Transmission des coordonnées GPS : trame NMEA - Correction

- **Contenu** : Protocole NMEA 0183.
- **Capacités attendues** : Décoder une trame NMEA pour trouver des coordonnées géographiques.

D'après Luc Vincent

Introduction

Les différents composants d'un appareil électronique (ex : un téléphone mobile) communiquent par des protocoles normalisés. Ainsi, les puces GPS qui effectuent les calculs de positionnement envoient leurs résultats présentés suivant une trame normalisée : la trame NMEA 0183. Le développeur d'un dispositif souhaitant interroger un GPS sait qu'il pourra exploiter cette trame pour obtenir des informations de date et de position.

La norme NMEA 0183

La norme NMEA est le protocole de transmission des données GPS. Ces données sont transmises sous la forme de trames. Chaque trame commence par le caractère $ et se compose de plusieurs éléments séparés par des virgules. Voici un exemple de trame :

```
$GPGGA, 064036.289, 4403.1410, N, 0502.745, E, 1, 04, 3.2, 200.2,
M, , , , 0000*0E
```

Les deux premiers caractères correspondent à l'identifiant du récepteur : ici GP pour Global Positioning System. Les trois lettres suivantes correspondent à l'identifiant de la trame : GGA pour GPS Fix et Date. C'est la trame la plus courante.

Eléments de la trame GGA

Décomposons maintenant cette trame selon les premiers éléments qui la composent :

- **GPGGA** : type de la trame
- **064036.289** : heure d'envoi de la trame, ici 06h 40min 36,289s (UTC)
- **4836.5375, N** : latitude Nord, ici 48°36,5375' (en DM, degrés minutes)
- **00740.9373, E** : longitude Est, ici 7°40,9373' (en DM également)
- **1** : type de positionnement (1 pour le positionnement GPS)
- **04** : nombre de satellites utilisés
- **3.2** : précision horizontale
- **200.2, M** : altitude, ici 200 mètres

Les notations DMS, DM et DD

Généralement, on exprime les coordonnées géographiques dans le système sexagésimal, noté DMS pour degrés, minutes, secondes. Par exemple 49°30'30'' pour 49 degrés, 30 minutes et 30 secondes. Une minute d'angle vaut 1/60 degrés tandis qu'une seconde d'angle vaut 1/3600 degrés.

Il est également possible d'utiliser les unités DM (Degré Minute) ou DD (Degré décimal) :

- En DMS : 49°30’30’’
- En DM : 49°30,5’
- En DD : 49,5083° (généralement avec quatre décimales)

Question 1

Vérifier par un calcul que la latitude 44°03.1410’ (DM) de la trame NMEA donnée en exemple en début d’activité correspond à 44°03’08.46” (DMS).

Sachant que 1′ = 60″, alors 0.1410′ = 60 × 0.1410 = 08.46″. .....

.....

.....

Extractions des données contenues dans une trame avec Python

Voici une vue des résultats de quelques instructions Python obtenues depuis la console.

```
>>> ligne = "nom,prenom,age,17"
>>> element = ligne.split(",")
>>> element
['nom','prenom','age','17']
>>> type(element)
<class 'list'>
>>> element[1]
'prenom'
>>> prenom = element[1]
>>> prenom[2:4]
'en'
>>> type(element[3])
<class 'str'>
>>> int(element[3])
17
>>> float(element[3])
17.0
>>>
```

Question 2

D’après ces résultats, quelle instruction en python permet d’obtenir une liste nommée `attribut` à partir d’une chaîne de caractères nommée `trame` ?

```
trame = "$GPGGA, 064036.289, 4403.1410, N, 0502.745, E, 1, 04, 3.2, 200.2, M, , , , 0000*0E"
```

.....

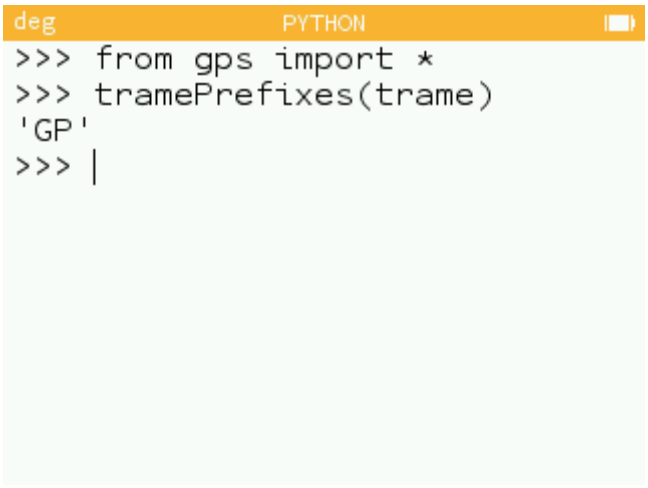
Il faut saisir l’instruction : `attribut = trame.split(",")` .....

