Rapport du Projet Metro Parisien

Présentation du projet :

Le projet Métro Parisien en Algorithmique des Graphes consiste en l'implémentation d'un programme permettant de trouver le plus court chemin entre deux stations données.

Pour cela a été mis en notre possession un fichier texte que nous devrions compléter contenant la liste des sommets et les chemins entre les sommets.

Algorithme choisi : Dijkstra

Langage utilisé : Java

Editeur : BlueJ

Les modifications apportées au fichier « Metro.txt » :

1. Les lignes commençant par « E » ont été complété en rajoutant le numéro de la ligne et le sens c'est-à-dire la direction ;
2. Des stations et chemins manquant ont été rajouté ;
3. Les sommets (stations) ayant plusieurs identifiants ont été lié par des correspondances ;

Découpage du Projet :

Ce projet est découpé en plusieurs classes qui sont :

- **ListeSommets**
- **Chemin**
- **MatriceChemins**
- **Nœud**
- **Dijkstra**
- **Main**
- **Un fichier texte « Metro.txt »**

Description des Classes :

La classe **« ListeSommets » :** cette classe permet de stocker les sommets (stations) dans un tableau de String et comme indice l'identifiant des sommets. En plus du stockage des sommets, elle contient les fonctions **« idenDestination »** et **« idenSource »** qui permettent de renvoyer les identifiants des sommets source et destination que l'utilisateur aura saisi à l'entré.
La classe **« Chemin » :** cette classe permet à son tour de stocker un chemin qui est composé de la distance, la ligne et la direction.

La classe **« MatriceChemins » :** contrairement à la classe **« Chemin »** elle stocke dans l'ensemble des chemins du fichier.

La classe **« Nœud » :** cette classe contient l'identifiant d'un sommet et un variable 'val' (distance à la source) qui détermine sa priorité.

La classe **« Dijkstra » :** est la classe cerveau du programme puis que c'est elle qui permet de trouver le plus court chemin. Elle contient les fonctions :

- update : qui s'occupe de la mise jour de la distance et le prédécesseur de des sommets traité.
- la fonction nextNoeud elle permet de trouver dans une file de priorité le prochain nœud non traité.
- uneEtape qui si le sommet a été traité fait la mise à jour à l'aide de la fonction update de la distance
- distancePlusCourtChemin qui retourne après exécution la distance minimale du sommet source au sommet destination

La classe **« Main » :** qui contient le test de l'algorithme.

Les Difficultés rencontrés :

1. Le traitement du fichier Metro.txt
2. Le codage de Dijkstra
3. L'affichage du plus court chemin

L'amélioration à laquelle nous avons songé est la réalisation de l'algorithme de BiDijkstra.

Réalisation du projet :

Ce projet a été réalisé par **Abdoulaye BALDE** et **Ibrahim BOUDO** étudiant en licence 2 informatique à l'Université de Versailles de Saint Quentin-En-Yvelines.