligne :
        lat = ligne[ligne.find(">")+1:ligne.find(" ")]
        ligne = ligne[ligne.find(" ")+1:]
        lon = ligne[:ligne.find(" ")]
        ligne = ligne[ligne.find(" ")+1:]
        alt=ligne[:ligne.find("<")]
        return [float(lat),float(lon),float(alt)]
    else:
        return []

def heure(ligne):
    if "when" in ligne :
        heure = ligne[ligne.find("T")+1:ligne.find("Z")]
        h = heure.split(":")
        return [float(h[0]),float(h[1]),float(h[2])]
    else :
        return None

def parse_kml(fichier):
```

```
    "Fonction permettant de lire le fichier kml."
    fid=open(fichier,'r')
    kml = []
    for ligne in fid.readlines() :
        if "when" in ligne:
            h = heure(ligne)
        if "gx:coord" in ligne:
            c = coord(ligne)
            kml.append([h[0],h[1],h[2],c[0],c[1],c[2]])

    fid.close()
    return kml

def orthodromie(loA,laA,loB,laB):

    R = 6371
    loA = loA*pi/180
    loB = loB*pi/180
    laA = laA*pi/180
    laB = laB*pi/180

    d=2*R*asin(sqrt(
        (sin((laB-laA)/2))**2 +
                cos(laA)*cos(laB)*(sin((loB-loA)/2))**2))
    return d

def distance(kml):
    "Génération d'un tableau contenant la distance entre deux points"
    tab_distance=[]
    for i in range(len(kml)-1):
        tab_distance.append(orthodromie(kml[i][3],kml[i][4],kml[i+1][3],
    return tab_distance

def distance_totale(tab):
    "Calcul de la distance totale"
    distance = 0
    for i in range(len(tab)):
        distance = distance + tab[i]
    return distance


def altitude(kml):
    "Génération d'un tableau contenant les altitudes."
    alti=[]
    for i in range (len(kml)):
        alti.append(kml[i][5])
    return alti

def distance_cumulee(tab):
    " Génération d'un tableau contenant les distances cumulées"
    cumul=[0]
    dist = 0
    for i in range (len(tab)):
        dist = dist+tab[i]
        cumul.append(dist)
```

```
   return cumul

k=parse_kml(fichier)
t=distance(k)
dist = distance_totale(t)
cumul= distance_cumulee(t)
alt=altitude(k)
print(len(cumul))
print(len(alt))

import matplotlib.pyplot as plt
import numpy as np
p1=plt.plot(cumul,alt,linewidth=3)
plt.grid(True, which="both", linestyle="dotted")
plt.show()
```