Question 3

Ecrire la fonction `tramePrefixes(trame)` qui reçoit une trame complète et renvoie l’identifiant du récepteur, c’est-à-dire les deux premières lettres du type de la trame (premier élément après le caractère $). Sur la trame d’exemple, la fonction doit renvoyer “GP”.

```python
def tramePrefixes(trame):
    recepteur = trame[1:3]
    return recepteur
```

On teste ensuite la fonction avec la trame d'exemple.



Question 4

Modifier cette fonction pour qu'elle renvoie le nom de l'équipement qui a émis la trame. On utilisera les correspondances suivantes :

- BD ou GB : Beidou *(Chine)*
- GA : Galileo *(Europe)*
- GP : GPS *(Etats-Unis)*
- GL : GLONASS *(Russie)*

```python
def tramePrefixes(trame):
    recepteur = trame[1:3]
    if recepteur == "GD" or recepteur == "GB":
        equipement = "Beidou"
    elif recepteur == "GA":
        equipement = "Galileo"
    elif recepteur == "GP":
        equipement = "GPS"
    elif recepteur == "GL":
        equipement = "GLONASS"
    return equipement
```

On teste alors la fonction.

```
deg                    PYTHON
>>> from gps import *
>>> tramePrefixes(trame)
'GPS'
>>>
```

Question 5

Ecrire une fonction `ggaUtc(trame)` qui reçoit une trame complexe et renvoie l'heure en h, min, s.

```python
def ggaUtc(trame):
    #on transforme la trame en liste
    Attribut = trame.split(',')
    #on sélectionne l'élément qui correspond à l'heure
    time = attribut[1]
    utc = time[:2] + " h " + time[2:4] + " min " + time[4:] + " s"
    return utc
```

```
deg                    PYTHON
>>> from gps import *
>>> ggaUtc(trame)
06
'06 h 40 min 36.289 s'
>>>
```

Question 6

Ecrire une fonction `ggaLat(trame)` qui reçoit une trame complète et renvoie la latitude convertie en DMS.

```python
def ggaLat(trame):
     attribut = trame.split(',')
     lat = attribut[2]
     lat_deg = lat[:2]
     lat_min = lat[2:4]
     lat_sec = str(float(lat[4:])*60)
     latDMS = lat_deg+" D "+lat_min+" M "+lat_sec+" S "
     return latDMS
```

A retenir

Pour définir sa position sur la Terre, on utilise le plus souvent les coordonnées géographiques : la latitude, la longitude et l'altitude. Pour la latitude et la longitude on rencontre trois notations :

- Degré Minute Seconde (DMS)
- Degré Minute (DM)
- Degré décimal (DD)

Les récepteurs "GPS" fournissent la localisation sous une forme normalisée facilement décodable, par exemple selon le protocole NMEA 0183 (National Marine Electronics Association).

Aller plus loin : le code du prof !

Voici quelques essais de code, l'idée étant de coder un programme permettant :

- L'analyse d'une trame NMEA donnée pour en extraire l'heure et les coordonnées, et les afficher à l'écran.
- L'ouverture du navigateur web et la représentation de l'emplacement sur une carte (Actuellement disponible pour Google Maps, OpenStreetMap ~~et Géoportail~~)
- L'analyse de deux frames successives pour déterminer la vitesse de déplacement du récepteur ($v = d/\Delta t$).
    - Approximations : distance à vol d'oiseau et altitude constante...

