**Prosit 1 : Recherche opérationnelle**

# **Contexte :**

Il y a une tournée des lampadaires quotidienne, l’objectif est d’optimiser la tournée. (1 fois par chaque pont)

# **Mots clés :**

Théorie des graphes : La théorie des graphes est une branche des mathématiques qui étudie les graphes, qui sont des structures composées de nœuds (ou sommets) reliés par des arêtes (ou des liens).

Algorithme (approche algorithmique) : Un algorithme est une série d'étapes systématiques utilisées pour effectuer une tâche ou résoudre un problème donné. L'approche algorithmique consiste à concevoir, analyser et implémenter des algorithmes pour résoudre efficacement divers problèmes informatiques ou mathématiques.

CPU : L'unité centrale de traitement (CPU) est la composante principale d'un ordinateur responsable de l'exécution des instructions et du traitement des données.

Paper board : Un paper board est une surface plane, généralement montée sur un chevalet, utilisée pour afficher des informations lors de présentations, de réunions ou d'enseignements.

Arbre couvrant minimal : Un arbre couvrant minimal d'un graphe est un sous-ensemble d'arêtes qui forment un arbre connecté et qui couvre tous les sommets du graphe avec un coût total minimum.

Complexité : En informatique, la complexité fait référence à la mesure de la quantité de ressources, telles que le temps et l'espace, nécessaires à l'exécution d'un algorithme ou à la résolution d'un problème. La complexité temporelle se réfère au temps d'exécution de l'algorithme en fonction de la taille de l'entrée, tandis que la complexité spatiale se réfère à la quantité de mémoire requise par l'algorithme.

Optimiser : Moins coûteux en temps et en mémoire

# **Problématique :**

Comment faire un code pour ne pas passer 2x par le même pont, pour ne pas repasser dans chaque rue ?

Comment calculer le temps de CPU ?

À partir de quel taille l’algorithme n’est plus utilisable ?

# **Contraintes :**

* 1x par chaque rue
* Temps : pas 2 semaines
* Programme adapté à toutes les villes
* Obliger de renvoyer un circuit optimal même s’il doit passer par la même rue
* Temps d’exécution en ms

# **Livrables :**

Programme informatique (notebook)

# **Généralisation :**

Théorie des graphes

Modélisation d’algorithme

Codage python

Optimisation de code

# **Piste de solution :**

Utilisation de la chaîne eulérienne pour savoir si tu peux passer une seule fois par chaque rue

Faire des boucles pour optimiser le code

Trouver une manière de faire savoir au code les chemins qu’il a déjà empreinté

# **Plan d’action :**

Tester (par calcul) la faisabilité du premier graphe

Faire le graphe papier/crayon

Numéroter les intersections puis réaliser la chaîne eulérienne complètent

Faire le programme

Optimisation du programme

# **Résolution du prosit :**

J’ai d’abord réalisé le schéma des rues. Afin d’avoir un graphe connexe, j’ai négligé l’intersection entre Euler et Ramon.

Une image contenant cercle, capture d’écran, ligne

Description générée automatiquement Une image contenant texte, capture d’écran, Police, ligne

Description générée automatiquement

Une image contenant texte, capture d’écran, Police

Description générée automatiquement

Une image contenant texte, capture d’écran, Police, nombre

Description générée automatiquement

Une image contenant texte, Police, capture d’écran

Description générée automatiquement

On vérifie maintenant le cycle Eulérien :

Une image contenant texte, écriture manuscrite, papier

Description générée automatiquement

Une image contenant cercle, diagramme, ligne, capture d’écran

Description générée automatiquement

# **Correction du prosit :**

Une image contenant carte, diagramme, ligne, cercle

Description générée automatiquement

Une image contenant texte, capture d’écran, Police, nombre

Description générée automatiquement

Une image contenant texte, capture d’écran, Police

Description générée automatiquement

Une image contenant texte, capture d’écran, nombre, Police

Description générée automatiquement

Une image contenant texte, Police, ligne, capture d’écran

Description générée automatiquement

Une image contenant texte, capture d’écran, Police, nombre

Description générée automatiquement

Une image contenant texte, capture d’écran, Police

Description générée automatiquement

Calcul du temps : [Workshop\_1\_Tuteur\_\_Francais\_2024 - Jupyter Notebook](http://localhost:8888/notebooks/Downloads/Workshop_1_Tuteur__Francais_2024.ipynb)