TP #4 - Correction

Installation et exploitation de librairies tierces

I. Création d'un nouveau projet

Ouvrez l'IDE de votre choix (VSCode ou PyCharm), et initiez un projet vierge (tp4).

Créez un fichier de type python (tp4.py).

II. Librairie Requests

Citi Bike est le système de vélos en libre-service de la ville de New York. Une API est disponible, permettant d'obtenir des informations en temps réel :
https://gbfs.citibikenyc.com/gbfs/fr/station_information.json

https://gbfs.citibikenyc.com/gbfs/fr/station_status.json

Exercice 1

Installez la librairie Requests, qui vous permettra de faire des requêtes web afin d'interroger cette API.

```
pip install requests
```

Exercice 2

En exploitant l'API mise en place (station_information.json), affichez le nom de toutes les stations avec leur capacité actuelle.

```python
import requests

r =
requests.get('https://gbfs.citibikenyc.com/gbfs/fr/station_information.json')

data = r.json()
stations = data['data']['stations']

for station in stations :
    print(station['name'] , ' : ' , station['capacity'])
```

```
St Johns Pl & Saratoga Ave  :  0
4567.07  :  0
Morton St & West St  :  23
Bergen St & Kingston Ave  :  0
Sterling Pl & Schenectady Ave  :  0
Pacific St & Thomas S. Boyland St  :  0
JC Medical Center  :  21
Carroll St & Rochester Ave  :  0
College Ave & E 169 St  :  19
Lafayette Ave & Stuyvesant Ave  :  24
```

Exercice 3

En exploitant l'API mise en place (station_status.json), affichez :

- Le nombre de stations en service (active)
- Le nombre de stations HS (out_of_service)
- Le ratio de stations en service

```python
import requests

r = requests.get('https://gbfs.citibikenyc.com/gbfs/fr/station_status.json')

data = r.json()
stations = data['data']['stations']

out_of_service = 0
active = 0
for station in stations :
    if station['station_status'] == 'out_of_service' :
        out_of_service = out_of_service + 1
    elif station['station_status'] == 'active' :
        active = active + 1

total = active + out_of_service
print("Ratio de stations en service : ", active / total)
```

```
Ratio de stations en service :  0.9459308807134894
```

Exercice 4

En exploitant l'API mise en place (station_status.json), affichez :

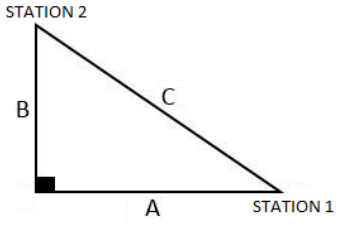
- Le nombre de vélos électriques
- Le nombre de vélos classiques
- Le ratio de vélos électriques

```python
import requests

r = requests.get('https://gbfs.citibikenyc.com/gbfs/fr/station_status.json')

data = r.json()
stations = data['data']['stations']

ebikes_available = 0
bikes_available = 0
for station in stations :
    ebikes_available = ebikes_available + station['num_ebikes_available']
    bikes_available = bikes_available + station['num_bikes_available']

total = ebikes_available + bikes_available
print("Ratio de vélos électriques : ", ebikes_available / total)
```

```
Ratio de vélos électriques :  0.08626536668079694
```

III. Librairie Math

Nous souhaitons trouver quelles sont les stations les plus éloignées. Pour cela, nous utiliserons la célèbre formule du théorème de Pythagore : $c = \sqrt{a^2 + b^2}$



Exercice 5

Trouvez l'algorithme permettant de trouver quelles sont les 2 stations les plus éloignées.

```python
import requests
import math

r =
requests.get('https://gbfs.citibikenyc.com/gbfs/fr/station_information.json')

data = r.json()
stations = data['data']['stations']

distance_max = 0
st1 = ""
st2 = ""
for station1 in stations :
```

```
    for station2 in stations :
        a = station1['lat'] - station2['lat']
        b = station1['lon'] - station2['lon']
        distance = math.sqrt(a * a + b * b)
        if distance > distance_max:
            distance_max = distance
            st1 = station1['name']
            st2 = station2['name']

print(st1)
print(st2)
```

```
Union St
4567.07
```