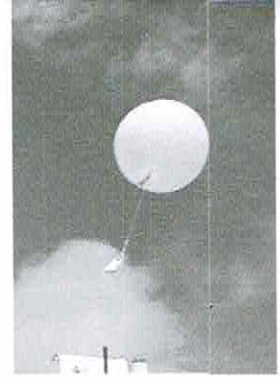


Un ballon-sonde (utilisé en météorologie et en astronautique) est un ballon à gaz libre utilisé pour faire des mesures locales dans l'atmosphère. Chaque mesure réalisée est horodatée et géolocalisée par GPS.



À la fin de leur vol, les radiosondes redescendent vers la terre, se posent « n'importe où » et sont « abandonnées » à leur sort. C'est pourquoi vous pouvez en trouver. Diane Delstar est une radio amatrice appartenant à un club de passionnés. Le club récupère les ballons équipés de leur radiosonde et répertorie toutes les infos de leur atterrissage sur la terre ferme dans une base de données consultable sur un site web (source : <https://sondehub.org>). Le programme, écrit par Diane, lui permet de récupérer les données relevées par les sondes. La variable `enregistrement` fait référence à un dictionnaire qui contient un ensemble de données pour une sonde.

Exemple :

```
enregistrement={
    'num_serie': 623, 'altitude': 1150.0,
    'datetime': '2024-06-27T23:36:01.000000Z',
    'latitude': 38.38825, 'longitude': 27.09004
}
```

1. Écrire la ligne de code qui, dans la console, permet d'afficher la valeur 38.38825 de la variable `enregistrement`.

Les valeurs dont la clé est `'datetime'` sont des chaînes de caractères.

Exemple : `'2024-06-27T23:36:01.000000Z'` est une chaîne de caractères qui comprend la date au format année-mois-jour(`'2024-06-27'`), suivie de l'indicatif (`'T'`), puis de l'heure (`'23:36:01.000000'`) et enfin d'un indicatif de fuseau horaire (`'Z'`).

Diane écrit la fonction `nettoyage_datetime` qui prend en paramètre une chaîne de caractères correspondant à la clé `'datetime'` au format ISO 8601.

```

1 def nettoyage_datetime(chaine):
2     date=''
3     horaire=''
4     for i in range(10):
5         date=date+chaine[i]
6     for i in range(8):
7         horaire=horaire+chaine[i+11]
8     return date,horaire

```

2. Écrire et expliquer ce que renvoie l'appel de la fonction ci-dessous:

```
nettoyage_datetime ('2024-06-27T23:36:01.000000Z')
```

Tous les enregistrements de données de plusieurs radiosondes sont stockés dans la variable `frames` de type `list` dont un extrait est donné ci-dessous :

```

1 frames=[
2     {'num_serie': 623, 'altitude': 620.1,
3     'datetime': '2024-07-01T01:58:55.000Z',
4     'latitude': 43.20223, 'longitude': -72.05708},
5     {'num_serie': 500, 'altitude': 6375.75,
6     'datetime': '2024-07-01T23:35:32.000Z',
7     'latitude': -20.8759, 'longitude': 55.58805},
8     {'num_serie': 623, 'altitude': 622.6,
9     'datetime': '2024-07-01T01:59:01.000Z',
10    'latitude': 43.20224, 'longitude': -72.05711},
11    {'num_serie': 700, 'altitude': 60000.0,
12    'datetime': '2024-07-01T11:51:23.000000Z',
13    'latitude': 40.87873, 'longitude': 29.09587}
14 ]

```

3. Donner les valeurs renvoyées par les appels `len(frames)` et `len(frames[1])`.

Une radiosonde peut monter au maximum à une altitude de 35 000 mètres et, bien sûr, ne peut atteindre une altitude de valeur négative. La fonction `detecter_anomalie` prend en paramètre un élément d'une liste analogue à `frames` et renvoie un booléen indiquant si la clé 'altitude' est associée à une valeur négative ou à une valeur strictement supérieure à 35 000 mètres.

Exemple :

```

>>> detecter_anomalie({'num_serie': 700, 'altitude': 60000.0,
'datetime': '2024-07-01T11:51:23.000000Z', 'latitude':
40.87873, 'longitude': 29.09587})
True

```

4. Écrire, en langage Python, la fonction `detecter_anomalie`.

Diane a besoin d'obtenir la liste des numéros de séries des radiosondes. Elle écrit donc la fonction `liste_num_serie`.

Une sonde ne doit figurer qu'une seule fois dans la liste représentée par son numéro de série. Exemple d'appel de la fonction `liste_num_serie` avec comme paramètre la variable `frames` ci-dessus :

```
>>> liste_num_serie(frames)
[623, 500, 700]
```

On rappelle que l'opérateur `in` permet de réaliser un test d'appartenance.

Exemple :

```
>>> fruits=['Banane','Pomme','Poire']
>>> 'Banane' in fruits
True
>>> 'Orange' in fruits
False
```

5. Écrire, en langage Python, la fonction `liste_num_serie`.

En exploitant les enregistrements de données d'une même sonde, Diane décide d'en déterminer la distance totale parcourue. La fonction `distance_haversine` :

- prend en paramètre deux tuples de coordonnées (latitude, longitude) ;
- renvoie la distance, en kilomètres, entre les deux points.

Exemple : Les points `point1` et `point2` ont les coordonnées géographiques :

Point 1 : `latitude1 = 46.815`, `longitude1 = 6.943`

Point 2 : `latitude2 = 47.049` et `longitude2 = 7.52828`

```
>>> point1 = (46.815,6.943)
>>> point2 = (47.049,7.52828)
>>> distance_haversine(point1, point2)
51.5
```

La fonction `distance_totale` prend en paramètre une liste de tuples de coordonnées (formées d'une latitude et d'une longitude) constituant le déplacement de la sonde. Elle additionne les distances entre les points successifs et renvoie la distance totale parcourue. Exemples :

```
>>> deplacement = [(46.8125, 6.9433), (46.91498, 7.23039),
(47.00661, 7.48054)]
>>> distance_totale(deplacement)
46.1
>>> deplacement = [(46.8125, 6.9433), (46.91498, 7.23039),
(47.00661, 7.48054), (47.2512,7.5201)]
>>> distance_totale(deplacement)
73.5
```

6. Compléter la fonction `distance_totale` :

```
1 def distance_totale(dep):  
2     total = 0  
3     for i in range(...):  
4         ...  
5     return total
```