**Itinéraire train Bretagne**

Objectif - un programme qui demande à l’utilisateur la gare de départ et la gare d’arrivée, et qui détermine l’itinéraire le plus court pour les gares de Bretagne. La saisie se fera de façon graphique (TKinter).

Une fois l’itinéraire réalisé, le programme génèrera un fichier html sur lequel seront mentionnés les gares de départ et d’arrivée et l’itinéraire affiché.

Le programme final sera sous Edupython. *Il est recommandé de faire plusieurs fichiers, reliés par import*

**Principe**

Vous travaillerez par groupe de 4 ou 5. Dans l’équipe, il y aura une personne chargée de coordonner l’équipe appelé scrum master. (chef de mêlée).

Au début de chaque séance, on déterminera une tâche précise pour chacun des développeurs, pendant ¼ d’heure max.

Le scrum master va aider les différents développeurs au cours de la séance.

**A la fin de la séance, chaque développeur devra avoir effectué une tâche, qu’il peut présenter osus forme papier ou sous forme de code.**

Cette tâche pourra être une portion de code, une explication, une analyse, ….quelque chose de précis.

La séance suivante commencera par un débriefing organisé par le scrum master.

**Exemples de tâches (peuvent être encore décomposées en tâches plus simples)**

- déterminer les champs du fichier csv qui seront utiles pour le programme

- déterminer comment trouver les distances entre gares d’une même ligne dans un tableau + écrire le code qui permet de le faire.

- à partir du tableau de départ, déterminer les différentes lignes de train, comme la ligne Paris-Brest, Landerneau- Quimper, ... ;

- à partir du tableau de départ, générer une liste des gares de Bretagne

- récupérer tous les sommets = villes

- récupérer pour chaque sommet les liens avec les autres villes

- réutiliser dijkstra de la fiche 3 et l’implémenter dans un fichier à part

- récupérer les coordonnées GPS de chaque ville

- afficher l’itinéraire avec folium

…..

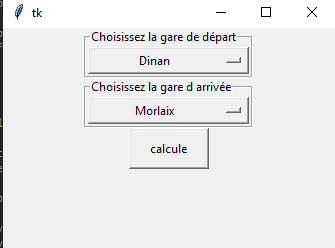
**CAHIER DES CHARGES**

**Structure du programme**

* A partir des données du csv, on devra faire un graphe sous la forme d’un dictionnaire, dont les clés sont les villes, et dont les valeurs sont les voisins. (même structure que la fiche 3). Ce graphe sera utilisé pour déterminer les distances avec dijkstra.
* Un fichier util\_dijkstra, qui devra contenir les fonctions :

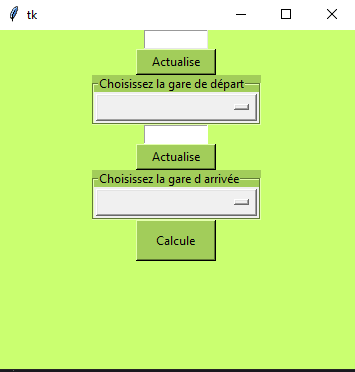
dijkstra(ville\_depart) : calcule la distance depuis la ville de départ avec les autres villes

calcule\_itineraire(ville\_depart, ville\_arrivee) renvoie l’itinéraire sous la forme d’une liste de villes



**Interface**

Quand on appuie sur calcule, calcule l’itinéraire et génère un fichier html contenant une carte avec l’itinéraire (folium). L’itinéraire s’affiche aussi sur la console.

Cet interface peut être amélioré :

On saisit la première lettre pour éviter que toutes les gares apparaissent

**RESSOURCES**

* Le fichier liste-des-gares.csv qui contient la liste des gares de France
* un fichier **Bibliotheque.py** qui contient des fonctions utiles à la lecture d’une table à partir d’un fichier, et aux manipulations sur les tables (tri, extraction, projection)
* Liens en Français documentation folium

<https://iut-info.univ-reims.fr/users/nocent/python/?section=9>

<https://python-visualization.github.io/folium/quickstart.html>

* Le JNB fiche 3 dijkstra, qui contient une fonction utilisant l’algorithme de Dijkstra, qui permet à partir d’un graphe et d’un sommet de départ, de générer la distance vers les sommets d’arrivée
* Un programme de départ qui contient l’interface
* Interface basique utilisant TKinter à compléter (gares\_0.py)

**BAREME** (indicatif)

* Suivi du projet : tâches bien définies pour chaque personne
* Fonctionnalités du programme

- calcul de l’itinéraire

- affichage de l’itinéraire

- interface graphique

- Structure du programme : respecte le cahier des charges, code lisible et commenté, utilise bibliotheque.py, utilisation d’un fichier util\_dijkstra qui contient l’algorithme de dijkstra.

- Oral individuel sur la portion de code écrite

A remplir chaque séance :

Equipe …..

Séance du ………………………………….

Nom scrum master : …………………………………………………

|  |  |
| --- | --- |
| **Nom développeur** |  |
| **Tâche prévue** |  |
| **Tâche réalisée ? oui / non / en partie** |  |
| **Nom développeur** |  |
| **Tâche prévue** |  |
| **Tâche réalisée** |  |
| **Nom développeur** |  |
| **Tâche prévue** |  |
| **Tâche réalisée** |  |
| **Nom développeur** |  |
| **Tâche prévue** |  |
| **Tâche réalisée** |  |

|  |  |
| --- | --- |
| **Nom développeur** |  |
| **Tâche prévue** |  |
| **Tâche réalisée** |  |

A chaque séance, le scrum master devra également rendre les travaux faits par les différents développeurs : soit un programme, soit un schéma ou un texte, etc…