API REST d'accès aux données

Planche de TD pour un cours dispensé à l'université de Rennes 2

1 Rappel : organisation de votre code

Pour ce TD, vous créerez un nouveau fichier `td_api_rest.py`. Dans ce fichier, votre code sera organisé de la manière suivante :

```
# Imports
import urllib.request
import json
import datetime

# Fonctions
def [...]

# Tests
[...]
```

Notamment, vous définirez vos fonctions en début de fichier et les appels seront listés en fin de fichier. De cette manière, vous pourrez, d'une question à l'autre, réutiliser les fonctions déjà codées au besoin.

2 Énoncé

1. Se rendre sur le site de la STAR et trouver l'API indiquant les prochains passages de métro rennais.
2. Écrire une fonction qui prenne en entrée une chaîne de caractères représentant une date et retourne une date en ignorant la "timezone" (fuseau horaire, partie à partir du symbole `"+"` dans la chaîne de caractères). La date devra être de la même forme que dans l'exemple suivant : `"2019-11-23T09:01:52+00:00"`. **Attention:** vous devrez considérer deux cas de figure différents :
   a. si la fin de la chaîne de caractères est `"+00:00"` : il faudra ajouter 1h à la date extraite car le fuseau horaire n'est pas le bon ;
   b. si la fin de la chaîne de caractères est `"+01:00"` : la date sera considérée comme correcte.

3. Écrire une fonction qui fasse une requête à l'API STAR et retourne la liste de tous les passages de métros à venir ayant pour valeur `Temps réel` pour l'attribut `precision` (limiter le résultat à 100 lignes). La liste retournée stockera des dictionnaires composés de 3 clés : `"depart"` (contenant l'heure de départ au format `datetime`), `"destination"` et `"nomarret"`. **Attention #1.**, contrairement à ce qui vous a été enseigné en CM, le filtrage sur les métros ayant la valeur `Temps réel` pour l'attribut `precision` devra se faire dans le code Python car si vous le faites via l'API STAR, les résultats obtenus ne sont pas fiables. **Attention #2.**, pour certains passages, l'attribut `"depart"` n'existe pas : ces passages doivent donc être ignorés
4. Écrire une fonction qui prenne en entrée une liste de dictionnaires tels que ceux retournés par la question précédente et un délai `t` en minutes et qui retourne une version de cette liste privée des métros qui n'arrivent pas dans les `t` minutes après l'instant présent. Tester cette fonction en affichant la liste des prochains passages de métro dans les 10 minutes à venir.