TD 10-11 : Projet Rennes2 Express

1 Préambule

Pour ce projet, vous devrez travailler par groupes de 2.

Le projet sera réalisé essentiellement lors des deux dernières séances de TD, et votre présence lors de ces séances de TD est obligatoire.

La date limite de rendu est indiquée sur CURSUS dans l'espace de dépôt.

2 Rennes2 Express

Lors de l'émission télévisée Rennes2 Express, 8 équipes s'affrontent pour tenter de remporter un grand prix ! Les concurrents s'élancent de Marseille le lundi 13 novembre, et ils ont 4 jours pour rallier l'Université Rennes 2, le plus rapidement possible.

Chaque journée de jeu se déroule de la manière suivante :

— Une étape le matin, entre 8h et 12h, dont le déplacement se fait à pieds ou en voiture.

— Une épreuve le midi, entre 12h et 14h. Les équipes s'affrontent, et l'équipe gagnante obtient le droit d'utiliser le train pour l'étape de l'après-midi.

— Une étape l'après-midi à partir de 14h, dont le déplacement se fait à pieds ou en voiture, sauf pour l'équipe gagnante de l'épreuve du midi qui peut utiliser le train.

Au début et à la fin de chaque étape, une borne est disposée sur le parcours, et les candidats doivent badger pour valider leur temps de parcours de l'étape.

Ainsi, le déplacement en train est interdit, en dehors du cas de l'équipe gagnante. On considère qu'une équipe qui parvient à se déplacer plus rapidement que le temps de trajet d'une voiture a triché, par exemple en empruntant un TGV.

Lors de l'édition 2023, des bruits courent... Certaines équipes auraient triché, en empruntant un TGV alors qu'elles n'y étaient pas autorisées !

La société de production de Rennes2 Express nous fournit les informations suivantes :

— La liste des équipes

— La liste des bornes de début et de fin de parcours, avec leur position GPS.

— La liste de tous les enregistrements des équipes sur les bornes, avec l'horaire précis de validation.

— La liste des épreuves du midi, et le nom des vainqueurs.

Votre rôle sera de déterminer quelles sont la ou les équipes qui ont triché !

3 Travail attendu

Pour valider votre projet, vous devrez fournir un script Python produisant les résultats ci-dessous. Différents niveaux de rendus sont proposés, et seront valorisés dans la note finale.

Niveau 0 Afficher le nom de la ou des équipes qui ont triché, c'est à dire qui ont pris un TGV alors qu'elles n'avaient pas le droit.

Niveau 1 Afficher le nom de l'équipe ganantes, c'est à dire l'équipe qui a mis le moins de temps cumulé sur l'ensemble des étapes.

Niveau 2 Afficher l'ordre du classement complet des équipes (en enlevant celles qui ont triché), selon le temps cumulé des étapes.

Niveau 3 Proposer d'autres classements : journalier, à points...

Bonus Produire une carte des étapes et des épreuves.

Il est également attendu de fournir un code clair et lisible, qui limite les traitements inutiles et les requêtes redondantes.

4 Implémentation en Python

Pour mener à bien votre mission, vous pourrez utiliser (outre votre intelligence) :

- le package graphh pour calculer des temps de trajet théoriques en voiture entre deux points ;
- le package requests pour interroger la base des équipes `http://my-json-server.typicode.com/alemaitr/python_opendata_l2/equipes` et des épreuves `http://my-json-server.typicode.com/alemaitr/python_opendata_l2/epreuves`
- le package csv et les DataFrame pandas pour l'analyse des deux fichiers csv de la production (disponibles sur Cursus).
- le package datetime pour la gestion des dates et des durées.

5 Rendu de devoir

Votre rendu se fera sous la forme d'un fichier Python. Ce fichier devra être nommé td10-11.py et contenir les noms et numéros étudiant de tous les membres du groupe commentés, en en-tête du fichier, comme dans l'exemple suivant :

```
# 22000002 Paul Machin
# 22000227 Yolène Truc

import ...
```

Les fichiers déposés sur CURSUS ne devront surtout pas contenir vos clés d'API. Celles-ci devront être lues par votre programme dans un fichier credentials.json (que vous ne fournirez pas pour ne pas divulguer votre clé d'API) au format :

```
{
    "graphhopper": "..."
}
```