Vous n'êtes pas obligé d'étudier ce code dans les moindres détails, je le propose d'ailleurs « en l'état », mais je répondrais à vos questions à la prochaine séance.

```
from math import *
import webbrowser as wb
import datetime
#import math

# Calcul de l'heure
def calculHeure(trame):
    Liste=trame.split(",")  # Découpage de la trame
    heure=Liste[1] #085410.00
    h=int(heure[0:2])  # On récupère depuis le caractère 0 inclus jusqu'au 2 exclu.
    m=int(heure[2:4])
    s=float(heure[4:]) # Du 4eme à la fin
    s=floor(float(heure[4:]))  # Du 4eme à la fin, arrondi à l'entier inférieur
```

```
    us=floor(1000000*(float(heure[4:])-floor(float(heure[4:]))))
#microsecondes
    instant = datetime.datetime(1, 1, 1, h, m, s, us) #Définition
d'un instant
    return instant

# Conversion coordonnées
def conversion(coord):
    ''' Conversion d'une coordonnée de trame degrés minute (sous la
forme DDMM,MMMM) en degrés décimaux DD,DDDD'''
    deg = floor(coord/100) # floor (=plancher) : arrondir à
l'entier inf
    m = round(coord-deg*100,4) # minutes arrondies à 2 chiffres
après la virgule
    s = round(60*(coord-floor(coord)),2) # secondes
    return round(deg+m/60,4) # Les degrés + les minutes converties
en degrés

# Calcul des coordonnées en degrés décimaux
def calculCoordonnees(trame):
    Liste=trame.split(",")
    lat=conversion(float(Liste[2]))
    lat_N_S= Liste[3]
    long=conversion(float(Liste[4]))
    long_E_W=Liste[5]
    coordonnées = [lat,lat_N_S,long,long_E_W]
    return coordonnées

# Ouverture cartes
def affichageCarte(coordonnées):
    reponse = str(input("Voulez-vous afficher la position sur une
carte? (Oui (O) / Non (N))") or "N").upper()
    if reponse == "OUI" or reponse == "O":
        print("Ouverture des cartes avec le navigateur par défaut
de votre ordinateur.")
        # Sur OSM, GMaps et Géoportail, latitude S et longitude O
sont négatives, en degrès décimaux.
        if coordonnées[1] == "S":
            coordonnées[0] = -coordonnées[0]
        if coordonnées[3] == "O":
            coordonnées[2] = -coordonnées[2]
        # Ouverture cartes

wb.open("https://www.openstreetmap.org/#map=19/"+str(coordonnées[0]
)+"/"+str(coordonnées[2]))

wb.open("https://duckduckgo.com/?q=!gm%20"+str(coordonnées[0])+"%20
"+str(coordonnées[1])+",%20"+str(coordonnées[2])+"%20"+str(coordonn
ées[3]))

wb.open("https://www.geoportail.gouv.fr/carte?c="+str(coordonnées[0
])+","+str(coordonnées[2])+"&z=19&l0=ORTHOIMAGERY.ORTHOPHOTOS::GEOP
ORTAIL:OGC:WMTS(1)&permalink=yes") #semble buggé ?

# Calcul de vitesse entre 2 trames
def calculVitesse(trame1, trame2):
```

```
    coord1 = calculCoordonnees(trame1)
    coord2 = calculCoordonnees(trame2)

    #Calcul de la distance
    deg2rad = 0.017453292519943
    distance = 6369000.0*acos(sin(coord1[0]*deg2rad)*sin(coord2[0]*deg2rad)+cos(coord1[0]*deg2rad)*cos(coord2[0]*deg2rad)*cos((coord2[2]-coord1[2])*deg2rad))
    distance = round(distance,3)
    print("Distance entre les deux trames:",distance,"mètres.")

    #Calcul du temps
    instant1 = calculHeure(trame1)
    instant2 = calculHeure(trame2)

    #print(instant1.totalsecondes())
    durée = (instant2-instant1).total_seconds() #Diff entre 2 instants : durée. La méthode ajoutée permet de récupérer le total en secondes.
    print("Durée entre les deux trames:",round(durée,3),"secondes.")

    #Calcul de la vitesse
    vitesse = round(distance / durée,2)
    print("Vitesse entre les deux trames:",vitesse,"m/s","(ou",vitesse * 3.6,"km/h).")



##########################################
# Début partie principale
##########################################

#Quelques trames exemples
trame1 = "$GPGGA,085410.0025,4403.1410,N,0502.745,E,1,8,1.092,124.760,M,,M,0,*75" # MP
trame2 = "$GPGGA,085550.0030,4403.4008,N,0502.5764,E,1,8,1.092,124.760,M,,M,0,*75" # Auzon
trame3 = "$GPGGA,123519,4357.238008,N,0448.281592,E,1,08,0.9,545.4,M,46.9,M,,*42" # Pont St Bénézet
trame4 = "$GPGGA,123519,0513.4280,N,5246.2960,O,1,08,0.9,545.4,M,46.9,M,,*42" # Centre spatial guyanais (Fusée Ariane !)
trame5 = "$GPGGA,064036.289,2958.7333,N,3108.0333,E,1,04,3.2,200.2,M,,,,0000*0E" # Pyramide de Khéops
```

```
trame = eval("trame"+str(input("Bonjour, choisissez une trame à analyser (de 1 à 4): ") or "1")) # évalue la chaine telle une variable
print("A l'heure : ", end="")
instant = calculHeure(trame)
print(instant)
print("Le récepteur est à la position : ",end="")
coordonnées = calculCoordonnees(trame)
print(str(coordonnées[0])+"°"+str(coordonnées[1])+',',str(coordonnées[2])+"°"+str(coordonnées[3])+".")
affichageCarte(coordonnées)

print("\nPour finir, évaluation de la vitesse entre les trames 1 et 2 :")
calculVitesse(trame1,trame2) #MP-Auzon :
print("Fin du programme.")


# Détails norme NMEA 0183 : https://fr.wikipedia.org/wiki/NMEA_0183
```